Applied Soft Computing Journal 99 (2021) 106925

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Dynamic evolutionary model based on a multi-sampling inherited HAPFNN for an aluminium electrolysis manufacturing system[☆]

Wei Ding[a], Lizhong Yao[b,a,*], Yanyan Li[a,**], Wei Long[a], Jun Yi[b], Tiantian He[c]

[a] School of Mechanical Engineering, Sichuan University, Chengdu 610065, PR China
[b] School of Electrical Engineering, Chongqing University of Science and Technology, Chongqing 401331, PR China
[c] School of Computer Science and Engineering, Nanyang Technological University, 639798, Singapore

A R T I C L E I N F O

Article history:
Received 24 October 2019
Received in revised form 24 August 2020
Accepted 16 November 2020
Available online 24 November 2020

Keywords:
Aluminium electrolysis manufacturing system
Hybrid annealed particle filter
Neural network
Dynamic evolutionary model
Multi-sampling

A B S T R A C T

It is technically difficult to accurately establish a dynamic adaptive model of an aluminium electrolysis manufacturing system (AEMS) because the system has many complex characteristics, such as multiple parameters, dynamic time variance, and a non-Gaussian distribution of process data. Inspired by the overall superiority of particle filtering theory in dealing with non-linear non-Gaussian problems, this paper presents a novel method based on a multi-sampling inherited hybrid annealed particle filter neural network (MSI-HAPFNN). Firstly, the neural network's (NN's) weights and thresholds are used as the state variables of hybrid annealed particle filter; Secondly, the hybrid proposal distribution obtained by sampling the above state variables is employed to replace the posterior proposal distribution in the standard particle filter (PF) algorithm as the importance density function, thereby adjusting the NN's weights and thresholds in real time. Thirdly, the model achieves the features of multi-sampling and inheritance by introducing NN and PF weights, and using adaptive inheritance method. Therefore, this paper systematically proposes the theoretical construction framework and experimental procedure of MSI-HAPFNN. Furthermore, this article also introduces a genetic algorithm to thoroughly evaluate the prediction potential. The proposed model has been tested on the real-world system for aluminium electrolysis manufacturing and compared with several closely related frameworks. The experimental results show that the MSI-HAPFNN model can significantly improve the self-adaptive ability of the object system to working conditions and the prediction accuracy of power consumption, which is helpful in finding optimal design parameters in an AEMS.

1. Introduction

In recent years, electrolytic aluminium products [1] have been widely used in aerospace, transportation, construction, machinery manufacturing and other significant fields and have played a crucial role in the development of the international industry [2]. Notably, the DC power consumption is quite high in aluminium electrolysis manufacturing systems (AEMSs). The research on energy saving technology [3] of AEMS has become a popular direction in the optimization of the electrolytic aluminium industry.

At present, the main ways to reduce the power consumption in an AEMS are as follows: (1) Improve the process equipment of AEMS [4–8]. For example, Allard et al. [4] established an improved heat transfer model of the top of aluminium electrolysis cells. The device used for handling electrolysis cell equipment to produce aluminium by igneous electrolysis was invented by TISON [5]. (2) Use the theories of intelligent modelling and decision parameters optimization to establish the accurate process model of AEMS. Then, employ the model to find best process design parameters without changing the hardware equipment in the AEMS. The fuzzy algorithm proposed by Chen et al. and Yue et al. transformed the qualitative problem of decision parameters into quantitative problems [9,10] and was used to identify and monitor the state of an aluminium electrolysis cell. Liu et al. adopted the quasi-Z-source network [11] to establish a hybrid power supply system for the aluminium electrolysis industry, and Yi et al. [12] used a recurrent neural network, performed information-based aluminium electrolytic modelling, and optimized the method via a machine learning algorithm [13]. A multi-objective optimization algorithm proposed by Yi et al. [14,15] was used to optimize the decision parameters in the aluminium electrolysis process.

☆ This document is the results of the research project funded by the National Natural Science Foundation of China.

* Corresponding author.

** Corresponding author at: School of Electrical Engineering, Chongqing University of Science and Technology, Chongqing 401331, PR China.

E-mail addresses: dingwei1995@stu.scu.edu.cn (W. Ding), cqust-ylz@cqust.edu.cn (L. Yao), yanyan.li@scu.edu.cn (Y. Li), scdxlongwei@yeah.net (W. Long), 2010027@cqust.edu.cn (J. Yi), tiantian.he@ntu.edu.sg (T. He).

https://doi.org/10.1016/j.asoc.2020.106925
1568-4946/© 2020 Elsevier B.V. All rights reserved.

Nomenclature and Abbreviations

NN	Neural network
PF	Particle filter
GA	Genetic algorithm
EKF	Extended Kalman filter
UKF	Unscented Kalman filter
PPD	Posterior proposal distribution
HPD	Hybrid proposal distribution
CV	Cross validation
GP	Gaussian processes
MLR	Multiple linear regression
NLMR	Multiple nonlinear regression
AEMS	Aluminium electrolysis manufacturing system
MSI-HAPFNN	Multi-sampling inherited hybrid annealed particle filter neural network
EKFNN	Extended Kalman filter neural network
UKFNN	Unscented Kalman filter neural network
BPNN	Back propagation neural network
HAPF	Hybrid annealed particle filter
SSE	Sum squared error
MSE	Mean squared error
RMSE	Root mean squared error
MAE	Mean absolute error
MRE	Mean relative error
R	Correlation coefficient

The above two methods are beneficial for improving the performance of an AEMS. However, note that the first method, which involves improving or developing new production equipment, is difficult, requiring a large amount of capital and technical support, and is more suitable for new industrial and mining enterprises. Under the condition of existing production equipment, the second method explores the optimal process decision parameters and realizes optimal process control using intelligent optimization design theory, which can largely reduce the electrolytic power consumption and the energy cost. The latter method is therefore a very popular energy-saving technology for enterprises. The premise of using this technology, however, is to establish a dynamic optimization model [16,17] that accurately reflects the operation principles in an AEMS. This model can centrally analyse various complex decision variables affecting the aluminium electrolysis process, reduce data redundancy, and realistically reflect the internal relations between operating parameters and power consumption. It is difficult to quantitatively analyse the mechanism of the manufacturing system due to the characteristics of an AEMS, such as multiple parameters, dynamic evolution, non-Gaussian distributions and a series of complex physical and chemical reactions in the system. Therefore, it is also quite hard to establish a prediction model that can accurately evaluate the real-time power consumption in an AEMS, which inevitably brings enormous challenges to design optimization.

Under the condition that the mechanism of AEMS is ambiguous, neural network (NN) modelling method is an effective approach to address it. This method does not need to know the complicated internal mechanism of manufacturing system and can obtain the mapping relationship between decision variables and energy consumption only by learning and training series of process data. Moreover, NN modelling is also suitable for the processing of large-scale and parallel mode problems. Therefore, the NN model has been widely used in the modelling and optimization of aluminium electrolysis. Li et al. [18] illustrated the multi-fault diagnosis of aluminium electrolysis based on modular fuzzy NNs. To achieve low power consumption and high efficiency in aluminium electrolysis, an aluminium electrolysis model and optimization method based on a recurrent NN and preference information were proposed by Yi et al. [15]. Zhou et al. [19] used a general regression neural network to predict the anode effect of aluminium electrolysis.

However, the weights remain fixed after training. When the conditions of aluminium electrolysis change, the BPNN's weights cannot be adjusted by the new data unless another new BPNN model is retrained, but this is not a true "dynamical adjustment". The "dynamical adjustment" in this paper is to adjust or update the weights and thresholds dynamically using the new data coming in based on the established model, instead of retraining a new model as the traditional NN does. In addition, as the statistical characteristics of process noise may not satisfy the Gaussian distribution in practical aluminium electrolysis applications, so the prediction model established by BPNN or other methods may have a significant deviation from the real model of the manufacturing system. If the predictive model with a large deviation is used to carry out process optimization design, it will seriously affect the optimization effect of the manufacturing system.

Inspired by particle filtering theory [20,21], the conditional distribution of BPNN's discrete weights can be obtained approximately by parameter probability estimation. Moreover, according to the real-time measurement data obtained by sensors and other devices, the probability distribution is constantly adjusted to update the model parameters of the NN in real-time. Finally, the resampling technique is introduced to avoid the model search from becoming trapped in suboptimal positions while using particle filtering theory to overcome the problem of non-Gaussian process noise statistical characteristics, so that the modelling process can approximate the optimal estimation.

Based on the above analysis, the main contributions of this paper are as follows:

(1) To accurately establish the mathematical model for solving non-linear and non-Gaussian problems, we adopt the fusion of hybrid annealed particle filter (HAPF) theory and a BPNN algorithm. The BPNN's weights and thresholds are used as the HAPF state variables, and the BPNN's outputs are used as the HAPF measurement variables. Therefore, this paper proposes a hybrid annealed particle filter neural network (HAPFNN) nonlinear non-Gaussian modelling algorithm to solve the above problem.

(2) To reduce the negative impact from the loss of particle diversity on modelling performance, this paper presents an HAPFNN that uses a multi-sampling technique to adjust the probability distribution of particle sets to modify the data sampling range.

(3) Considering the problem of increased data storage caused by the introduction of hybrid proposal distribution, this paper uses an adaptive inheritance method to improve the processing ability of real-time data information inherited by the algorithm.

(4) Based on the above research findings, this paper systematically proposes a novel multi-sampling inherited hybrid annealed particle filter neural network (MSI-HAPFNN) algorithm and provides a detailed algorithm design process.

(5) The above-improved algorithm is applied to the modelling problem of actual industrial manufacturing systems, namely, AEMS modelling. The experimental results show that the MSI-HAPFNN algorithm can significantly improve the predictive capacity of the model.

The organization of the paper is as follows: Section 2 gives a brief description of the problems encountered in the process of aluminium electrolysis modelling. Section 3 introduces HAPF

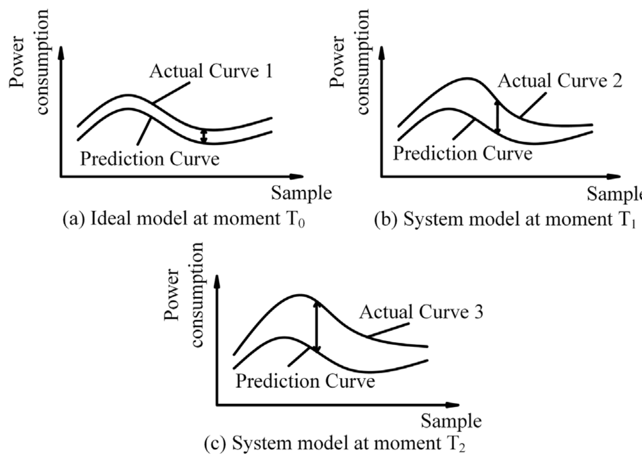
Power consumption
Actual Curve 1
Prediction Curve
Sample
(a) Ideal model at moment T$_0$
Actual Curve 2
(b) System model at moment T$_1$
Actual Curve 3
(c) System model at moment T$_2$

Fig. 1. The dynamic evolutionary model of the system.

theory and proposes a theoretical construction framework and experimental procedure for the HAPFNN algorithm in detail. Section 4 introduces the multi-sampling and inheritance concepts and presents the MSI-HAPFNN algorithm. In Section 5, the algorithm is applied and verified in AEMS modelling. Section 6 provides a summary.

