



Article

# Adaptive Hyperparameter Fine-Tuning for Boosting the Robustness and Quality of the Particle Swarm Optimization Algorithm for Non-Linear RBF Neural Network Modelling and Its Applications

Zohaib Ahmad <sup>1</sup>, Jianqiang Li <sup>1,2</sup>  and Tariq Mahmood <sup>3,\*</sup> <sup>1</sup> Faculty of Information Technology, Beijing University of Technology, Beijing 100024, China<sup>2</sup> Beijing Engineering Research Center for IoT Software and Systems, Beijing 100124, China<sup>3</sup> Faculty of Information Sciences, Vehari Campus, University of Education, Vehari 61100, Pakistan

\* Correspondence: tmsherazi@ue.edu.pk

**Simple Summary:** A radial basis function neural network (RBFNN) is proposed for identifying and diagnosing non-linear systems. The neural network developed was optimized not only for RBFNN output layer parameters, including centers, width, and weights, but also for network size (the count of neurons stored in the hidden layer). Two optimization methods, namely, particle swarm optimization (PSO) and hybrid PSO coupled with a spiral-shaped mechanism (HPSO-SSM), were used to train the RBFNN hyperparameters and network size. Simulation results showed that the suggested technique outperformed other current approaches with respect to prediction precision and network compactness.

**Abstract:** A method is proposed for recognizing and predicting non-linear systems employing a radial basis function neural network (RBFNN) and robust hybrid particle swarm optimization (HPSO) approach. A PSO is coupled with a spiral-shaped mechanism (HPSO-SSM) to optimize the PSO performance by mitigating its constraints, such as sluggish convergence and the local minimum dilemma. Three advancements are incorporated into the hypothesized HPSO-SSM algorithms to achieve remarkable results. First, the diversity of the search process is promoted to update the inertial weight  $\omega$  based on the logistic map sequence. Then, two distinct parameters are trained in the original position update algorithm to enhance the work efficiency of the successive generation. Finally, the proposed approach employs a spiral-shaped mechanism as a local search operator inside the optimum solution space. Moreover, the HPSO-SSM method concurrently improves the RBFNN parameters and network size, building a model with a compact configuration and higher precision. Two non-linear benchmark functions and the total phosphorus (TP) modelling issue in a waste water treatment process (WWTP) are utilized to assess the overall efficacy of the creative technique. The results of testing indicate that the projected HPSO-SSM-RBFNN algorithm performed very effectively.

**Keywords:** RBFNN; non-linear system modelling; time-series prediction; HPSO

**MSC:** 68T20



**Citation:** Ahmad, Z.; Li, J.; Mahmood, T. Adaptive Hyperparameter Fine-Tuning for Boosting the Robustness and Quality of the Particle Swarm Optimization Algorithm for Non-Linear RBF Neural Network Modelling and Its Applications. *Mathematics* **2023**, *11*, 242. <https://doi.org/10.3390/math11010242>

Academic Editors: António Lopes, Omid Nikan and Zakieh Avazzadeh

Received: 4 December 2022

Revised: 26 December 2022

Accepted: 28 December 2022

Published: 3 January 2023



**Copyright:** © 2023 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 (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

It is widely recognized that many real-world problems can be addressed by employing a non-linear model.

Hence, non-linear system modelling has been a consistent focus of research [1]. Several novel methods have been used for the development of non-linear modelling systems, including non-linear regression, multivariate statistical methods, and neural network-based architectures [2,3]. Due to their efficient approximation abilities, research interest in

artificial neural networks (ANNs) has increased enormously [4,5]. In contrast to traditional approaches, ANNs can model non-linear connections between input and output variables.

Artificial network-based techniques are used in several scientific and technical domains and applications, such as non-linear systems modelling, adaptive learning control, and pattern recognition strategies [6–9]. Various neural network topologies are available due to their adaptability and extensive parameter-tuning options. RBFNN algorithms have been widely utilized for modelling and controlling non-linear systems owing to their universal computation capability and structural elegance [10]. It has been demonstrated that the RBFNN approach can simulate any continuous function given adequate RBF neurons [11]. Standard RBFNN learning-based algorithms can address problems in predicting network sizes (the degree of RBF neurons) and updating hyperparameters (i.e., RBF center coordinates, RBF neuron widths, and output weights) to attain the required approximation precision [12,13].

Error back-propagation (BP) is one of the most used methods for modifying the RBFNN architecture [14]. However, the BP approach results in a poor global search experience and the problem of a prolonged convergence rate. Several studies have been conducted to determine the local minimum. Recursive least squares (RLS) may yield a more optimal convergence rate and heightened accuracy than the BP solution [15]. This enhancement necessitates a set of very complex mathematical expressions and greater use of computing resources. The RBFNN size has a substantial effect on its performance and it is important to improve RBFNN size for overall performance enhancement. The adaptive growth and pruning method [16] is used to develop a recurrent neural network, with the hyperparameters fine-tuned to enhance the network's performance. The self-organizing RBFNN network is then trained, while simultaneously modifying the network size and features using forward and backward selection techniques [17].

Evolutionary algorithms (EA) have been used to boost the performance of networks. Xiao et al. [18], and Sheng et al. [19] introduced evolutionary algorithms to develop an RBFNN approach. Feng et al. [20] employed PSO to overcome the widespread convergence time problem encountered while training RBFNNs, resulting in a PSO-based self-generating RBFNN. Alexandridis et al. [21] proposed a new approach based on fuzzy means and PSO algorithms for training RBFNNs. Abdi et al. [22] developed an RBFNN model using a non-linear time-varying evolutionary PSO (NTVE-PSO) technique. An NTVE-PSO method was used to assess the RBFNN's optimal hyperparameters and network size concurrently for time-series prediction problems. Other EA approaches, including the ant colony search algorithm [23], genetic algorithms [24] and artificial immune systems [25] have been applied to prediction problems.

From the perspective of evolutionary approaches, the PSO algorithm has a relatively quick approximation speed for optimal outcomes, enabling it to optimize system parameters efficiently [26–28]. The PSO method continuously updates the particle's position using the particle's current position, global extremum, and individual extremum. The PSO algorithm can meet the challenge of continually optimizing functions. However, converging in advance is more manageable (especially when dealing with highly non-linear optimization problems) because the local optimization potential is restricted. The core formula for position updates must balance exploration and exploitation throughout the probe operation. This might impair the efficacy of the accompanying algorithms. Moreover, the current PSO-based algorithms need not fully use existing optimum solution data to steer the search space.

To address these obstacles, a unique global optimization strategy, namely hybrid PSO combined with a spiral-shaped mechanism (HPSO-SSM) [29,30], is proposed to design RBFNNs with enhanced performance in the application of non-linear-system modelling and prediction. The proposed HPSO-SSM-based RBF neural network (HPSO-SSM-RBFNN) has dual advantages. Firstly, the logistic map sequence is employed in the original PSO method to boost the search strategy's variety. A unique position-updating model is incorporated for

enhancing position quality and a spiral-shaped process. Second, the suggested HPSO-SSM technique optimizes the RBFNN parameters and size.

The key contributions of the present study are as follows:

- To alleviate the limitations of PSO, such as sluggish convergence, eliminate the local minimum dilemma and boost the quality of PSO outcomes, a hybrid HPSO-SSM algorithm is developed.
- Three significant improvements are made to the original PSO: First, a logistic map sequence is used to adjust the inertial weight  $\omega$ , which provides sufficient variety and facilitates the avoidance of optimal solutions throughout the selection process.
- Second, a significantly improved update equation for creating the next-generation position is proposed, which can more effectively integrate exploration and exploitation.
- Third, a spiral-shaped mechanism (SSM) is coupled to the original PSO as a local search strategy for the known optimum solution.
- Finally, the proposed HPSO-SSM approach is applied to optimize the network RBFNN parameters and size. Two non-linear benchmark functions, namely, (1) Lorenz time-series, (2) non-linear system identification and (3) the total phosphorus (TP) modelling problem in a waste water treatment process (WWTP), were tested. Simulation results of the RBFNN trained by HPSO-SSM were obtained to assess its performance with respect to the root mean squared error (RMSE). The presented HPSO-SSM-RBFNN is designed to enable greater solution precision with smaller network size. The findings indicate that this model is superior to other RBFNN models.