2. Problem description

In process industry manufacturing systems [22], it is usually required to maintain fast response performance, accurate prediction performance and excellent robustness. Although the traditional NN model can be used to approximate the performance index of the real system, the development potential of the manufacturing system model is still very large in the face of the such complex conditions.

We define a nonlinear system:

$$\begin{cases} x_{k+1} = f(x_k, u_k) + \theta_k \\ \quad y_k = h(x_k, u_k) + \nu_k \end{cases} \tag{1}$$

where x_k signifies the system state vector at time k, u_k represents the input variables of the industrial system and y_k signifies the system measurement vector at time k. θ_k and ν_k correspond to the independent process noise and measurement noise (not necessarily Gaussian distribution noise), respectively. The functions f and h describe the functional relationship of state variables and measured values with time.

It is assumed that Fig. 1 shows a system model established based on the above equation and the possible state of the model, in which the prediction curve is stable in the coordinate system above. The actual curve evolves with the operation of the process manufacturing system. Fig. 1(a), (b) and (c) show that the degree of fitting between the actual curve and the prediction curve at three point (T_0, T_1 and T_2) of the system worsens with time. It is shown that the established model does not have the ability of adaptive adjustment, and cannot gradually approach the real system with time. Therefore, it is particularly important to establish a dynamic evolutionary model that can solve process system problems.

To establish the model described above, the previous literature proposed an extended Kalman filter neural network [23–25] (EKFNN) and an unscented Kalman filter neural network [26–30] (UKFNN) based on Kalman filtering theory and a NN algorithm. The above two methods essentially make use of the extended Kalman filter (EKF) and unscented Kalman filter (UKF) to dynamically adjust NN's weights and thresholds and establish a dynamic evolutionary model that changes with the production conditions in real-time to realize the optimal design of the process conditions in the AEMS.

However, the traditional Kalman filter requires that the process noise θ_k and measurement noise ν_k of the system are Gaussian white noise and are independent of each other. Otherwise, Kalman filter may diverge. Particle filtering theory has the potential to solve the above problems. Literature [31] used particle filter to adaptively train neural networks, in which the covariance matrix of the noise is set to evolve over time; Literature [32] employed particle filter and radial basis functions neural networks to optimize the posterior probability distribution of weights, which was used to predict the life of lithium-ion batteries.

Therefore, inspired by the existing literature, to give full play to the dual advantages of PFs and NNs, this paper designs an HAPFNN, which can address non-linear non-Gaussian problems by exploring the fusion strategy of PFs and NNs. Under the premise of fully considering the dynamic characteristics in an AEMS, the multi-sampling inheritance theory is introduced, and then the MSI-HAPFNN is systematically proposed. The detailed design process of the theory will be introduced in turn.

3. Design and analysis of HAPFNN algorithm

3.1. Hybrid annealed particle filter

The hybrid annealed particle filter (HAPF) [31,32] is one of the typical representatives of particle filtering theory. The hybrid proposal distribution (HPD) in this filtering algorithm replaces the posterior proposal distribution (PPD) in traditional particle filter as an importance function. Compared with the PPD, the HPD has the advantages of easy weight update and simple calculation. In addition, the measurement information obtained recently is not lost, so its importance weight has a smaller variance.

As can be seen from Table 1, HAPF still uses the prior distribution to calculate the importance distribution. Therefore, there still exists the problem that the proposed distribution has a large deviation from the likelihood distribution. For this reason, according to the relationship between the statistical characteristics of state noise and measurement noise, an annealing factor β is introduced to solve this problem.

Under real conditions, it is often difficult to predict the statistical characteristics of noise in nonlinear non-Gaussian aluminium electrolysis. The HAPF theory achieves the optimal estimation [33,34] of non-Gaussian problems based on the Monte Carlo method and recursive Bayesian estimation. There are two reasons why this theory can solve non-Gaussian problems: (1) The sample is extracted from the state parameter containing non-Gaussian process noise instead of performing numerical operations; (2) The process is adjusted to indirectly correct the weights of the measured data with non-Gaussian noise, rather than directly.

3.2. Design of the HAPFNN algorithm

The HAPFNN utilizes HAPF to probabilistically estimate the BPNN's weights and thresholds. The BPNN's weights and thresholds are used as the state variables of HAPF, and the NN's output is used as the measurement variable of HAPF.

Firstly, the initial weights and thresholds can be obtained by BPNN training. Secondly, the acquired weights and thresholds are introduced into HAPF for updating and optimization. Finally, the HAPFNN algorithm construction framework is established. Compared with the EKFNN and UKFNN, the HAPFNN updates the sample set by probability distribution estimation instead of a large number of complex recursive matrix operations. Compared with the traditional particle filter neural network (PFNN),

Table 1
Comparison of particle filter and hybrid annealing particle filter.

Related distribution	Particle filter	Hybrid annealing particle filter
Likelihood distribution	$p(y_k\|x_k)$	$p(y_k\|x_k)$
Prior distribution	$p(x_k\|x_{k-1})$	$p(x_{2,k}\|x_{2,k-1})^{\beta}$
Importance distribution	$q(x_k\|x_{k-1},y_k)=p(x_k\|x_{k-1})$	$q(x_k\|x_{k-1},y_k)=p(x_{2,k}\|x_{2,k-1})^{\beta}\times p(x_{1,k}\|x_{2,k},x_{k-1},y_k)$
Weight recursive expression	$\omega_k=\omega_{k-1}\ p(y_k\|x_k)$	$\omega_k=\omega_{k-1}\ p(x_{2,k}\|x_{2,k-1})^{1-\beta}\times p(y_k\|x_{2,k},x_{k-1},y_{k-1})$

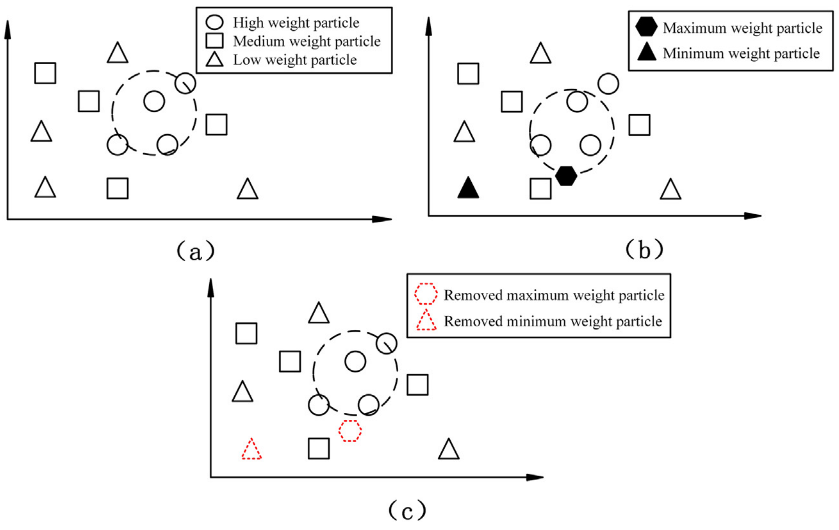

Fig. 2. Comparison between single sampling and multiple sampling.

the HAPFNN replaces the PPD with the HPD and introduces the annealing factor for further optimization.

In terms of algorithm's convergence, the algorithm uses neural network as the main model, and uses hybrid annealed particle filter to optimize weights and thresholds. Therefore, the convergence performance of the neural network has not been changed [35]. Meanwhile, the convergence performance of the hybrid annealed particle filter is consistent with that of the particle filter [36,37]. Appendix analyses the robustness of HAPFNN algorithm.

4. Design and analysis of the MSI-HAPFNN algorithm

In this paper, HAPF theory [38,39] is incorporated into the BPNN algorithm, and the HAPFNN algorithm, which can dynamically address non-linear non-Gaussian problems, is proposed. In the model updating process, since the particles with larger weights are selected multiple times, the sampling results often contain many repetition points, which leads to the abnormal sampling direction/range; And the hybrid proposal distribution (HPD) replaces the posterior proposal distribution (PPD) as an importance function, which increases the amount of data storage. Therefore, to further improve the stability of the algorithm, the theory of multi-sampling and inheritance are introduced to overcome these disadvantages, and a dynamic MSI-HAPFNN algorithm is systematically presented.

4.1. Multi-sampling

Although the HAPFNN algorithm uses the resampling method or improved resampling method [40,41] to solve the problem of particle scarcity, it also reduces the overall variance of the particles, resulting in the loss of particle diversity. Due to the loss of particle diversity, the sampling direction/range is changed, which seriously affects the operation accuracy. Multi-sampling is used to solve this problem. Multi-sampling combines general resampling with linear resampling based on the idea of combinatorial optimization. This sampling method arranges all the particles in order of weights from small to large. Particles with smaller weights are not directly excluded; rather, they are combined with the particles with larger weights according to a certain linear proportion (or nonlinear proportion) to generate new particles to avoid exclusion, and the larger particle weights become half of the original weights. This approach not only eliminates the dominant position of particles with maximum weight due to abnormal sampling but also prevents reduction in particle diversity. Therefore, this paper utilizes the multi-sampling method to reduce or even eliminate the above effects. In other words, general resampling and linear optimal resampling are integrated into the algorithm by combining two or more sampling techniques.

As shown in Fig. 2, particles with different weights in Fig. 2(a) are represented by different graphs and are randomly distributed in state space. Due to the use of resampling technology, abnormal particles with maximum weights appear in Fig. 2(b), which offset the sampling range (represented by the dotted circle). If single sampling technology is used, the negative effect of this phenomenon cannot be avoided, causing some particles to be difficult to sample, thus affecting the accuracy of weight updating. Therefore, through the introduction of multi-sampling, a pair of particles with maximum and minimum weights is discarded (the polygons and triangles represented by dotted lines in Fig. 2(c) will disappear) so that the sampling range returns to normal.