The paper is structured as follows: Section 2 explains the operational principles of the RBF neural network. In Section 3 particle swarm optimization (PSO), the hybrid method HPSO-SSM models and the new implementation strategy for HPSO-SSM-RBFNN are described. Section 4 presents the findings for evaluation of several optimization techniques applied to a variety of benchmark situations. The norms obtained are relevant to non-linear structural analysis, time-series prediction of the Lorenz system, and TP estimation in WWTP. These findings are compared to data for existing approaches. Finally, Section 5 highlights and summarizes the study's results.

## 2. Methods and Materials

A unified hybrid approach is proposed for optimizing the hyperparameters and size of RBFNNs. Optimization of neural networks remains a difficult endeavour in the field of machine learning. Traditional training methods are often susceptible to local optima and premature convergence, rendering them inefficient for datasets containing multiple attributes. However, EC-based training algorithms have become more widely considered because of their robustness in overcoming the constraints of traditional methods. In the present study, three benchmark datasets are used to investigate an integrated training algorithm that combines particle swarm optimization (PSO) with a spiral-shaped mechanism (SSM). The proposed logistic map sequence boosts the search process diversity to overcome the conventional PSO algorithm's limitations, such as slow convergence, the local minimum problem, and low-quality results. In addition, an updated algorithm and spiral-shaped method are implemented to enhance the position features.

A novel machine-learning-based algorithm is proposed to handle imbalanced datasets, which can be misleading when training algorithms during the classification task. The proposed approach contributes to improving algorithm performance and reducing the problems of over-fitting and under-fitting. A strategy for creating a master feature vector (MFV) is also presented, which avoids the unfavourable effects of the classification scheme caused by data inconsistencies, missing values, and classification-related issues. The proposed MFV achieves high prediction performance scores, while addressing data inconsistencies, issues related to missing values, and data imbalance problems. The primary purpose of the present study is to boost data quality by introducing a cutting-edge approach to impute missing thresholds and enrich the predictive performance of algorithms.

### 2.1. Datasets

Three experiments were carried out to assess the efficacy of the newly created algorithms. Five hundred input-output data samples were collected from a natural small-scale wastewater treatment plant in China (Beijing) between June 2019 and May 2020. After denoising and normalizing employing the wavelet approach, 250 cases were selected for model training and 150 for testing [31,32]. The proposed algorithm’s performance was improved by discrediting attribute values and allowing missing values. The efficacy of the proposed hybrid HPSO-SSM-based RBF neural networks (HPSO-SSM-RBFNN) was evaluated using Lorenz time-series and non-linear system identification benchmark datasets. Experiments were then conducted on a real application issue for the TP modelling problem in a WWTP to demonstrate the proposed algorithm’s superiority in prediction accuracy.

### 2.2. Data Enhancement

Initial data preprocessing was crucial for achieving the remarkable results obtained in the experiments. Data-processing techniques used included data discretization, the imputation of missing values, and the accommodation of imbalanced data for the robust training of the recommended models. Moreover, essential features were recognized throughout the discretization procedure.

The proposed strategy eliminated noise from the dataset and enhanced the algorithm’s performance.

### 2.3. Cutting-Edge Methods

#### 2.3.1. Improved Particle Swarm Optimization (PSO) Algorithms

PSO is a stochastic searching efficient algorithm that mimics fish schooling and bird flocking within a swarm [33]. Each particle in PSO depicts a possible remedy within the search range. In this study, every particle optimizes its onward path set on its optimal initial position and the most suitable existing location of every particle to obtain the globally optimal solution. A vector indicates the previous position of particle  $i$  during the searching phase, as described in Equation (1).

$$x_i = (x_{i1}, x_{i2}, x_{i3}, \dots, x_{iD}) \tag{1}$$

Here,  $D$  specifies the extent of the pest space and  $N$  swarm size with  $i = 1, 2, \dots, N$ . Every particle has a flight velocity, explained by Equation (2).

$$v_i = (v_{i1}, v_{i2}, \dots, v_{iD}) \tag{2}$$

Each particle’s velocity is based on its acquired memory so that it may explore all of the search areas. Now, consider  $pbest_i$ , the previous best position of the particle  $i$ , is designated as  $pbest_i = pbest_{i1}, pbest_{i2}, pbest_{i3}, \dots, pbest_{iD}$ . Consequently,  $gbest_i = gbest_{i1}, gbest_{i2}, \dots, gbest_{iD}$  is stated to be the optimal position determined by the whole swarm.

The improved PSO algorithm requires adjustment of the flight speed and position every iteration. The updating methods of the  $i^{th}$  particle’s velocity  $v_{id}^{t+1}$  and position  $x_{id}^{t+1}$  of the  $(t + 1)^{th}$  iteration and  $d$ -dimension are defined using Equations (3) and (4).

$$v_{id}^{t+1} = \omega \times v_{id}^t + c_1 \times r_{i1} \times (pbest_{id} - x_{id}^t) + c_2 \times r_{i2} \times (gbest_{id} - x_{id}^t) \tag{3}$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \tag{4}$$

where,  $r_{1i}$  and  $r_{2i}$  are random numbers within 0 and 1.  $\omega$  represents the inertia weight.  $c_1$  and  $c_2$ , are the position constants also known as acceleration constants.

#### 2.3.2. HPSO-SSM Algorithm

Although the PSO approach is effective for discovering an appropriate solution, it has three drawbacks, including early convergence, inadequate imbalance between local and

global search efficiency, and a lack of modification near locally optimal locations. Here, a new fused particle swarm optimization approach melded with a spiral-shaped mechanism (HPSO-SSM) is proposed to overcome the inadequacies of PSO. In the suggested HPSO-SSM approach, the original PSO is modified in three significant ways. The PSO approach uses a logistic map to update the inertia weight  $\omega$  to drive the result's convergence toward the global optimum solution, as represented in Equation (5).

$$\omega^t = \mu \times \omega^{t-1} \times (1 - \omega^{t-1}) \tag{5}$$

Here,  $\mu$  is a positive constant ( $\mu = 4$ ) and  $t$  is the current iteration number. The features of randomness and ergodicity of the sequence boost the algorithm's capacity to avoid locally optimum solutions.

In the proposed HPSO-SSM technique, the particle locations are updated in two distinct ways. Firstly, spiral-shaped mechanisms are used, and, in the second step, the modified standard PSO position update formula is implemented. The designed spiral-shaped mechanism enhances the local recognition ability and maximizes the probability of finding the PSO algorithm's global result during the search phase. The trajectory of the spiral-shaped process is computed based on the distance of the preceding and  $i^{th}$  particle positions in the search area [34]. A suggested mechanism is proposed for developing the next generation of particle locations to describe the two position update techniques.

A probability of 80% based on the results of several investigations was obtained.

The variable  $p$  determines the method using a random value within 0 and 1 selected at every operation.

If the  $p > 0.8$ , the spiral-shaped mechanism is applied; otherwise a modified updating formula is used. Using Equation (6), the spiral-shaped mechanism determines the following location of a particle.

$$x_{id}^{t+1} = |gbest_d - x_{id}^t| \times \exp(b \times l) \times \cos(2\pi l) + gbest_d \tag{6}$$

Here,  $b$  represents a positive constant ( $b = 2$ ), and  $l$  is a random value between  $-1$  and  $1$ . Otherwise, the position is updated using the modified formula, as described in Equation (7).

$$x_{id}^{t+1} = \mathfrak{R}_1^t \times x_{id}^t + \mathfrak{R}_2^t \times v_{id}^{t+1} \tag{7}$$

In Equation (7), the  $\mathfrak{R}_1$  and  $\mathfrak{R}_2$  are dynamic correction factors, introduced in the HPSO-SSM algorithm to control the trade-off between exploration and exploitation throughout the search process. Since, in the HPSO-SSM algorithm, the position values are limited to the range between 0 and 1, the formulas for the dynamic correction factors were slightly modified, to preserve their functionality in the domain of the chosen objective function, as illustrated in Equations (8)–(11).

$$\mathfrak{R}_1^t = \frac{1}{\left(1 + \exp\left(a \times \left(-\frac{\min(SP)}{\max(SP)}\right)\right)\right)^t} \tag{8}$$

$$\mathfrak{R}_2^t = 1 - \mathfrak{R}_1^t \tag{9}$$

$$SP_i^t = \frac{x_i^{t-1}}{M} \times \left[\frac{x_i^{t-1}}{M}\right]^T \tag{10}$$

$$M = \max(|A_{min}|, |A_{max}|) \tag{11}$$

where  $a$  is a constant ( $a = 2$ ) and  $SP$  is the modified position magnitude formula.  $M$  is the normalization factor in which  $A_{min}$  and  $A_{max}$  denote the boundaries of the search space. The flow diagram of the developed HPSO-SSM approach is shown in Figure 1.

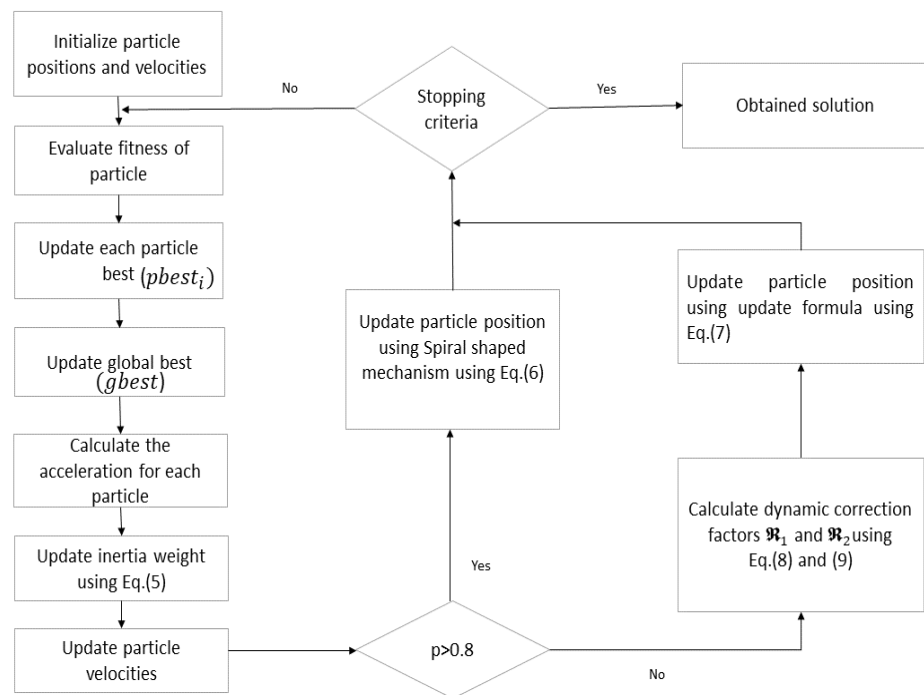


Figure 1. Step-wise description of the suggested HPSO-SSM algorithm.

2.3.3. RBFNN Architecture

The RBFNN is a straightforward network architecture with robust non-linear approximation capability, resulting in an efficient learning rate. The suggested model is based on a multi-layer architecture comprising input, hidden, and output layers, as depicted in Figure 2.

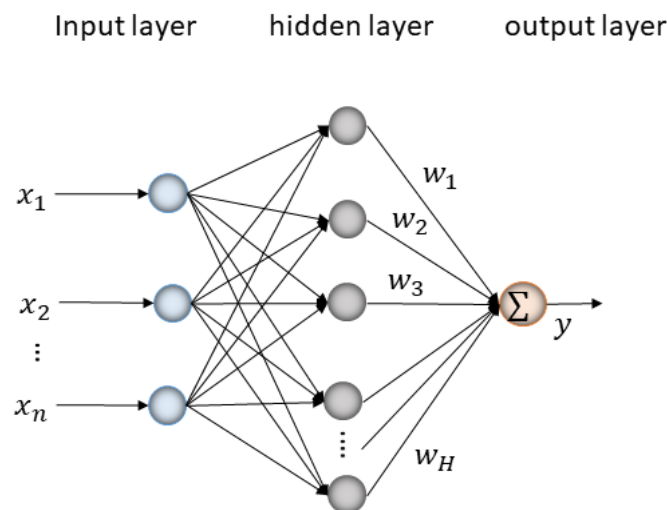


Figure 2. The multi-layer architecture of RBFNN algorithms comprises an input layer, hidden layer, and output layer.

The RBF architecture transfers the vector from the input to the hidden layer. RBF handles the concealed layer and has solid non-linear fitting capability. The hidden layer node remakes the non-linear operation into a high-dimensional linearly separable space with the aid of the basis function. The final predicted finding is yielded by linearly merging

the outcomes of the hidden nodes. A Gaussian function is utilized as the RBF in the hidden layer. Equation (12) illustrates finding of the  $i^{th}$  hidden node.

$$\phi_i(x) = R\|x - c_i\| = \exp\left(-\frac{\|x - c_i\|}{2\sigma_i^2}\right) \tag{12}$$

Here,  $x$  represents the inputs of RBFNN;  $c_i$  and  $\sigma_i$  represent the hidden node’s centre vector and width value, respectively. Equation (13) shows the RBFNN predicted output  $\bar{y}$ .

$$\bar{y} = \sum_{i=1}^H \omega_i \Phi_i(x) \tag{13}$$

In Equation (13),  $\omega_i$  represents the weight of the hidden nodes and  $i = 1, 2, \dots, H$ ,  $H$  designates the count of hidden nodes. During the learning phase, parameters ( $c_i, \sigma_i, \omega_i$ ) and the hidden node are fine-tuned using the training data samples. The HPSO-SSM strategy is proposed to determine the best combination of these parameters iteratively.

### 3. Proposed HPSO-SSM-RBFNN Architecture

This section explores the HPSO-SSM-RBF-based neural network (HPSO-SSM-RBFNN) with configurable network scope and parameters. During the learning phase, the position of the  $i^{th}$  particle is described in Equation (14).

$$x_i = [c_{i,1}^T, \sigma_{i,1}, \omega_{i,1}, c_{i,2}^T, \sigma_{i,2}, \omega_{i,2} \dots c_{i,H_i}^T, \sigma_{i,H_i}, \omega_{i,H_i}] \tag{14}$$

In Equation (14), the  $H_i$  denotes network size (number of hidden neurons) and  $D_i = (2 + n)H_i$  describes the dimension of the  $i^{th}$  particle. The HPSO-SSM-RBFNN approach searches for the optimal particle to select the optimal RBFNN between all particles. The dimensions of other particles may also be modified, as specified in Equation (15)

$$H_i = \begin{cases} H_i - 1 & \text{if } H_{best} < H_i \\ H_i - 1 & \text{if } H_{best} \geq H_i \end{cases} \tag{15}$$

here,  $H_{best}$  represents the best particle’s network size.

The following are the HPSO-SSM-RBFNN functioning procedures.

**Step. 1:** Initialize algorithm parameters, comprising the volume of swarm  $N$ , the boundary of position  $x_{max}$  and velocity  $v_{max}$ , the acceleration coefficients  $c_1$  and  $c_2$  and the maximum iteration number  $t = 1$ . For each particle  $i$ , initialize  $t = 1$  and pick two  $D$ -dimensional vectors at random to initialize the  $x_i$  position and  $v_i$  velocity of each particle  $i$ .