In the HAPFNN algorithm, two weights are introduced as a "bridge" between the sampling method and the sampling data, namely, the NN weight and the PF weight. The NN weight is used as the state variable of the PF, and then the corresponding proposal distribution is obtained. Moreover, the sample set is updated according to the resampling method. The PF weight is used to determine whether to perform linear optimization resampling. Each sample corresponds to a weight, and these samples refer to the above NN weight. It is shown that each PF weight corresponds to an NN weight, which is equivalent to a kind of "marked price" for NN weight. The essential information, however, is still the data

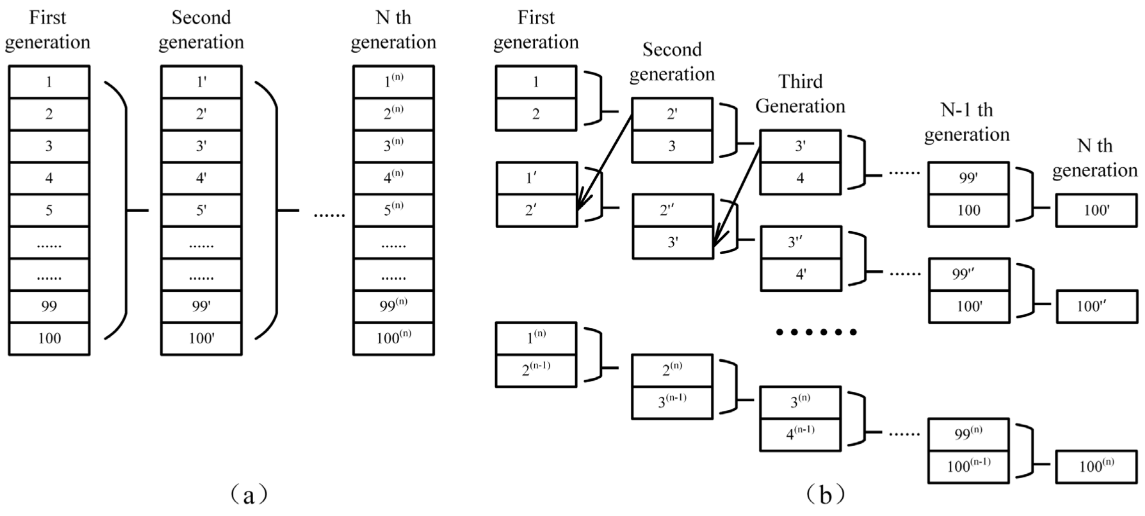

Fig. 3. Comparison between batch inheritance and adaptive inheritance.

behind the weight. Next, the normalized weight ω_k^i is calculated; if $N_{eff} < N_{th}$, then the linear optimization resampling is carried out; otherwise, it is not. (Here, N_{eff} is the number of valid samples, and N_{th} is the threshold of the number of samples.)

4.2. Inheritance

Inheritance is prominently reflected in the weight update process. Due to the introduction of the hybrid proposal distribution, the amount of calculation storage is increased, which seriously affects the operation speed.

Usually, the inheritance methods can be divided into two categories, batch inheritance and adaptive inheritance. In batch inheritance, the whole data block is accepted and inherited at one time. As the amount of calculation increases, the data memory capacitance increases in the inheritance process; simultaneously, the previous data take up space and waste resources. Adaptive inheritance carries out the procedure at the same time as the data are received. A new set of data is received each time, and the HPD of the model is updated, which is suitable for real-time online processing. When the new data are integrated into the model, the previous data do not need to occupy space, so the adaptive inheritance of the dynamic model data can be achieved well.

As shown in Fig. 3, batch inheritance inherits 100 data in the first generation at one time, while adaptive inheritance updates pairwise data to inherit the data obtained last time, which not only realizes synchronous inheritance but also frees up space, which is beneficial to reduce the calculation storage amount at the latest time point.

4.3. Design and analysis of the MSI-HAPFNN algorithm

The MSI-HAPFNN algorithm is optimized on the basis of the traditional BPNN, mainly including the following:

(1) The traditional BPNN is a static model, while MSI-HAPFNN is a dynamic model. The latter can constantly adjust itself, and it can actively approach the best model when it receives new data.

(2) To reduce the impact of the loss of particle diversity, two kinds of weights (PF weights and NN weights) are introduced. The NN weight is used as the state variable to calculate the corresponding proposal distribution in the HAPF algorithm. The PF weight is used to determine whether to perform linear optimization resampling to discard particles with higher weights and maintain the range of resampling.

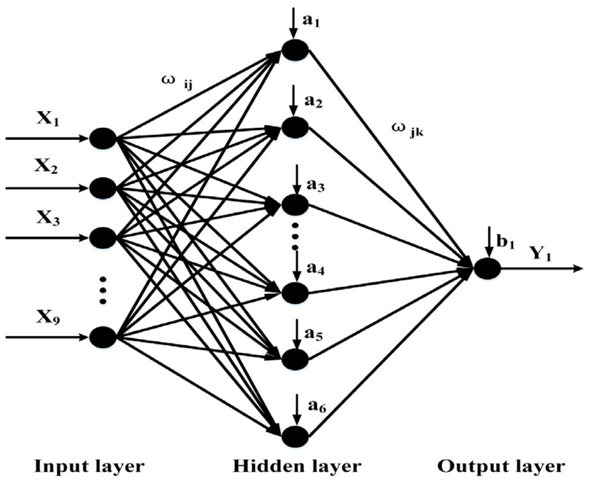

Fig. 4. BPNN's topology diagram.

(3) In the process of utilizing the HAPF algorithm to approximately replace the PPD with the HPD, the adaptive inheritance is used to reduce the amount of data storage;

In this paper, based on the BPNN algorithm, the MSI-HAPFNN algorithm is obtained by combining HAPF, kernel density estimation, linear optimal resampling and so on. The MSI-HAPFNN algorithm is as follows:

Step 1: Initialize the BPNN's weights and thresholds, and set the number of BPNN's input layer i, hidden layer j, output layer m and determine the hidden layer and output layer activation function. (The structure of the neural network is shown in Fig. 4.)

Step 2: Acquire data, and obtain the best training set and test set through cross-validation (CV).

In Fig. 4, we establish a BPNN model with an input layer number of 9, a hidden layer number of 25, and an output layer number of 1. $X_1, X_2 \cdots, X_9$ is the input value of the BPNN, and Y_1 is the output value of the BPNN, where $\omega_{i,j}$ represents the connection weight of the ith input layer to the jth hidden layer; $\omega_{j,m}$ represents the connection weight of the ith hidden layer to the jth output layer; a represents the threshold of hidden layer; and b represents the threshold of output layer.

Step 3: Run the BPNN to obtain and update the weights and thresholds according to Eq. (2).

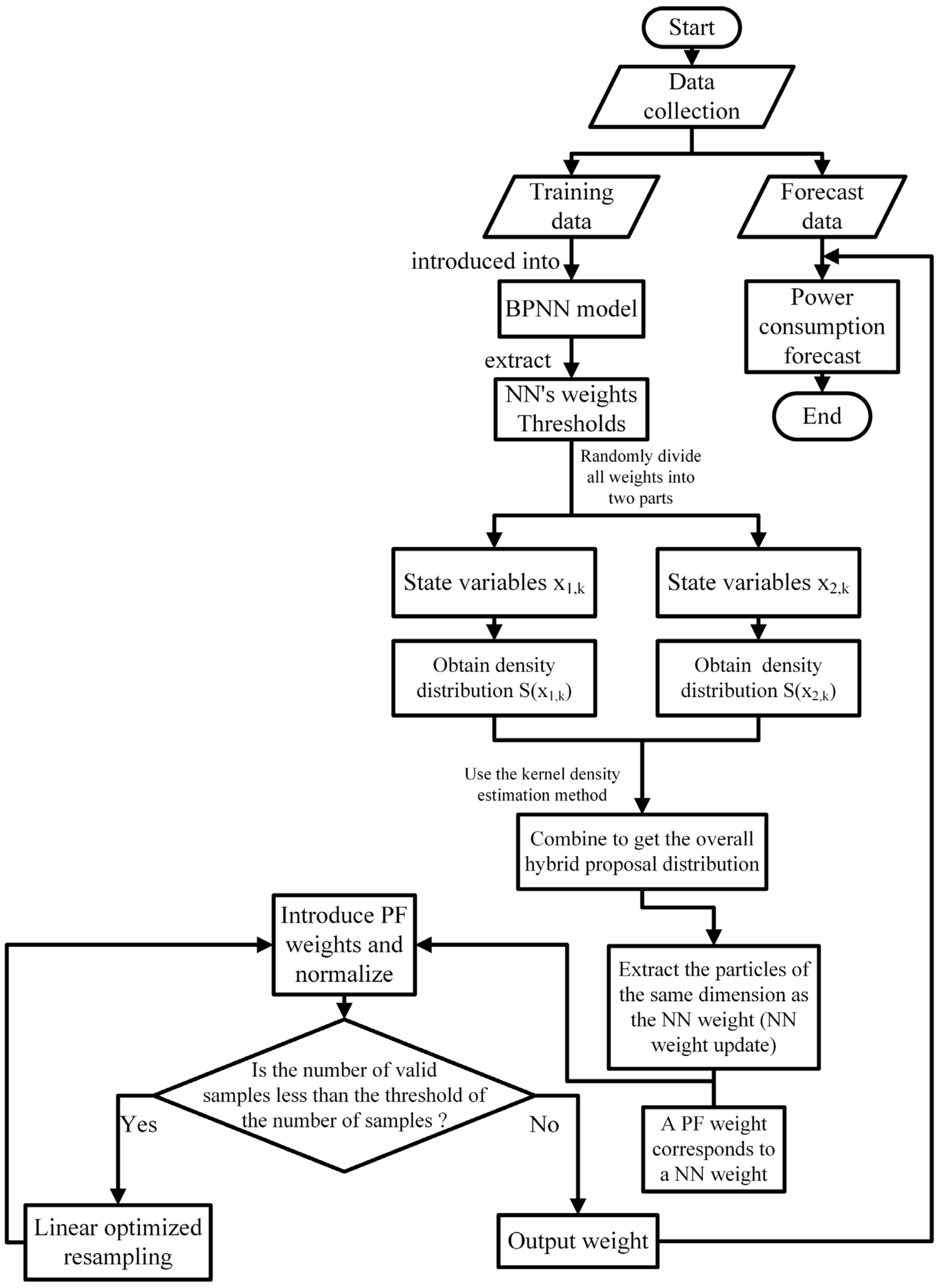
Start
Data collection
Training data
Forecast data
introduced into
BPNN model
extract
NN's weights Thresholds
Randomly divide all weights into two parts
State variables $x_{1,k}$
State variables $x_{2,k}$
Obtain density distribution $S(x_{1,k})$
Obtain density distribution $S(x_{2,k})$
Use the kernel density estimation method
Combine to get the overall hybrid proposal distribution
Extract the particles of the same dimension as the NN weight (NN weight update)
A PF weight corresponds to a NN weight
Introduce PF weights and normalize
Is the number of valid samples less than the threshold of the number of samples ?
Yes
No
Linear optimized resampling
Output weight
Power consumption forecast
End

Fig. 5. Flowchart of the MSI-HAPFNN algorithm.

Weights and thresholds update:

$$\left.\begin{aligned}
\omega_{ij}^{(k+1)} &= \omega_{ij}^{(k)} + \eta H_j(1-H_j)x(i)\sum_{m=1}^{l}\omega_{jm}e_m \\
\omega_{jm}^{(k+1)} &= \omega_{ij}^{(k)} + \eta H_j e_m \\
a_j^{(k+1)} &= a_j^{(k)} + \eta H_j(1-H_j)x(i)\sum_{m=1}^{l}\omega_{jm}e_m \\
b_m^{(k+1)} &= b_m^{(k)} + e_m
\end{aligned}\right\} \tag{2}$$

where H_j represents the output of the jth hidden layer, η represents the learning rate, and e represents the error between the predicted output and the desired output. The superscript k indicates the time.

Step 4: The weights and thresholds obtained by step 3 are ultimately used as the initial value of HAPF.

State space representation of neural network (Filtering equation):

$$\begin{cases} \omega_k = \omega_{k-1} + \theta_k \\ y_k = h(\omega_k, u_k) + \nu_k \end{cases} \tag{3}$$

Step 5: Determine the state variables (particles) ω_k and measurement variables y_k of the HAPF model. The particles are divided into two parts $\omega_{1,k}$ and $\omega_{2,k}$, and the appropriate state variables and measurement variables are determined by the CV method.

Step 6: Determine the annealing factor β based on the relationship between the statistical characteristics of the state noise and the measurement noise. Use the kernel density estimation method [42,43] to get the HPD.

$$q(\omega_k|\omega_{k-1}, y_k) = p(\omega_{1,k}|\omega_{2,k}, \omega_{k-1}, y_k) \times p(\omega_{2,k}|\omega_{2,k-1})^{\beta} \tag{4}$$

Step 7: Resampling. Randomly select the same dimensional particle from the HPD, update the weights w_k according to Eq. (5), and then calculate the normalized weights w_k according to Eq. (6).

$$w_k = w_{k-1}\ p(y_k|\omega_{2,k}, \omega_{k-1}, y_{k-1}) \times p(\omega_{2,k}|\omega_{2,k-1})^{1-\beta} \tag{5}$$

$$\tilde{w_k}^{(i)} = w_k^{(i)} / \sum_{i=1}^{N} w_k^{(i)} \tag{6}$$

Step 8: Calculate the number of valid samples according to Eq. (7).

$$N_{eff} = 1/\sum_{i=1}^{N} (w_k^{(i)})^2 \tag{7}$$

Step 9: Set the sample number threshold N_{th}, and compare the sizes of N_{eff} and N_{th}.

Step 10: If $N_{eff} < N_{th}$, skip to Step 11; otherwise, skip to Step 12.

Step 11: Adopt linear optimized resampling [44] according to Eq. (8).

$$\omega_n = \omega_\alpha + L(\omega_\alpha - \omega_s) \tag{8}$$

where ω_n is a new sampling point generated by combination; ω_α is a repeatedly selected sampling point; ω_s is a discarded sampling point; and L is a suitable step size of $(\omega_\alpha - \omega_s)$.

Step 12: Calculate the posterior mean estimation of the state; that is, obtain the weight and threshold of the algorithm.

Step 13: Re-import the output weights into BPNN for performance testing and achieve its dynamic balance.

To more clearly express how the data are updated and how the particles undergo multi-sampling, Fig. 5 shows the flow chart for the MSI-HAPFNN algorithm.

4.4. Description of genetic algorithm (GA) optimization

To take full advantage of the prediction potential of the HAPFNN algorithm, the best prediction model is required. The GA [45–47], which simulates the natural genetic mechanism and biological evolutionary theory, is applied to the parallel search optimization of the model.

The optimization process is shown in Fig. 6. The weights and thresholds are obtained in the MSI-HAPFNN algorithm, and we utilize the real coding method to perform the initial value encoding of GA optimization. Then, the absolute value of the error between the predicted output and the expected output is taken as the fitness function F. By using the roulette method, the real number uniform intersection method and the multi-point variation method, the data are ultimately updated and eliminated to make the prediction results more accurate.

$$F = k(\sum_{i=1}^{n} abs(y_i - o_i)) \tag{9}$$

Here, n is the number of output nodes in the algorithm, y_i is the expected output of the ith node in the algorithm, o_i is the predicted output of the ith node in the algorithm, and k is the coefficient.

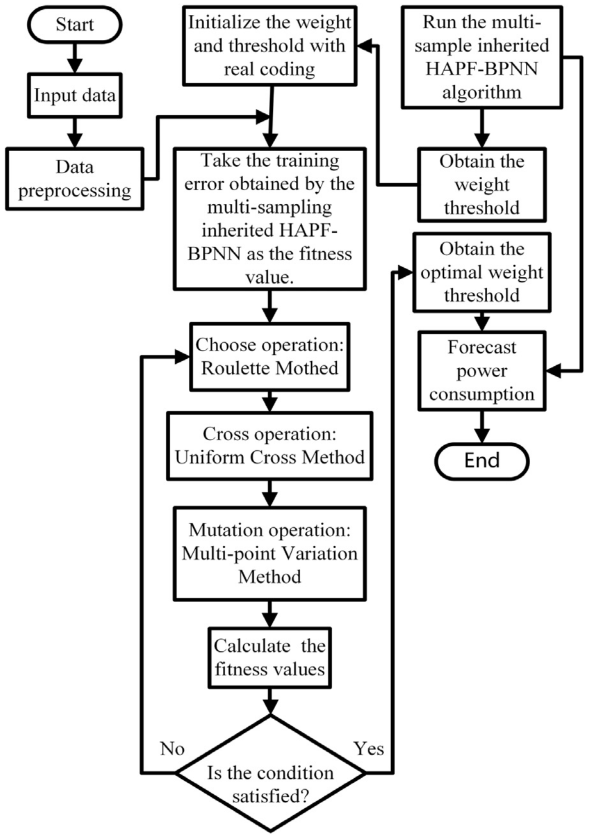
Start
Input data
Data preprocessing
Initialize the weight and threshold with real coding
Take the training error obtained by the multi-sampling inherited HAPF-BPNN as the fitness value.
Choose operation: Roulette Mothed
Cross operation: Uniform Cross Method
Mutation operation: Multi-point Variation Method
Calculate the fitness values
Is the condition satisfied?
No
Yes
Run the multi-sample inherited HAPF-BPNN algorithm
Obtain the weight threshold
Obtain the optimal weight threshold
Forecast power consumption
End

Fig. 6. Flowchart of MSI-HAPFNN optimization by a GA.

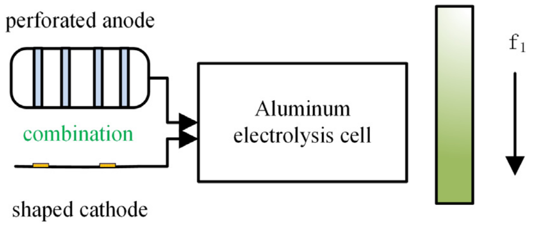

Fig. 7. The schematic diagram of the core components in aluminium electrolysis cell.

5. Application experiment of dynamic evolutionary modelling of process energy consumption in a novel aluminium electrolytic cell

5.1. Experimental objects

In this paper, the above algorithm is applied to an aluminium electrolytic cell based on special-shaped perforation and shaped cathode technology [48] as shown in Fig. 7 for industrial experiments. In the figure, $f1$ represents the energy consumption of aluminium electrolysis. Ideally, we wish to make the energy consumption as low as possible.

The aluminium electrolysis process has many features, such as nonlinear, multi-parameters, strong coupling, time-varying delay and high noise, and many operations, such as changing the anode, lifting the bus, shelling and aluminium discharge. Thus, the above process is very complex. It is difficult to accurately obtain a power consumption model for process optimization by using traditional modelling methods. The improved HAPFNN algorithm can avoid the impact of abnormal data and improve the processing ability

Table 2
The sample data of aluminium reduction cell from No. 158.

parameters	Sample				
	1	2	3	⋯	300
Series current (A)	1676	1676	1676	⋯	1682
Molecular ratio (1)	2.52	2.41	2.41	⋯	2.34
Aluminium level (cm)	24	24	25	⋯	23
Electrolyte level (cm)	17	18	19	⋯	14
Cell temperature (°C)	952	954	947	⋯	946
Aluminium output (kg)	1320	1260	1300	⋯	1320
Daily consumption of fluoride salt (kg)	23	23	22	⋯	20
Blanking interval (s)	153	153	153	⋯	148
Cell voltage (mV)	3812	3819	3808	⋯	3827
DC power consumption (kW.h/t-Al)	12 250	12 819	12 348	⋯	11 710

Table 3
The full names and abbreviations of different algorithms in the experiment.

Full names	Abbreviations
Back propagation neural network	BPNN
Hybrid annealed particle filter neural network	HAPFNN
Single-sampling inherited HAPFNN	SSI-HAPFNN
Multi-sampling non-inherited HAPFNN	MSNI-HAPFNN
Multi-sampling inherited HAPFNN	MSI-HAPFNN
Multi-sampling inherited HAPFNN after GA optimization	MSI-HAPFNN-GA

of real-time data information, which can effectively ensure the accuracy and reliability of the model.

Through the analysis of the influencing factors of the unit DC power consumption in aluminium electrolysis cells, combined with expert experience and considering the actual difficulty of on-site data acquisition, the effective decision parameters are selected, such as the series current, molecular ratio, aluminium level, electrolyte level, cell temperature, aluminium output, daily consumption of fluoride salt, blanking interval and cell voltage. The sample of No. 158 in the 170 kA series aluminium electrolysis cell of an aluminium plant was sampled, and the daily data of 300 groups of 9 kinds of decision parameters were obtained, as shown in Table 2.

5.2. *Dynamic evolutionary modelling and result analysis of the process power consumption based on MSI-HAPFNN*

5.2.1. *Dynamic evolutionary modelling of the process power consumption*

The 300 groups of aluminium electrolysis data were divided into 250 groups of training set samples and 50 groups of test set samples. MSI-HAPFNN as proposed in this paper is used to construct a 3-layer feedforward network, with inputs of 9 decision parameters and an output of the unit DC power consumption. The hidden layer transfer function is the Sigmoid function, and the output layer transfer function is the Purelin function. To maintain accuracy and operation speed, 25 hidden layer nodes are selected in this article.

The annealed operator β in the HAPF algorithm is set to 0.5; the crossover probability in the GA is set at 0.4; the mutation probability is 0.2; the population size is 60; the number of iterations is 100. Under the same experimental conditions, different algorithms are used to predict the process power consumption.

5.2.2. *Analysis and discussion of experimental results*

Through the design and implementation of the following models, the power consumption of the aluminium electrolysis process is compared with the experimental results. All experiments were performed using the same sample data and MATLAB R2014b (CPU: i7-9750H; RAM: 8.00 GB; GPU: GTX 1660 Ti) as the simulation platform.

Fig. 8 show the fitting effect diagrams of the test samples based on the dynamic evolutionary model of process power consumption established by the above 6 schemes.

Fig. 8(a) shows that the prediction output of comparison between the BPNN, HAPFNN and MSI-HAPFNN; the fitting effect of the latter is significantly better than that of the former. Therefore, it is feasible to introduce the HAPF model into the BPNN algorithm and transform the static model into a dynamic model. The application of the multi-sampling and adaptive inheritance techniques is beneficial to update the particle weights. Fig. 8(b) shows the prediction output of comparison between SSI-HAPFNN and MSI-HAPFNN. The main difference is whether linear optimization resampling is introduced into the algorithm. It is found that the multi-sampling method can maintain the diversity of the particles and improve the prediction accuracy of the algorithm. Fig. 8(c) shows the prediction output of comparison between MSI-HAPFNN and MSNI-HAPFNN, and the fitting effect of the former is significantly better than that of the latter. The main reason is that the previous state variables and measurement variables are dynamically updated and modified so that the HPD gradually approaches the PPD. Fig. 8(d) shows the prediction output of comparison between MSI-HAPFNN-GA. A GA is used to further optimize the weight and threshold to obtain the optimal individual to improve the prediction accuracy of the model.

According to the simulation experiments, the fitting effect of the dynamic model is far better than that of the BPNN static model, and the implementation of inheritance and multi-sampling improves the fitting effect of the algorithm as well. Overall, the predictions of the MSI-HAPFNN model are in good agreement with the actual characteristics of aluminium electrolysis cells.

Fig. 9 show the relative percentage errors predicted by the five models. The maximum relative error percentage of MSI-HAPFNN-GA is less than 0.25%, corresponding to a better performance index than other algorithms.