**Step. 2 Encoding:** The RBF algorithm is bound by each rule, including precedent and subsequent parameters; hence, with the centre  $c_i$ , the width  $\sigma_i$ , and output weight  $\omega_i$ , as well as the network size, are evenly encoded into an individual. All individuals could characterise the RBFNN structure.

**Step. 3:** Each particle’s fitness value reflects the network’s precision. The RBFNN hyperparameters and network size are optimized by determining the fitness value of every particle depending on the error criteria and network size. The less the fitness value, the more superior the  $i^{th}$  approach, as explained in Equation (16).

$$f(x_i(t)) = E_i(t) + aH_i(t) \tag{16}$$

Here,  $f(x_i(t))$  shows the cost value of  $i^{th}$  particle,  $H_i$  is the network size, and  $E_i(k)$  is the root mean square error (RMSE). The RMSE is applied to assess the efficacy of the network, as in Equation (17).

$$E_i(t) = \sqrt{\frac{1}{T} \sum_{t=1}^T (y(t) - y_d(t))^2} \tag{17}$$

Here,  $y_d(t)$  represents the required output,  $y(t)$  is the predicted network output, and  $T$  is number of data instances.

**Step. 4:** Compute the fitness function for every particle  $i$  according to Equation (17). Set each particle's current position as the personal fine fitness  $gbest_i$ . Then, as the global finest fitness  $gbest$  of the whole swarm, estimate the finest fitness value.

**Step. 5:** Evaluate the velocities and positions of each of the particles outlined in Equations (3) and (4).

**Step. 6:** Update the inertia weight  $\omega$  using Equation (5).

**Step. 7:** Based on the results of many tests, the likelihood is set at 80%. The particle position is updated by SSM using Equation (6).

**Step. 8:** Determine whether the current iterations exceed the highest number of iterations. If that happens, the method terminates and produces an optimum solution. Then, orders  $t = t + 1$ , returns to Step 3, and enters into step9.

**Step. 9:** Calculate dynamic correction factors  $\mathfrak{R}_1$  and  $\mathfrak{R}_2$  using Equations (8) and (9).

**Step. 10:** Update the particle's position using update formula Equation (7).

**Step. 11:** Update the network size  $H_i$  using Equation (15).

**Step. 12:** Select the optimal particle position as the RBFNN structural parameter for the network topology. Train the network to develop an RBFNN suitable for predicting non-linear time-series.

The pseudo-code of the proposed HPSO-SSM-RBFNN model is represented in Algorithm 1.

---

**Algorithm 1:** Pseudo-code of HPSO-SSM-RBFNN

---

**Input** : Population size  $N$ , maximum number of iterations for optimization  $T$ , dimensionality of decision variables

**Output:** trained HPSO-SSM-RBFNN

- 1 Initialize algorithm relative parameters,  $c_1, c_2, v_{min}, v_{max}$ , population size  $N$ , and maximal iterations  $T$ .
  - 2 Construct the topology structure of RBFNN for every particle and initialize the position of the particles.
  - 3 Calculate the dimension of the particle  $D_i(t)$
  - 4 Estimate each particle with fitness value (using Equation (16))
  - 5 Assess the RMSE (using Equation (17))
  - 6 Select the best position of particles as  $gbest$  from the swarm
  - 7 Set the position of each particle as  $pbest_i$
  - 8  $T = 1$  to maximum iterations
  - 9 **for**  $i = 1$  to  $N$  **do**
  - 10 | Update velocity  $v_i^{(t+1)}$  and position  $x_i^{(t+1)}$  of all particles (using Equations (3) and (4))
  - 11 | Compute the inertia weight  $\omega$  (using Equation (5))
  - 12 **end**
  - 13 Define the awareness probability.
  - 14 **if**  $p \leq 0.8$  **then**
  - 15 | Calculate dynamic correction factors  $\mathfrak{R}_1$  and  $\mathfrak{R}_2$  (using Equations (8) and (9))
  - 16 | Update particle's next position using updated formula (using Equation (7))
  - 17 **end**
  - 18 **else**
  - 19 | Update the particle's next position using SSM (using Equation (6)) Update network size  $H_i$  (using Equation (15))
  - 20 **end**
  - 21 Attain optimal value
-



## 4. Results Analysis and Discussion

### 4.1. Experimental Discussion

The simulation results for three benchmark problems for the proposed RBFNN and hybrid HPSO-SSM algorithm were compared with those of other state-of-the-art models. These problems involve non-linear model identification, time-series prediction of the Lorenz system, and TP estimation in WWTP. Several experiments were carried out to verify the effectiveness of the HPSO-SSM-RBFNN, including consideration of two benchmark optimization problems and TP estimation in WWTP. All datasets were normalized to the range  $[-1, 1]$ . The training datasets were gathered to construct the system model and testing datasets were utilized to assess the model's generalization competency. In addition, the RMSE derived from Equation (17) was used to intuitively evaluate the performance of HPSO-SSM-RBFNN. Many indicators were used for evaluation, including the values for training and testing RMSE and the number of hidden neurons stored. The superior results confirm that the suggested technique achieves greater solution precision with a smaller network size than other RBFNN models. Each experiment was carried out on the MATLAB R2016b platform. Each experiment was assessed using an Intel(R) Core processor i7 with a 3.8GHz, 16GB of RAM, and GTX 1020Ti NVIDIA graphics card.

### 4.2. Model Hyper-Parameters Fine-Tuning

The network architecture of HPSO-SSM-RBFNN is critical in the training phase, especially for the network size ( $H_i$ ) and network parameters ( $c_i, \sigma_i, \omega_i$ ). These parameters are set according to the corresponding tasks. To optimize the RBFNN, the PSO and HPSO-SSM algorithms involve numerous control parameters, such as the number of iterations for optimization, the number of particles, the inertia weight, acceleration coefficients, and random values linked to the social and cognitive features of the velocity update equation. Additionally, the maximum velocity and coefficient of constriction are determined to be mainly operational control parameters if velocity clamping and constriction are utilized [35]. These factors greatly impact the solution's performance, quality, and convergence rate. To avoid early convergence, it is necessary to fine-tune the parameters. Han et al. [36,37] proposed efficient neural network architectures for simulating uncertain non-linear dynamics. The suggested strategy increased the particle's searching capabilities by fine-tuning the network parameters and modifying the inertia weights concurrently with the network size throughout the learning processes. The experimental findings suggest that the proposed methods are more effective and have significant potential for real-world application, such as in the dynamical estimation of crucial parameters in WWTP systems and the adaptive control of uncertain non-linear systems.

In the HPSO-SSM algorithm, the initial parameter settings, including the population size, were fixed at 30, while the maximum number of iterations for optimization was dynamically adjusted in each testing experiment. In the first testing experiment, the prediction of Lorenz time-series, the iterations were set to 300; in the remaining two testing experiments, the iterations were set to 200. Every particle's minimum  $v_{min}$  and maximum velocity  $v_{max}$  were assigned in the interval of  $[-1 - 1]$ . Conversely, every particle's minimum position  $x_{min}$  and maximum position  $x_{max}$  were assigned in the interval of  $[-5, 5]$ . Additionally, control parameters were applied to the PSO and HPSO-SSM algorithms. To achieve optimal results, the position constants  $c_1$  and  $c_2$  parameters required to be adjusted during the proposed algorithm iterations; the ideal range for these values was chosen to be in the range  $[0.5-2.5]$ . The starting value of the inertia weight  $\omega$  was set to 0.5. Each simulation experiment employed the proposed methodology with these parameters adjusted to guarantee consistent results. The parameters used in the PSO and HPSO-SSM algorithms are presented in Table 1. Setting the parameters of the RBFNN model requires expert knowledge of the process, such as setting the initial number of hidden nodes to 3 in the first and third testing experiment and 6 in the second testing experiment. All the experiments were performed several times; the overall average was determined to avoid the consequences of random initialization of particular network parameters. In HPSO-

SSM-RBFNN, membership function parameters of each rule of RBFNN were encoded into distinct particle position features. Each particle represents a particular RBFNN structure. Particle swarm optimization proceeded until the ideal individual was determined. Among the 40 best particles, the one with the smallest RMSE was chosen as the optimal RBFNN.