Table 4 compares the index data of multiple linear regression (MLR), multiple nonlinear regression (NLMR) [49], Gaussian processes (GP) [50] and 6 models in Table 3 to establish the process power consumption model and lists 6 types of evaluation

Table 4
The comparison of related performance indicators from different models.

Model category	Comparison of absolute error indicators in test sets						Statistical test	Algorithm complexity	
	Max (%)	Min (%)	Average (%)	SSE (10^6)	MSE (10^4)	*RMSE*	*p*-value	Time complexity	Space complexity
MLR	19.2667	1.1156	11.6739	36.7660	50.3750	709.7532	1.03×10^{-4}	$O(n^3)$	$O(1)$
NLMR	13.8420	0.9589	7.1247	21.3883	29.2991	541.2861	4.77×10^{-4}	$O(n^3)$	$O(1)$
GP	8.7259	0.7543	4.8736	16.0177	21.9421	468.4240	8.81×10^{-4}	$O(n^3)$	$O(n^2)$
BPNN	5.8472	0.2519	2.6953	6.1462	9.1861	361.1520	3.24×10^{-3}	$O(n^3)$	$O(1)$
HAPFNN	4.2187	0.3599	1.4858	3.8521	7.0093	251.7539	7.48×10^{-3}	$O(n^3)$	$O(n)$
SSI-HAPFNN	2.7186	0.2863	0.9722	1.6817	3.5834	188.5238	7.80×10^{-3}	$O(n^3)$	$O(n)$
MSNI-HAPFNN	3.4511	0.1882	1.1683	2.7254	4.2797	201.1136	2.28×10^{-2}	$O(n^3)$	$O(n)$
MSI-HAPFNN	0.8466	0.0781	0.2207	0.4793	0.7122	79.4639	3.97×10^{-2}	$O(n^3)$	$O(n)$
MSI-HAPFNN-GA	**0.2404**	**0.0127**	**0.0782**	**0.0589**	**0.0627**	**30.5723**	\	$O(n^3)$	O(n)

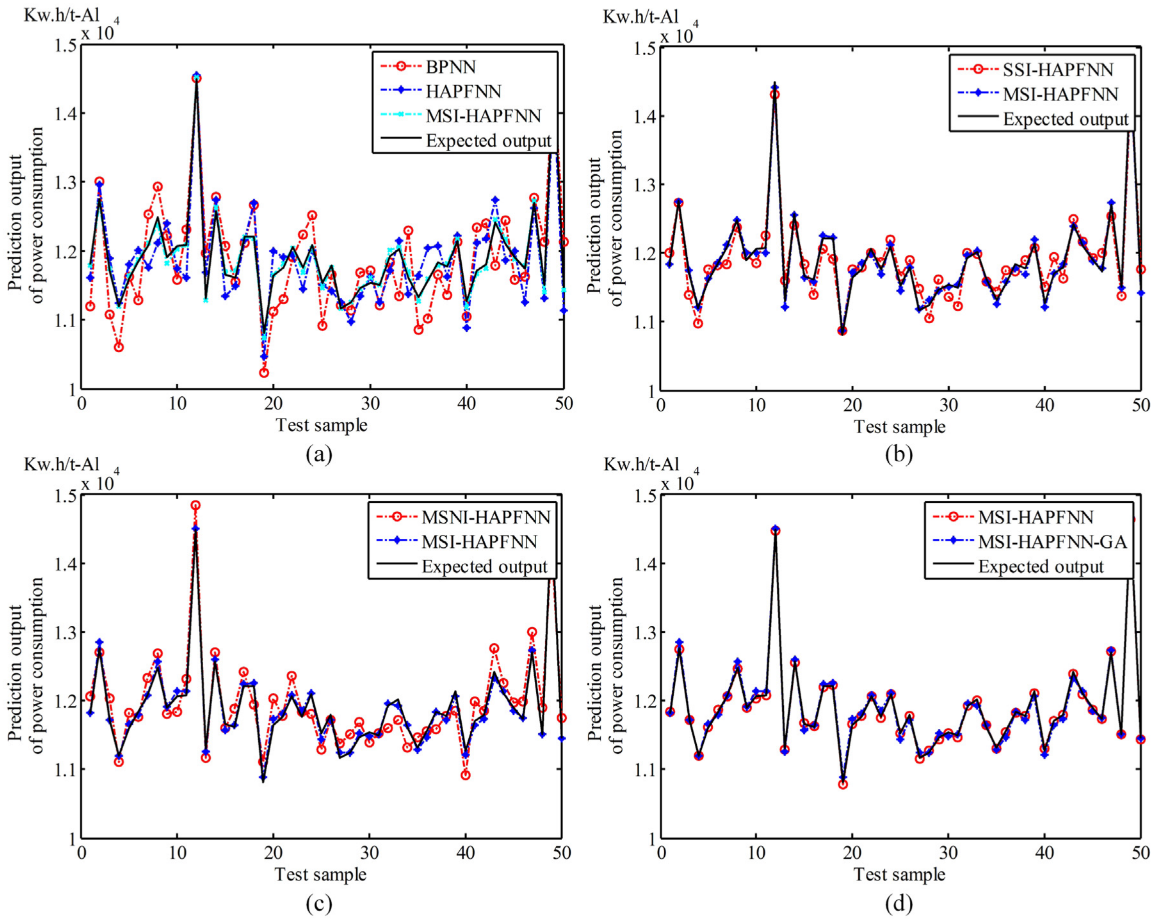

Fig. 8. Comparison of predicted output and expected output (a) BPNN, HAPFNN and MSI-HAPFNN (b) SSI-HAPFNN and MSI-HAPFNN (c) MSNI-HAPFNN and MSI-HAPFNN (d) MSI-HAPFNN and MSI-HAPFNN-GA.

Table 5
Statistics of related performance indexes for 20 tests base on 6 models in Table 3.

Number of tests	BPNN			HAPFNN			SSI-HAPFNN			MSNI-HAPFNN			MSI-HAPFNN			MSNI-HAPFNN-GA		
	MAE	MRE	R	MAE	MRE	R	MAE	MRE	R	MAE	MRE	R	MAE	MRE	R	MAE	MRE	R
1	6.1924	0.0333	0.7893	5.2352	0.0191	0.8870	2.0730	0.0111	0.9405	3.7677	0.0166	0.9235	1.1228	0.0032	0.9905	0.3628	0.0019	0.9986
2	6.7399	0.0339	0.7816	5.0658	0.0114	0.9037	2.0473	0.0084	0.9227	3.6454	0.0109	0.9144	1.1225	0.0026	0.9927	0.3509	0.0017	0.9987
3	6.6086	0.0278	0.8764	5.1270	0.0120	0.8975	2.6959	0.0139	0.9203	3.8448	0.0180	0.9140	1.0549	0.0045	0.9920	0.3131	0.0016	0.9993
4	6.0645	0.0259	0.8424	5.1215	0.0133	0.8879	2.5406	0.0122	0.9246	3.7765	0.0163	0.9249	1.0848	0.0023	0.9944	0.3404	0.0010	0.9987
5	6.3324	0.0339	0.7383	5.9071	0.0189	0.8938	2.0530	0.0130	0.9445	3.6046	0.0102	0.9234	1.0337	0.0034	0.9928	0.3022	0.0015	0.9998
6	6.8813	0.0292	0.7679	4.6448	0.0124	0.8961	2.6093	0.0123	0.9151	4.0693	0.0129	0.9003	1.1230	0.0032	0.9950	0.3981	0.0018	0.9986
7	7.2104	0.0253	0.7320	5.2611	0.0199	0.8856	2.5912	0.0135	0.9330	3.7146	0.0114	0.9112	1.0611	0.0035	0.9912	0.3027	0.0012	0.9994
8	6.0362	0.0283	0.7628	5.3170	0.0131	0.9006	2.3763	0.0149	0.9193	3.8893	0.0141	0.9000	1.0308	0.0041	0.9929	0.3326	0.0019	0.9996
9	6.9435	0.0266	0.7382	4.8445	0.0106	0.8878	2.4766	0.0103	0.9278	3.7657	0.0143	0.9277	1.0576	0.0025	0.9908	0.3967	0.0014	0.9996
10	6.5488	0.0297	0.7276	5.4678	0.0144	0.8899	2.4764	0.0104	0.9390	4.0829	0.0156	0.9194	1.0126	0.0037	0.9950	0.3024	0.0016	0.9994
11	6.0323	0.0278	0.7849	5.5599	0.0209	0.8822	2.2349	0.0100	0.9268	3.6059	0.0126	0.9206	1.0503	0.0044	0.9905	0.3696	0.0012	0.9998
12	6.2157	0.0327	0.8135	5.6336	0.0128	0.9028	2.2010	0.0147	0.9486	4.0339	0.0130	0.9281	1.1285	0.0031	0.9949	0.3717	0.0020	0.9996
13	6.5716	0.0328	0.8427	5.2913	0.0194	0.8824	2.1020	0.0081	0.9194	3.9911	0.0121	0.9063	1.0613	0.0045	0.9912	0.3724	0.0013	0.9998
14	6.7419	0.0298	0.7893	4.9059	0.0129	0.9076	2.5440	0.0082	0.9257	3.6330	0.0139	0.9145	1.0452	0.0027	0.9933	0.3787	0.0014	0.9995
15	7.1635	0.0263	0.8646	5.1606	0.0185	0.9065	2.0546	0.0122	0.9172	4.0189	0.0152	0.9015	1.0915	0.0024	0.9912	0.3539	0.0016	0.9986
16	6.6058	0.0299	0.8073	5.0154	0.0133	0.8894	2.4523	0.0091	0.9217	3.7988	0.0121	0.9088	1.1093	0.0022	0.9932	0.3417	0.0011	0.9998
17	6.5372	0.0326	0.7773	4.7613	0.0131	0.8883	2.9365	0.0127	0.9435	3.7659	0.0129	0.9098	1.0111	0.0040	0.9923	0.3599	0.0016	0.9989
18	6.6311	0.0272	0.8376	4.5131	0.0141	0.8926	2.2504	0.0105	0.9321	4.0485	0.0151	0.9183	1.0286	0.0021	0.9948	0.3203	0.0015	0.9993
19	6.0157	0.0308	0.7955	5.5664	0.0208	0.8846	2.5267	0.0108	0.9272	3.9635	0.0120	0.9231	1.0523	0.0033	0.9935	0.3059	0.0016	0.9985
20	6.0800	0.0261	0.7542	4.9108	0.0154	0.8845	2.4939	0.0147	0.9311	3.9054	0.0172	0.9261	1.1011	0.0034	0.9947	0.3658	0.0020	0.9995

indicators for different model performance, including Max, Min, Average, SSE, MSE and RMSE [51].

Max and Min respectively reflect the upper and lower bounds of prediction error percentage. The prediction error percentage of the BPNN algorithm is significantly higher than that of the HAPFNN algorithm. The maximum prediction error percentage of MSI-HAPFNN-GA is only 0.2404%, which is quite high in terms of prediction accuracy.

From the data SSE and MSE in the table, it can be seen that HAPF theory plays an advantageous role in optimizing the BPNN

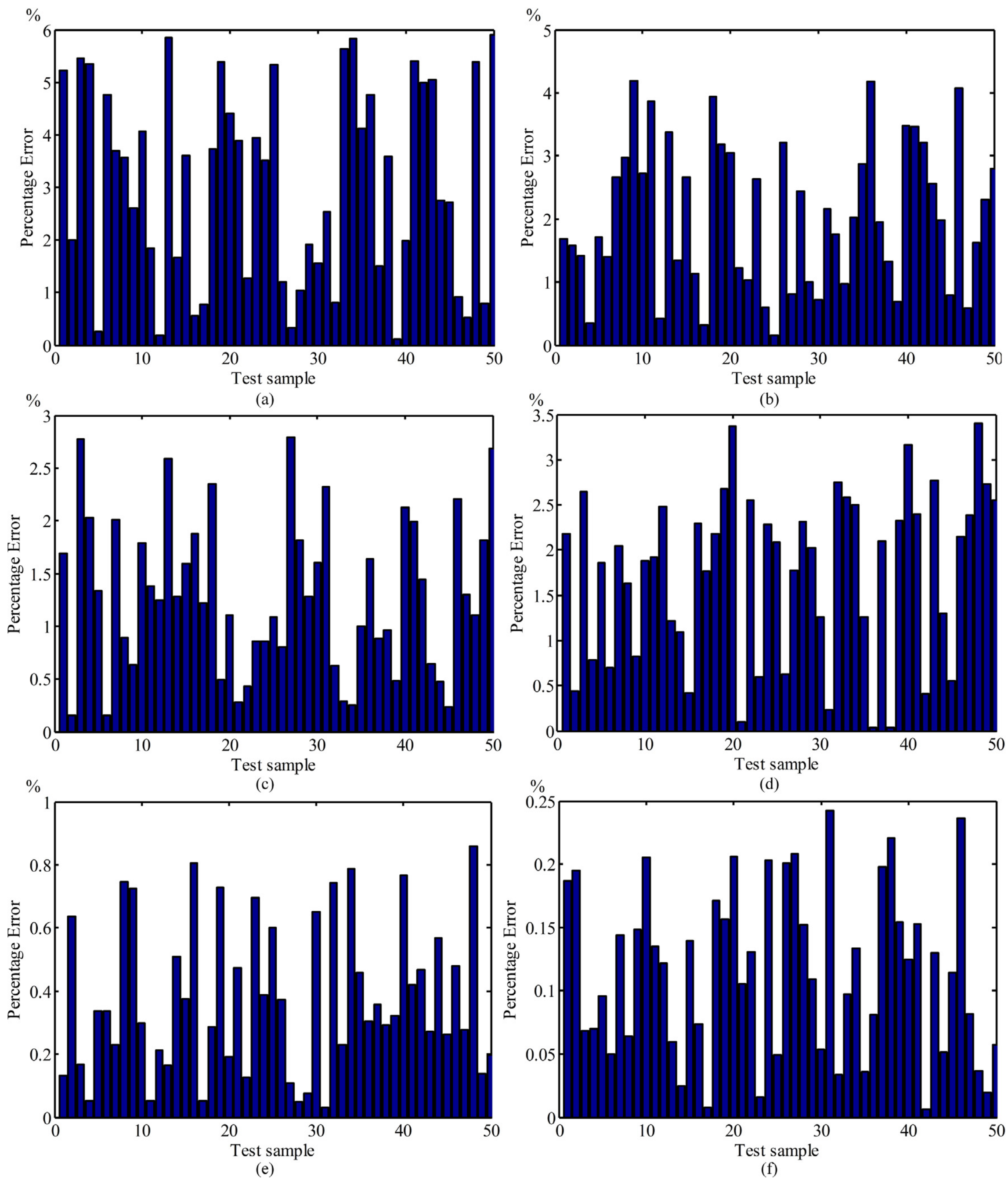

Fig. 9. The histogram of relative error percentage (a) BPNN (b) HAPFNN (c) SSI-HAPFNN (d) MSNI-HAPFNN (e) MSI-HAPFNN (f) MSI-HAPFNN-GA.

algorithm. The introduction of multi-sampling and inheritance further improves the performance of the model, and finally, the optimal weight is obtained by GA optimization.

The data in the table show that the absolute error indicators are relatively large because BPNN is a static model and lacks adjustment ability. The errors of SSI-HAPFNN and MSNI-HAPFNN are similar, but it can be concluded that inheritance has a greater impact on model prediction accuracy than multi-sampling. The indexes of the MSI-HAPFNN model are obviously improved. After further optimization by the GA, the average value of the absolute error is approximately 0.03%.

To further prove the effectiveness of the proposed method, Wilcoxon signed rank test is employed to perform a non-parametric test using the MSI-HAPFNN-GA as the reference objective. The corresponding results have been summarized in Table 4. Based on the presented results, the proposed method is significantly better than other approaches. In addition, the algorithm complexity has also been added in Table 4. It can be seen that the space complexity of the HAPFNN algorithm is better than BPNN. Even if HAPFNN is continuously optimized to obtain MSI-HAPFNN-GA, the complexity of its algorithm has not increased.

To avoid the accidental influence of external interference on the model, Table 5 show the statistics of different model-related performance indicators after 20 tests, including MAE, MRE and R (the correlation coefficient) [52].

Table 6 shows the statistical analysis of data from 20 tests in Table 5 to better contrast the advantages and disadvantages of the various models. The MSI-HAPFNN algorithm optimized by the GA is superior to the other algorithms.

Table 6
The comparison of performance indexes related to multiple tests base on different models.

Inspection standards	Model category	Average	Min	Max	Std	Median	Lower quantile	Upper quantile
MAE	BPNN	6.5076	6.0157	7.2104	0.3667	6.5602	6.1643	6.7404
	HAPFNN	5.1655	4.5131	5.9071	0.3436	5.1438	4.9096	5.3547
	SSI-HAPFNN	2.3868	2.0473	2.9365	0.2431	2.4644	2.1763	2.5415
	MSNI-HAPFNN	3.8463	3.6046	4.0829	0.1577	3.8218	3.7529	3.9981
	MSI-HAPFNN	1.0692	1.0111	1.1285	0.0375	1.0594	1.0423	1.1032
	MSI-HAPFNN-GA	**0.3471**	**0.3022**	**0.3981**	**0.0304**	**0.3524**	**0.3185**	**0.3701**
MRE	BPNN	0.0295	0.0253	0.0339	0.0028	0.0295	0.0271	0.0326
	HAPFNN	0.0153	0.0106	0.0209	0.0033	0.0138	0.0129	0.0190
	SSI-HAPFNN	0.0116	0.0081	0.0149	0.0021	0.0117	0.0102	0.0131
	MSNI-HAPFNN	0.0138	0.0102	0.0180	0.0021	0.0135	0.0121	0.0153
	MSI-HAPFNN	0.0033	0.0021	0.0045	0.0008	0.0033	0.0026	0.0038
	MSI-HAPFNN-GA	**0.0015**	**0.0010**	**0.0020**	**0.0003**	**0.0016**	**0.0014**	**0.0017**
R	BPNN	0.7912	0.7276	0.8764	0.0430	0.7871	0.7607	0.8195
	HAPFNN	0.8925	0.8822	0.9076	0.0079	0.8897	0.8867	0.8983
	SSI-HAPFNN	0.9290	0.9151	0.9486	0.0096	0.9270	0.9214	0.9345
	MSNI-HAPFNN	0.9158	0.9000	0.9281	0.0089	0.9164	0.9096	0.9234
	MSI-HAPFNN	0.9928	0.9905	0.9950	0.0016	0.9929	0.9912	0.9945
	MSI-HAPFNN-GA	**0.9993**	**0.9985**	**0.9998**	**0.0005**	**0.9994**	**0.9987**	**0.9996**

6. Conclusion

In this paper, a novel framework for adaptively forecasting the power consumption of aluminium electrolysis system is proposed. Fully taking the advantage of particle filter (PF) and neural networks, a hybrid annealed particle filter neural networks (HAPFNN) is adopted by the proposed framework to perform the prediction task. To ensure the framework to be robust to the outliers in the training data, a hybrid proposal distribution with an annealed operator, which makes the noise adjustable is used as the priors of HAPFNN and an effective multi-sampling technique is also incorporated into the updating of PF and back-propagation weights. The proposed model is then optimized by a genetic algorithm, which is capable of guiding the framework to search for the global optimum. The proposed framework has been compared with a number of state-of-the-art approaches with respect to the prediction of the power consumption of aluminium electrolysis cell. The experimental results show that the proposed framework achieves the best performance in terms of predictive accuracy and instantaneous track of slot-wise conditions. In future, we will attempt to further improve the framework robustness via allowing the Kalman/particle filters to deal with data coming from multiple sources and apply it to more complicated prediction tasks of power consumption of aluminium electrolysis systems.

CRediT authorship contribution statement

Wei Ding: Software, Validation, Writing - original draft, Writing - review & editing, Visualization. **Lizhong Yao:** Writing - original draft, Funding acquisition, Project administration, Writing - review & editing, Conceptualization, Methodology, Investigation. **Yanyan Li:** Conceptualization, Software, Supervision. **Wei Long:** Resources, Supervision. **Jun Yi:** Formal analysis, Visualization. **Tiantian He:** Formal analysis, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This study was supported by the National Natural Science Foundation of China (No. 51805059), Chongqing Research Program of Basic Research and Frontier Technology, China under grant (cstc2018jcyjAX0350), Achievement Transfer Program of Institutions of Higher Education in Chongqing, China (Grant No. KJZH17134) and in part by the China Scholarship Council under Grant 201802075004.

Appendix. Robustness analysis of HAPFNN algorithm

Taking Eq. (3) as the research object, inspired by the proof of the PF convergence [36,37], we further prove the robustness of the HAPFNN algorithm.

Definition 1. For function $\varphi(\omega_{0:k})$ on $R^{(k+1)n_\omega}$, if the following formula holds, then it is called $L_{k,p}$ space.

$$(q_{0:k|k}(d\omega_{0:k}), |\varphi(\omega_{0:k})|^p) < \infty, p \geq 1 \tag{A.1}$$

The module of $L_{k,p}$ space can be defined as

$$\|\varphi(\omega_{0:k})\|_{k,p} = [\int_{R^{(k+1)n_\omega}} |\varphi(\omega_{0:k})|^p q_{0:k|k}(d\omega_{0:k})]^{\frac{1}{p}} \tag{A.2}$$

where ω represents the weights and thresholds of HAPFNN algorithm.

Hypothesis 1. Assuming that the measurement sequence $y_{1:k}$ is known, and when $k > 0$, the following formula holds.

$$(q_{0:k|k-1}(d\omega_{0:k}), p(y_k|\omega_{0:k})) > 0 \tag{A.3}$$

Hypothesis 2. Assuming that when $k > 0$, the parameter γ_k in HAPFNN satisfies the following formula.

$$(q_{0:k|k-1}(d\omega_{0:k}), p(y_k|\omega_{0:k})) > \gamma_k > 0 \tag{A.4}$$

Hypothesis 3. When $k > 0$

$$\|p(y_k|\omega_k)K(\omega_k|\omega_{k-1})/G(\omega_k|\omega_{k-1}, y_{1:k})\| = \|\rho\| < \infty \tag{A.5}$$

$$\|p(y_k|\omega_k)\| = \|g\| < \infty \tag{A.6}$$

Lemma 1. *When Hypotheses 1 and 3 are satisfied, if $\varphi(\omega_{0:k}) \in L_{k,p}$, then*

$$(K(d\omega_{0:k}|\omega_{0:k-1}), g(y_k, \omega_{0:k})|\varphi(\omega_{0:k})|^p)^{1/p} \in L_{k-1,p} \tag{A.7}$$

Lemma 2. *If $\varphi(\omega_{0:k})$ is a bounded function, then $\varphi(\omega_{0:k}) \in L_{k,p}$.*

The following Lemmas are drawn under Hypotheses 1, 2, and 3.