**Table 1.** PSO and HPSO-SSM parameter fine-tuning configuration.

Configuration	Value
Population Size	40
Iterations (Lorenz time-series)	300
Iterations (Non-linear system identification)	200
Iterations (TP prediction in WWTP)	200
Inertia weight $\omega$	0.5
Constriction coefficient (C)	1
$r_1, r_2$	Random in [0, 1]
$c_1, c_2$	Dynamic adjusts
HPSO Positive constants ( $\mu, a, b$ )	4, 2, 2

### 4.3. Lorenz Time-Series Predicting Scheme

The Lorenz series is utilized as a standard in different applications for various problems [38]. The Lorenz series example is a collection of numerical models for atmospheric convection.

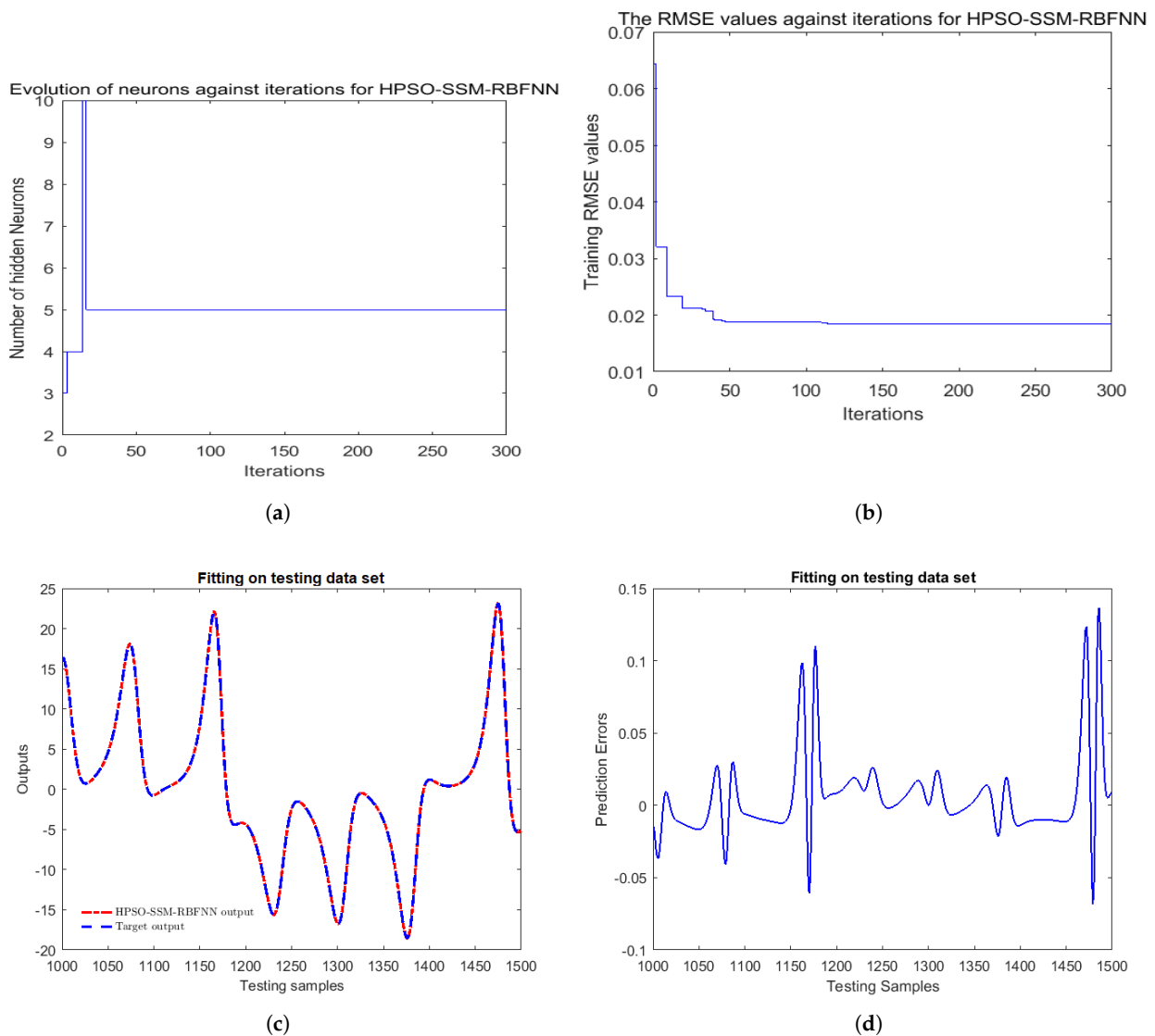
The three-dimensional (3-D) and non-linear framework of the Lorenz series are defined as in Equation (18).

$$\begin{cases} \frac{dx}{dt} = a(-x + y) \\ \frac{dy}{dt} = bx - y + xz \\ \frac{dz}{dt} = xy - cz \end{cases} \tag{18}$$

The model’s parameters are adjusted to  $a = 10, b = 28$  and  $c = 8/3$ . The Kutta–Runge method with phase 0.01 generates values for the Lorenz time sequence.  $y(k - 3), y(k - 2)$  and  $y(k - 1)$ , and  $y(k - 1)$  are employed to predict  $y(k)$  for each training pair. A total of 1500 samples were collected for model evaluation, of which 1000 were used for the training and 500 for the testing set.

There were initially three hidden neurons in the suggested approach and the pre-determined training RMSE was set at 0.01. Figure 3 shows the RMSE findings, such as the number of hidden neurons during training, the predicted outcomes, and the predicted error between the target findings and the proposed HPSO-SSM-RBFNN algorithm findings.

Figure 3a depicts the findings of hidden neurons against algorithm iterations during the training period. As shown in Figure 3a, the final number of hidden neurons was approximately five. The training RMSE values of the proposed approach against algorithm iterations are shown in Figure 3b, gradually decreasing and finally stabilizing at about 0.0107. The obtained values highlight the proposed approach’s advantages with respect to convergence speed and modelling precision. Figure 3c shows the prediction outputs of the proposed approach HPSO-SSM-RBFNN against the target outputs. Figure 3c shows that the prediction results based on HPSO-SSM-RBFNN accurately match the real pattern of the Lorenz time-series test samples 1000 – 1500. Figure 3d highlights the testing error findings for HPSO-SSM-RBFNN, showing that the suggested HPSO-SSM-RBFNN algorithms produced excellent results, with a prediction error for the test samples 1000–1500 falling within the range  $[-0.03, 0.12]$ .



**Figure 3.** The presented approach’s efficacy is measured in terms of (a) comparison of hidden neurons for the training of proposed and other conventional algorithms, (b) RMSE values during the training phase, (c) predicted outcomes of HPSO-SSM-RBFNN during the testing phase, (d) the prediction errors during the testing phase.

4.4. Non-Linear Model Identification

The non-linear models are applied to realistically characterize the neural network performance [39], as expressed by Equation (19).

$$y \times (t + 1) = 0.72y \times (t) + 0.025y \times (t - 1) \times u(t - 1) + 0.01u^2 \times (t - 2) + 0.2u \times (t - 3) \tag{19}$$

Two inputs,  $u(t)$ ,  $y(t)$  and  $y(t + 1)$  are required for Equation (19).