Lemma 3. *When Hypotheses 1–3 are satisfied, there is a set of numbers $b^*_{0|0}(\varphi(\omega_0))$ and $m^*_{0|0}(\varphi(\omega_0))$ independent of N for any $\varphi(\omega_0) \in L_{k,4}$, so that the following two formulas are true.*

$$E[((q^{*N}_{0:0|0}(d\omega_0), \varphi(\omega_0)) - (q_{0:0|0}(d\omega_0), \varphi(\omega_0)))^4] \le \frac{b^*_{0|0}(\varphi(\omega_0))}{N^2} \tag{A.8}$$

$$E[(q^{*N}_{0:0|0}(d\omega_0), \varphi(\omega_0))^4] \le m^*_{0|0}(\varphi(\omega_0)) \tag{A.9}$$

Lemma 3 shows that the initialization phase of HAPFNN is convergent.

Lemma 4. *When Hypotheses 1–3 are satisfied, there is a set of numbers $b^*_{k-1|k-1}(\varphi(\omega_{0:k-1}))$ and $m^*_{k-1|k-1}(\varphi(\omega_{0:k-1}))$ independent of N for any $\varphi(\omega_{0:k-1}) \in L_{k-1,4}$, so that the following two formulas are true.*

$$E[((q^{*N}_{0:k-1|k-1}(d\omega_{0:k-1}), \varphi(\omega_{0:k-1})) - (q_{0:k-1|k-1}(d\omega_{0:k-1}), \varphi(\omega_{0:k-1})))^4] \le \frac{b^*_{k-1|k-1}(\varphi(\omega_{0:k-1}))}{N^2} \tag{A.10}$$

$$E[(q^{*N}_{0:k-1|k-1}(d\omega_{0:k-1}), \varphi(\omega_{0:k-1}))^4] \le m^*_{k-1|k-1}(\varphi(\omega_{0:k-1})) \tag{A.11}$$

Then there is a set of numbers $b_{k|k-1}(\varphi(\omega_{0:k}))$ and $m_{k|k-1}(\varphi(\omega_{0:k}))$ independent of N for any $\varphi(\omega_{0:k}) \in L_{k,4}$, so that the following two formulas are true.

$$E[((q^{N}_{0:k|k-1}(d\omega_{0:k}), \rho(\omega_{0:k}, y_{1:k})\varphi(\omega_{0:k})) - (q_{0:k|k-1}(d\omega_{0:k}), g(y_k, \omega_{0:k})\varphi(\omega_{0:k})))^4] \le \frac{b_{k|k-1}(\varphi(\omega_{0:k}))}{N^2} \tag{A.12}$$

$$E[(q^{N}_{0:k|k-1}(d\omega_{0:k}), \rho(\omega_{0:k}, y_{1:k})\varphi(\omega_{0:k}))^4] \le m_{k|k-1}(\varphi(\omega_{0:k})) \tag{A.13}$$

Lemma 5. *When Hypotheses 1–3 are satisfied, there is a set of numbers $b_{k|k-1}(\varphi(\omega_{0:k}))$ and $m_{k|k-1}(\varphi(\omega_{0:k}))$ independent of N for any $\varphi(\omega_{0:k}) \in L_{k,4}$, so that Eqs. (A.12) and (A.13) are true. Then there is a set of numbers for any $\varphi(\omega_{0:k}) \in L_{k,4}$. The numbers $b^{**}_{k|k-1}(\varphi(\omega_{0:k}))$ and $m^{**}_{k|k-1}(\varphi(\omega_{0:k}))$ independent of N make the following two formulas true.*

$$E[((q^{**N}_{0:k|k-1}(d\omega_{0:k}), \rho(\omega_{0:k}, y_{1:k})\varphi(\omega_{0:k})) - (q_{0:k|k-1}(d\omega_{0:k}), g(y_k, \omega_{0:k})\varphi(\omega_{0:k})))^4] \le \frac{b^{**}_{k|k-1}(\varphi(\omega_{0:k}))}{N^2} \tag{A.14}$$

$$E[(q^{**N}_{0:k|k-1}(d\omega_{0:k}), \rho(\omega_{0:k}, y_{1:k})\varphi(\omega_{0:k}))^4] \le m^{**}_{k|k-1}(\varphi(\omega_{0:k})) \tag{A.15}$$

Lemmas 4–5 indicate that HAPFNN is convergent in the prediction phase from moment k-1 to moment k.

Lemma 6. *When Hypotheses 1, 2, and 3 are satisfied, there is a set of numbers $b^{**}_{k|k-1}(\varphi(\omega_{0:k}))$ and $m^{**}_{k|k-1}(\varphi(\omega_{0:k}))$ that are independent of N for any $\varphi(\omega_{0:k}) \in L_{k,4}$, so that Eqs. (A.14) and (A.15) are true. Then there is a number d_k that is independent of N, so that the probability satisfies the following formula.*

$$P[\frac{1}{N}\sum_{i=1}^{N} \rho(\omega^{(i)}_{0:k|k-1}, y_{1:k}) < \gamma_k] \le \frac{d_k}{N^2} \tag{A.16}$$

And when $N \ge [d_k^{0.5}] + 1$,

$$P[\frac{1}{N}\sum_{i=1}^{N} \rho(\omega^{(i)}_{0:k|k-1}, y_{1:k}) < \gamma_k] \le \frac{d_k}{|[d_k^{0.5}] + 1|^2} < 1 \tag{A.17}$$

Lemma 7. *When Hypotheses 1–3 are satisfied, there is a set of numbers $b^{**}_{k|k-1}(\varphi(\omega_{0:k}))$ and $m^{**}_{k|k-1}(\varphi(\omega_{0:k}))$ independent of N for any $\varphi(\omega_{0:k}) \in L_{k,4}$, so that Eqs. (A.14) and (A.15) are true, and when $N \ge [d_k^{0.5}] + 1$, there is a set of numbers for any $\varphi(\omega_{0:k}) \in L_{k,4}$. The numbers $b^{*}_{k|k-1}(\varphi(\omega_{0:k}))$ and $m^{*}_{k|k-1}(\varphi(\omega_{0:k}))$ independent of N make the following two formulas true.*

$$E[((q^{*N}_{0:k|k-1}(d\omega_{0:k}), \rho(\omega_{0:k}, y_{1:k})\varphi(\omega_{0:k})) - (q_{0:k|k-1}(d\omega_{0:k}), g(y_k, \omega_{0:k})\varphi(\omega_{0:k})))^4] \le \frac{b^{*}_{k|k-1}(\varphi(\omega_{0:k}))}{N^2} \tag{A.18}$$

$$E[(q^{*N}_{0:k|k-1}(d\omega_{0:k}), \rho(\omega_{0:k}, y_{1:k})\varphi(\omega_{0:k}))^4] \le m^{*}_{k|k-1}(\varphi(\omega_{0:k})) \tag{A.19}$$

Lemma 8. *When Hypotheses 1–3 are satisfied, there is a set of numbers $b^{*}_{k|k-1}(\varphi(\omega_{0:k}))$ and $m^{*}_{k|k-1}(\varphi(\omega_{0:k}))$ independent of N for any $\varphi(\omega_{0:k}) \in L_{k,4}$, so that Eqs. (A.18) and (A.19) are true. Then, there is a set of numbers for any $\varphi(\omega_{0:k}) \in L_{k,4}$. The numbers $b^{*}_{k|k}(\varphi(\omega_{0:k}))$ and $m^{*}_{k|k}(\varphi(\omega_{0:k}))$ independent of N make the following two formulas true.*

$$E[((q^{*N}_{0:k|k}(d\omega_{0:k}), \varphi(\omega_{0:k})) - (q_{0:k|k}(d\omega_{0:k}), \varphi(\omega_{0:k})))^4] \le \frac{b^{*}_{k|k}(\varphi(\omega_{0:k}))}{N^2} \tag{A.20}$$

$$E[(q^{*N}_{0:k|k}(d\omega_{0:k}), \varphi(\omega_{0:k}))^4] \le m^{*}_{k|k}(\varphi(\omega_{0:k})) \tag{A.21}$$

Lemmas 7–8 indicate that HAPFNN is still convergent after resampling.

Lemma 9. *When Hypotheses 1–3 are satisfied, there is a set of numbers $b^{*}_{k|k}(\varphi(\omega_{0:k}))$ and $m^{*}_{k|k}(\varphi(\omega_{0:k}))$ independent of N for any $\varphi(\omega_{0:k}) \in L_{k,4}$, so that Eqs. (A.20) and (A.21) are true. Then, there is a set of numbers for any $\varphi(\omega_{0:k}) \in L_{k,4}$. The numbers $b_{k|k}(\varphi(\omega_{0:k}))$ and $m_{k|k}(\varphi(\omega_{0:k}))$ independent of N make the following two formulas true.*

$$E[((q^{N}_{0:k|k}(d\omega_{0:k}), \varphi(\omega_{0:k})) - (q_{0:k|k}(d\omega_{0:k}), \varphi(\omega_{0:k})))^4] \le \frac{b_{k|k}(\varphi(\omega_{0:k}))}{N^2} \tag{A.22}$$

$$E[(q^{N}_{0:k|k}(d\omega_{0:k}), \varphi(\omega_{0:k}))^4] \le m_{k|k}(\varphi(\omega_{0:k})) \tag{A.23}$$

According to Lemmas 3–9, the following conclusions can be drawn.

Conclusion 1. When Hypotheses 1–3 are satisfied, the following formula holds for any $\varphi(\omega_{0:k}) \in L_{k,4}$:

$$e^N_{HAPFNN}[\varphi(\omega_{0:k})] \to e_{optimal}[\varphi(\omega_{0:k})] \quad a.s. \tag{A.24}$$

Since MSI-HAPFNN performs linear optimization resampling on the basis of HAPFNN, this approach does not change the robustness of its algorithm. So the MSI-HAPFNN finally obtained in this paper is still convergent.

References

[1] C. Yang, L. Zhou, K. Huang, H. Ji, C. Long, X. Chen, Y. Xie, Multimode process monitoring based on robust dictionary learning with application to aluminium electrolysis process, Neurocomputing 332 (2019) 305–319.

[2] W. Gui, W. Yue, Y. Xie, H. Zhang, C. Yang, A review of intelligent optimal manufacturing for aluminum reduction production, Zidonghua Xuebao/Acta Autom. Sin. 44 (11) (2018) 1957–1970.

[3] N.A. Menad, Z. Noureddine, A. Hemmatisarapardeh, S. Shamshirband, Modeling temperature-based oil-water relative permeability by integrating advanced intelligent models with grey wolf optimization : application to thermal enhanced oil recovery processes, Fuel 242 (2019) 649–663.

[4] F. Allard, M. Désilets, M. LeBreux, A. Blais, Improved heat transfer modeling of the top of aluminum electrolysis cells, Int. J. Heat Mass Transfer 132 (2019) 1262–1276.

[5] T. A, Device used for handling equipment of electrolysis cell to produce aluminum by igneous electrolysis, 2016, WO 2016174313-A1.

[6] J. Xiao, J. Yuan, Z. Tian, K. Yang, Z. Yao, B. Yu, L. Zhang, Comparison of ultrasound-assisted and traditional caustic leaching of spent cathode carbon (SCC) from aluminum electrolysis, Ultrason. Sonochem. 40 (2018) 21–29.
[7] F. Allard, M. Désilets, A. Blais, A modeling approach for time-dependent geometry applied to transient heat transfer of aluminum electrolysis cells, Metall. Mater. Trans. B 50 (2) (2019) 958–980.
[8] S. Zhan, Y. Huang, Z. Wang, C. Li, J. Yang, J. Wang, CFD simulations of gas–liquid multiscale flow characteristics in an aluminum electrolysis cell with population balance model: Effect of anode slot configuration, JOM 71 (1) (2019) 23–33.
[9] Z. Chen, Y. Li, X. Chen, C. Yang, W. Gui, Semantic network based on intuitionistic fuzzy directed hyper-graphs and application to aluminum electrolysis cell condition identification, IEEE Access 5 (2017) 20145–20156.
[10] W. Yue, X. Chen, W. Gui, Y. Xie, H. Zhang, A knowledge reasoning fuzzy-Bayesian network for root cause analysis of abnormal aluminum electrolysis cell condition, Front. Chem. Sci. Eng. 11 (3) (2017) 414–428.
[11] Q. Sun, S. Chen, L. Chen, D. Ma, Quasi-Z-source network-based hybrid power supply system for aluminum electrolysis industry, IEEE Trans. Ind. Inf. 13 (3) (2016) 1141–1151.
[12] J. Yi, Bai, Recurrent neural network and preference information based aluminum electrolytic modeling and optimizing method, 2018, CN 109086469-A.
[13] J. Yi, J. Bai, W. Zhou, H. He, L. Yao, Operating parameters optimization for the aluminum electrolysis process using an improved quantum-behaved particle swarm algorithm, IEEE Trans. Ind. Inf. 14 (8) (2017) 3405–3415.
[14] J. Yi, D. Huang, S. Fu, H. He, T. Li, Multi-objective bacterial foraging optimization algorithm based on parallel cell entropy for aluminum electrolysis production process, IEEE Trans. Ind. Electron. 63 (4) (2015) 2488–2500.
[15] J. Yi, Bai, Aluminum electrolysis preference algorithm optimization involves selecting control parameters to affect current efficiency, cell voltage, and perfluorinated emissions constitute decision variable, 2018, CN 109085752-A.
[16] K. Huang, H. Wen, H. Ji, L. Cen, X. Chen, C. Yang, Nonlinear process monitoring using kernel dictionary learning with application to aluminum electrolysis process, Control Eng. Pract. 89 (2019) 94–102.
[17] L. Yao, T. Li, Y. Li, L. Wei, J. Yi, An improved feed-forward neural network based on UKF and strong tracking filtering to establish energy consumption model for aluminum electrolysis process, Neural Comput. Appl. 31 (8) (2019) 4271–4285.
[18] L. Jiejia, Z. Peng, P. Jinxiang, Multi-fault diagnosis of aluminum electrolysis based on modular fuzzy neural networks, Asian J. Chem. 26 (11) (2014) 3339–3343.
[19] K. Zhou, D. Yu, Z. Lin, B. Cao, Z. Wang, S. Guo, Anode effect prediction of aluminum electrolysis using GRNN, in: 2015 Chinese Automation Congress (CAC), IEEE, 2015, pp. 853–858.
[20] K. Peng, R. Jiao, J. Dong, Y. Pi, A deep belief network based health indicator construction and remaining useful life prediction using improved particle filter, Neurocomputing 361 (2019) 19–28.
[21] M. Tian, Y. Bo, Z. Chen, P. Wu, C. Yue, Multi-target tracking method based on improved firefly algorithm optimized particle filter, Neurocomputing 359 (2019) 438–448.
[22] A. Dineva, A. Mosavi, S.F. Ardabili, I. Vajda, S. Shamshirband, T. Rabczuk, K. Chau, Review of soft computing models in design and control of rotating electrical machines, Energies 12 (6) (2019) 1049.
[23] M. Acosta, S. Kanarachos, Tire lateral force estimation and grip potential identification using neural networks, extended Kalman filter, and recursive least squares, Neural Comput. Appl. 30 (11) (2018) 3445–3465.
[24] S.-H. Hur, Estimation of useful variables in wind turbines and farms using neural networks and extended Kalman filter, IEEE Access 7 (2019) 24017–24028.
[25] Q. Zhu, Y. Han, P. Liu, Y. Xiao, P. Lu, C. Cai, Motion planning of autonomous mobile robot using recurrent fuzzy neural network trained by extended Kalman filter, Comput. Intell. Neurosci. (2019) http://dx.doi.org/10.1155/2019/1934575.
[26] L.J. Yao L, SUKFNN algorithm based aluminum electrolysis cost model constructing method, involves correcting predicted state parameter, and establishing aluminum electrolytic consumption model according to correction result, 2018, CN 108038330-A.
[27] L. Xiaodong, L. Aijun, Y. Changjun, S. Fulin, Widely linear quaternion unscented Kalman filter for quaternion-valued feedforward neural network, IEEE Signal Process. Lett. 24 (9) (2017) 1418–1422.
[28] S.K. Dash, R. Bisoi, P. Dash, A hybrid functional link dynamic neural network and evolutionary unscented Kalman filter for short-term electricity price forecasting, Neural Comput. Appl. 27 (7) (2016) 2123–2140.
[29] T. Novi, R. Capitani, C. Annicchiarico, An integrated artificial neural network–unscented Kalman filter vehicle sideslip angle estimation based on inertial measurement unit measurements, Proc. Inst. Mech. Eng. D 233 (7) (2019) 1864–1878.
[30] M. Masoumnezhad, A. Jamali, N. Nariman-Zadeh, Robust GMDH-type neural network with unscented Kalman filter for non-linear systems, Trans. Inst. Meas. Control 38 (8) (2016) 992–1003.
[31] X. Wang, Y. Wang, W. Wan, J.-N. Hwang, Object tracking with sparse representation and annealed particle filter, Signal Image Video Process. 8 (6) (2014) 1059–1068.
[32] H. Zhao, O. Kim, J.-S. Won, D.-J. Kang, Lane detection and tracking based on annealed particle filter, Int. J. Control Autom. Syst. 12 (6) (2014) 1303–1312.
[33] S.-K. Liao, Y.-C. Shu, X. Liu, Optimal estimation for the Fujino–Morley interpolation error constants, Japan J. Ind. Appl. Math. 36 (2) (2019) 521–542.
[34] H.-S. Jang, M.S. Muhammad, T.-S. Choi, Optimal depth estimation using modified Kalman filter in the presence of non-Gaussian jitter noise, Microsc. Res. Tech. 82 (3) (2019) 224–231.
[35] M. Breda, Appendix 1: Convergence aspects on back propagation neural networks, Subst. Use Misuse 33 (2) (1998) 503–514.
[36] Y. Qu, E. Zhang, Y. Jingyu, Convergence property of a generic particle filter algorithm, J. Comput. Res. Dev. 047 (1) (2010) 130–139.
[37] X. Hu, T.B. Schon, L. Ljung, A basic convergence result for particle filtering, IEEE Trans. Signal Process. 56 (4) (2008) 1337–1348.
[38] S.M. Colby, R.S. McClure, C.C. Overall, R.S. Renslow, J.E. McDermott, Improving network inference algorithms using resampling methods, BMC Bioinform. 19 (1) (2018) http://dx.doi.org/10.1186/s12859--018--2402--0.
[39] J. Park, M. Jeon, W. Pedrycz, Score-based resampling method for evolutionary algorithms, IEEE Trans. Syst. Man Cybern. B 38 (5) (2008) 1347–1355.
[40] X. Chen, Z. Xie, X. Liu, L. Xue, Y. Zhao, Research on application of hybrid annealed particle filter algorithm in mimo-ofdm channel estimation, J. Air Force Eng. Univ. (Nat. Sci. Ed.) 112 (3) (2016) 255–284.
[41] X. Ran, J. Tao, S. He, Gaussian mixture particle probability hypothesis density filter based on fuzzy hybrid annealed distribution in multi-target tracking, J. Proj. Rocket. Missiles Guid. 39 (3) (2019) 130–134,139.
[42] P.T. Roy, L. Jofre, J. Jouhaud, B. Cuenot, Versatile sequential sampling algorithm using kernel density estimation, European J. Oper. Res. 284 (1) (2020) 201–211.
[43] Y. Wang, R. Tan, Z. Zheng, F. Ni, X. Xiao, Method to estimate sag frequency in doubly fed induction generator integrated power system based on adaptive kernel density estimation, Iet Gener. Transm. Distrib. 14 (7) (2020) 1261–1270.
[44] C. Gonzales, S. Dubuisson, Combinatorial resampling particle filter: An effective and efficient method for articulated object tracking, Int. J. Comput. Vis. 112 (3) (2015) 255–284.
[45] Y. Shin, Z. Kim, J. Yu, G. Kim, S. Hwang, Development of NOx reduction system utilizing artificial neural network (ANN) and genetic algorithm (GA), J. Cleaner Prod. 232 (2019) 1418–1429.
[46] J. Zhao, Y. Ma, Z. Zhang, S. Wang, S. Wang, Optimization and matching for range-extenders of electric vehicles with artificial neural network and genetic algorithm, Energy Convers. Manage. 184 (2019) 709–725.
[47] H. Zhou, W.-F. Ding, Z. Li, H.-H. Su, Predicting the grinding force of titanium matrix composites using the genetic algorithm optimizing back-propagation neural network model, Proc. Inst. Mech. Eng. B 233 (4) (2019) 1157–1167.
[48] L.I. Tai-Fu, L.Z. Yao, Y.I. Jun, H.U. Wen-Jin, S.U. Ying-Ying, W. Jia, An improved UKFNN based on square root filter and strong tracking filter for dynamic evolutionary modeling of aluminum reduction cell, Acta Automat. Sinica 40 (3) (2014) 522–530.
[49] M.M.E. Genidy, Multiple nonlinear regression of the Markovian arrival process for estimating the daily global solar radiation, Commun. Statist.-Theory Methods 48 (22) (2019) 5427–5444.
[50] Z. Guo, L. Song, C. Park, J. Li, R.T. Haftka, Analysis of dataset selection for multi-fidelity surrogates for a turbine problem, Struct. Multidiscip. Optim. 57 (6) (2018) 2127–2142.
[51] C.H. Aladag, A new multiplicative seasonal neural network model based on particle swarm optimization, Neural Process. Lett. 37 (3) (2013) 251–262.
[52] W. Li, D. Kong, J. Wu, A novel hybrid model based on extreme learning machine, k-nearest neighbor regression and wavelet denoising applied to short-term electric load forecasting, ENERGIES 10 (5) (2017) 694–710.