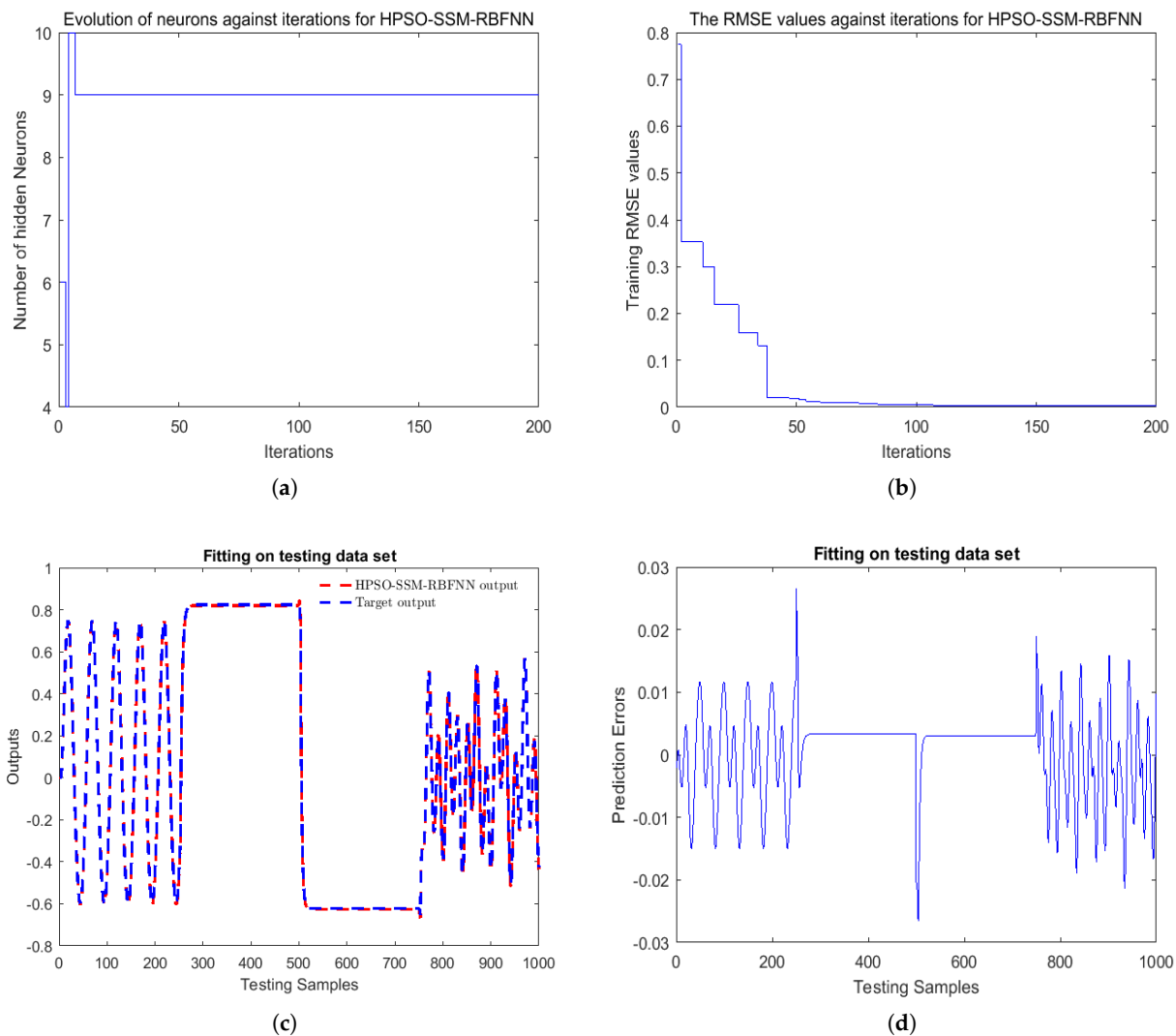
The training inputs are constantly classified for approximately half of the training time throughout the interval  $[-2, 2]$ .

In contrast, the rest of the training period comprises a single sinusoid signal emitted by  $1.5 \times \sin(t/45)$ . This method uses 1000 sample sizes earmarked for the training and testing

phases. The check input signal is described as in Equation (19) to evaluate the specific proof outcomes for the test signals.

$$u(t) = \begin{cases} \sin(\frac{\pi}{25}), & 250 > t > 0 \\ 1.00 & 500 \geq t \geq 250 \\ -1.00 & 750 \geq t \geq 500 \\ 0.3 \sin \times ((\frac{\pi t}{25}) + 0.1((\frac{\pi t}{32}) + 0.6((\frac{\pi t}{10}))), & 1000 \geq t \geq 750 \end{cases} \quad (20)$$

In the training stage, the HPSO-SSM-RBFNN algorithm and the training dataset are used to obtain the trained network. In the evolving phase of the proposed approach, the training RMSE value is set as the fitness function. The results are displayed in Figure 4a–d.



**Figure 4.** The presented approach’s efficacy is measured in terms of (a) the number of hidden neurons during training, (b) RMSE values in the training phase, (c) the predicted results by HPSO-SSM-RBFNN during the testing phase, (d) prediction errors during the testing phase.

Figure 4a shows the findings for comparison of hidden neurons with the algorithm iterations.

The initial number of hidden neurons in the proposed approach was set to six and its final number was nine. The training RMSE values of the proposed approach against algorithm iterations are presented in Figure Figure 4b. The training RMSE values gradually

decrease and finally stabilize at about 0.0029, demonstrating that the proposed approach can approximate the desired accuracy, while maintaining rapid convergence speed.

The testing data sample was utilized to assess the generalisation effectiveness of the designed HPSO-SSM-RBFNN algorithm after network training. The one-step-ahead prediction findings of HPSO-SSM-RBFNN for test samples 0 – 1000 are shown in Figure Figure 4c. Figure 4c shows that the prediction results based on HPSO-SSM-RBFNN accurately match the system output. Figure 4d illustrates that most testing errors for HPSO-SSM-RBFNN were confined to the range  $[-0.03, 0.03]$ , indicating that the suggested approach showed superior testing performance.

4.5. Total Phosphorus (TP) Prognosis in WWTP

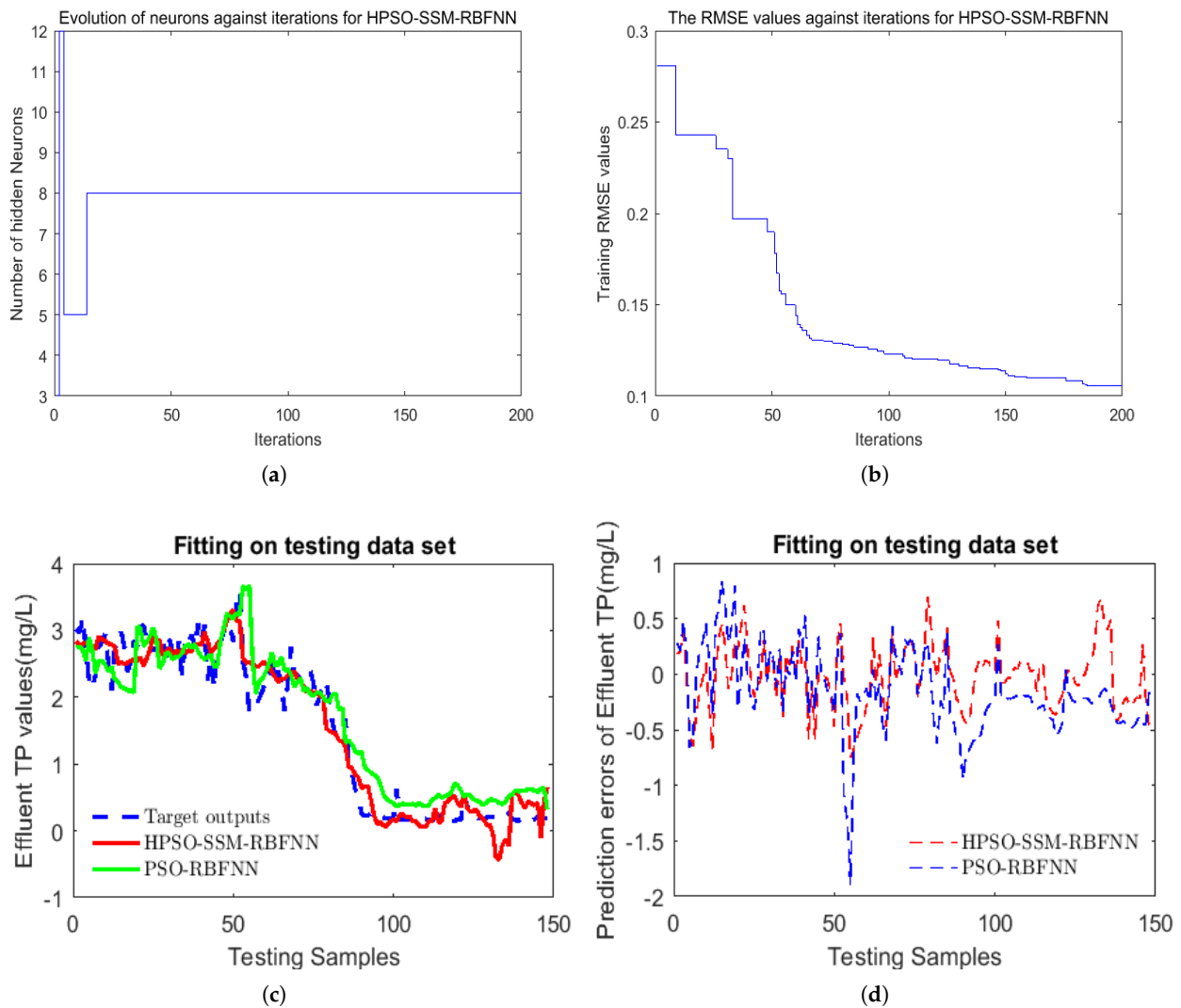
Wastewater treatment techniques are one of best options for tackling the challenge of water contamination. The TP quantity is the fundamental parameter for determining the performance of wastewater processes. The online computation of suitable TP values is challenging in wastewater treatment. The developed HPSO-SSM-RBFNN was used to estimate the TP value. Several easily observable measurements were selected as input variables for the HPSO-SSM-RRBFNN, including pH, the concentration of dissolved oxygen (DO), temperature (T), total suspended solids (TSS), and the influent TP. Experimental datasets for the training of models were obtained from a wastewater treatment plant in Beijing, China. After removing anomalous data, 250 instances for training and 150 samples were utilized for testing. Figure 5a–d shows the experimental findings for the training and testing approach.

Figure 5a shows the number of hidden neurons throughout the training period. With eight hidden neurons, the designed HPSO-SSM-RBNN provided the most compact framework of all the algorithms. Figure 5b shows the RMSE value of the suggested algorithms during the training phase. The developed algorithm gave the smallest root mean square error when comparing the training RMSE of the HPSO-SSM-RBFNN. In Figure 5c, the TP values of the two prediction models, i.e., HPSO-SSM-RBFNN and the PSO-RBFNN are plotted together with the actual values for the test data samples 0 – 150. Figure 5d illustrates the modelling capability of the HPSO-SSM-RBFNN by displaying the error rates of these two models. Figure 5d indicates that the proposed HPSO-SSM-RBFNN showed higher prediction performance with the range of predicted error  $[-0.2, 0.2]$ .

In Table 2, the prediction values are compared for RBFNN [16], PSO-RBFNN [20], and AI-PSO-RRBFNN [40] to demonstrate the performance of the designed HPSO-SSM-RBFNN. The suggested HPSO-SSM-RBFNN produced the lowest (0.103) testing RMSE value, indicating that the suggested technique performed better than existing algorithms for TP value prediction in the WWTP.

Table 2. Comparing the performance of suggested algorithms with conventional approaches.

Algorithms	Lorenz Time-Series Pre.			Nonlinear System Ident.			TP Prediction in WWTP		
	No. of HN	RMSE for Training	RMSE for Testing	No. of HN	RMSE for Training	RMSE for Testing	No. of HN	RMSE for Training	RMSE for Testing
RBFNN [16]	10	0.073	0.098	10	0.073	0.098	10	0.227	0.409
PSO-RBF [20]	9	0.061	0.054	9	0.013	0.025	14	0.188	0.156
AI-PSO-RBFNN [40]	6	0.063	0.058	11	0.018	0.022	12	0.191	0.170
HPSO-SSM-RBFNN	5	0.017	0.013	9	0.0029	0.010	8	0.162	0.103



**Figure 5.** The presented approach’s efficacy is measured in terms of (a) neurons buried during training, (b) the RMSE values in the training phase, (c) prediction results during the testing phase, (d) the proposed HPSO-SSM-RBFNN for total phosphorus (TP) prediction in WWTP.

**4.6. Comparative Analysis with Cutting-Edge Methods**

The effectiveness of the presented algorithms was evaluated for RBFNN [16], PSO-RBFNN [20], and AI-PSO-RBFNN [40], as summarized in Table 2. Various indices were selected for performance evaluation, including hidden neurons stored, and the training and testing RMSE values. HPSO-SSM-RBFNN achieved the most compact network topology, with six, nine, and eight hidden neurons and the lowest training and testing RMSE values compared to the other standard techniques. In addition, HPSO-SSM-RBFNN is known for its superior prediction and generalization capabilities. Hence, the experimental findings demonstrate that the HPSO-SSM-RBFNN shows remarkable identification capabilities with a compact network architecture.

**5. Conclusions and Future Work**

An RBF computational approach for the detection of non-linear models is described. The presented approach is based on the PSO convergence capacity and incorporates a spiral search mechanism. The hybrid HPSO-SSM algorithm can simultaneously boost the parameter and network topology of the RBFNN during the learning process. Performance assessment of the suggested approach HPSO-SSM-RBFNN indicated a compact neural

network. Experimental findings obtained indicate that the suggested strategy was effective in minimizing prediction error and RMSE values relative to existing RBFNNs. The results of the investigation are relevant to real-world applications. The number of simulation results for different time series and different engineering models indicates potential for real-world applications. The HPSO-SSM-RBFNN can calculate dynamical estimates of essential parameters in WWTP systems and govern uncertain non-linear systems adaptively. In the future, we intend to examine the effectiveness of the HPSO-SSM in training various kinds of neural networks. Moreover, techniques to encourage use of the algorithm should be considered.

**Author Contributions:** Conceptualization, Z.A.; methodology, Z.A.; validation, J.L.; formal analysis, Z.A. and T.M.; investigation, T.M., resources, J.L.; data curation, T.M.; writing—original draft, Z.A. and T.M.; visualization, T.M.; supervision, J.L. and T.M.; project administration, Z.A. and T.M.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This study is supported by the National Key R&D Program of China with the project no. 2020YFB2104402.

**Data Availability Statement:** Data is available and can be provided over the emails querying directly to the author (ahmedzohaib03@gmail.com).

**Acknowledgments:** The authors would like to thank China's National Key R&D Program for providing the experimental facilities used to perform these experiments.

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

## References

1. Qiao, J.; Wang, G.; Li, X.; Li, W. A self-organizing deep belief network for nonlinear system modeling. *Appl. Soft Comput.* **2018**, *65*, 170–183. [\[CrossRef\]](#)
2. Chen, G.Y.; Gan, M.; Chen, C.P.; Li, H.X. Basis function matrix-based flexible coefficient autoregressive models: A framework for time series and nonlinear system modeling. *IEEE Trans. Cybern.* **2019**, *51*, 614–623. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Qiao, J.; Wang, L.; Yang, C. Adaptive lasso echo state network based on modified Bayesian information criterion for nonlinear system modeling. *Neural Comput. Appl.* **2019**, *31*, 6163–6177. [\[CrossRef\]](#)
4. Kuriscak, E.; Marsalek, P.; Stroffek, J.; Toth, P.G. Biological context of Hebb learning in artificial neural networks, a review. *Neurocomputing* **2015**, *152*, 27–35. [\[CrossRef\]](#)
5. Jordan, M.I.; Mitchell, T.M. Machine learning: Trends, perspectives, and prospects. *Science* **2015**, *349*, 255–260. [\[CrossRef\]](#)
6. Adhikari, R. A neural network based linear ensemble framework for time series forecasting. *Neurocomputing* **2015**, *157*, 231–242. [\[CrossRef\]](#)
7. Mahmood, T.; Li, J.; Pei, Y.; Akhtar, F.; Imran, A.; Yaqub, M. An automatic detection and localization of mammographic microcalcifications ROI with multi-scale features using the radiomics analysis approach. *Cancers* **2021**, *13*, 5916. [\[CrossRef\]](#)
8. Nguyen, H.N.; Zhou, J.; Kang, H.J. A calibration method for enhancing robot accuracy through integration of an extended Kalman filter algorithm and an artificial neural network. *Neurocomputing* **2015**, *151*, 996–1005. [\[CrossRef\]](#)
9. Mahmood, T.; Li, J.; Pei, Y.; Akhtar, F. An Automated In-Depth Feature Learning Algorithm for Breast Abnormality Prognosis and Robust Characterization from Mammography Images Using Deep Transfer Learning. *Biology* **2021**, *10*, 859. [\[CrossRef\]](#)
10. Huang, S.C.; Do, B.H. Radial basis function based neural network for motion detection in dynamic scenes. *IEEE Trans. Cybern.* **2013**, *44*, 114–125. [\[CrossRef\]](#)
11. Yang, Y.; Wang, P.; Gao, X. A Novel Radial Basis Function Neural Network with High Generalization Performance for Nonlinear Process Modelling. *Processes* **2022**, *10*, 140. [\[CrossRef\]](#)
12. Faris, H.; Aljarah, I.; Mirjalili, S. Evolving radial basis function networks using moth–flame optimizer. In *Handbook of Neural Computation*; Elsevier: Amsterdam, The Netherlands, 2017; pp. 537–550.
13. Iqbal, S.; Qureshi, A.N.; Ullah, A.; Li, J.; Mahmood, T. Improving the Robustness and Quality of Biomedical CNN Models through Adaptive Hyperparameter Tuning. *Appl. Sci.* **2022**, *12*, 11870. [\[CrossRef\]](#)
14. Xu, Z.B.; Zhang, R.; Jing, W.F. When does online BP training converge? *IEEE Trans. Neural Netw.* **2009**, *20*, 1529–1539. [\[PubMed\]](#)
15. Chen, B.; Zhao, S.; Zhu, P.; Principe, J.C. Quantized kernel recursive least squares algorithm. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *24*, 1484–1491. [\[CrossRef\]](#) [\[PubMed\]](#)
16. Han, H.G.; Zhang, S.; Qiao, J.F. An adaptive growing and pruning algorithm for designing recurrent neural network. *Neurocomputing* **2017**, *242*, 51–62. [\[CrossRef\]](#)
17. Qiao, J.F.; Han, H.G. Identification and modeling of nonlinear dynamical systems using a novel self-organizing RBF-based approach. *Automatica* **2012**, *48*, 1729–1734. [\[CrossRef\]](#)

18. Wu, X.J.; Zhu, X.J.; Cao, G.Y.; Tu, H.Y. Modeling a SOFC stack based on GA-RBF neural networks identification. *J. Power Sources* **2007**, *167*, 145–150. [[CrossRef](#)]
19. Chen, S.; Hong, X.; Harris, C.J. Particle swarm optimization aided orthogonal forward regression for unified data modeling. *IEEE Trans. Evol. Comput.* **2010**, *14*, 477–499. [[CrossRef](#)]
20. Feng, H.M. Self-generation RBFNs using evolutionary PSO learning. *Neurocomputing* **2006**, *70*, 241–251. [[CrossRef](#)]
21. Alexandridis, A.; Chondrodima, E.; Sarimveis, H. Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2012**, *24*, 219–230. [[CrossRef](#)]
22. Abdi, H.; Moradi, M.; Rashidi, R. Hybrid transmission expansion planning and reactive power planning considering the real network uncertainties. *Int. J. Numer. Model. Electron. Netw. Devices Fields* **2022**, *35*, e2937. [[CrossRef](#)]
23. Zhou, X.; Ma, H.; Gu, J.; Chen, H.; Deng, W. Parameter adaptation-based ant colony optimization with dynamic hybrid mechanism. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105139. [[CrossRef](#)]
24. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126. [[CrossRef](#)] [[PubMed](#)]
25. Kanwal, S.; Hussain, A.; Huang, K. Novel Artificial Immune Networks-based optimization of shallow machine learning (ML) classifiers. *Expert Syst. Appl.* **2021**, *165*, 113834. [[CrossRef](#)]
26. Bhatasana, M.; Marconnet, A. Machine-learning assisted optimization strategies for phase change materials embedded within electronic packages. *Appl. Therm. Eng.* **2021**, *199*, 117384. [[CrossRef](#)]
27. Fathi, V.; Montazer, G.A. An improvement in RBF learning algorithm based on PSO for real time applications. *Neurocomputing* **2013**, *111*, 169–176. [[CrossRef](#)]
28. Han, M.; Fan, J.; Wang, J. A dynamic feedforward neural network based on Gaussian particle swarm optimization and its application for predictive control. *IEEE Trans. Neural Netw.* **2011**, *22*, 1457–1468. [[CrossRef](#)]
29. Chen, K.; Zhou, F.Y.; Yuan, X.F. Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. *Expert Syst. Appl.* **2019**, *128*, 140–156. [[CrossRef](#)]
30. Kareem, S.S.; Mostafa, R.R.; Hashim, F.A.; El-Bakry, H.M. An effective feature selection model using hybrid metaheuristic algorithms for IoT intrusion detection. *Sensors* **2022**, *22*, 1396. [[CrossRef](#)]
31. Huang, M.; Zhang, T.; Ruan, J.; Chen, X. A new efficient hybrid intelligent model for biodegradation process of DMP with fuzzy wavelet neural networks. *Sci. Rep.* **2017**, *7*, 41239. [[CrossRef](#)]
32. Bao, Y.; Tian, Q.; Chen, M. A weighted algorithm based on normalized mutual information for estimating the chlorophyll-a concentration in Inland Waters Using Geostationary Ocean Color Imager (GOCI) data. *Remote. Sens.* **2015**, *7*, 11731–11752. [[CrossRef](#)]
33. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995; pp. 39–43.
34. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [[CrossRef](#)]
35. Engelbrecht, A.P. *Computational Intelligence: An Introduction*; John Wiley & Sons: Hoboken, NJ, USA, 2007.
36. Han, H.G.; Lu, W.; Hou, Y.; Qiao, J.F. An adaptive-PSO-based self-organizing RBF neural network. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *29*, 104–117. [[CrossRef](#)]
37. Han, H.; Wu, X.; Zhang, L.; Tian, Y.; Qiao, J. Self-organizing RBF neural network using an adaptive gradient multiobjective particle swarm optimization. *IEEE Trans. Cybern.* **2017**, *49*, 69–82. [[CrossRef](#)] [[PubMed](#)]
38. Chen, H.; Gong, Y.; Hong, X. Online modeling with tunable RBF network. *IEEE Trans. Cybern.* **2013**, *43*, 935–947. [[CrossRef](#)] [[PubMed](#)]
39. Han, H.; Zhou, W.; Qiao, J.; Feng, G. A direct self-constructing neural controller design for a class of nonlinear systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 1312–1322. [[CrossRef](#)]
40. Nickabadi, A.; Ebadzadeh, M.M.; Safabakhsh, R. A novel particle swarm optimization algorithm with adaptive inertia weight. *Appl. Soft Comput.* **2011**, *11*, 3658–3670. [[CrossRef](#)]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.