sensors

MDPI

Article

# A Sensor Dynamic Measurement Error Prediction Model Based on NAPSO-SVM

**Minlan Jiang [1,*], Lan Jiang [1], Dingde Jiang [2,*], Fei Li [1] and Houbing Song [3]**

[1] College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua 321004, China; sa1105587@163.com (L.J.); m18329019792@163.com (F.L.)
[2] School of Astronautics and Aeronautic, University of Electronic Science and Technology of China, Chengdu 611731, China
[3] Department of Electrical, Computer, Software, and Systems Engineering, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA; h.song@ieee.org
* Correspondence: xx99@zjnu.cn (M.J.); jiangdd@uestc.edu.cn (D.J.); Tel.: +86-138-6797-9259 (M.J.)

Received: 16 November 2017; Accepted: 11 January 2018; Published: 15 January 2018

**Abstract:** Dynamic measurement error correction is an effective way to improve sensor precision. Dynamic measurement error prediction is an important part of error correction, and support vector machine (SVM) is often used for predicting the dynamic measurement errors of sensors. Traditionally, the SVM parameters were always set manually, which cannot ensure the model's performance. In this paper, a SVM method based on an improved particle swarm optimization (NAPSO) is proposed to predict the dynamic measurement errors of sensors. Natural selection and simulated annealing are added in the PSO to raise the ability to avoid local optima. To verify the performance of NAPSO-SVM, three types of algorithms are selected to optimize the SVM's parameters: the particle swarm optimization algorithm (PSO), the improved PSO optimization algorithm (NAPSO), and the glowworm swarm optimization (GSO). The dynamic measurement error data of two sensors are applied as the test data. The root mean squared error and mean absolute percentage error are employed to evaluate the prediction models' performances. The experimental results show that among the three tested algorithms the NAPSO-SVM method has a better prediction precision and a less prediction errors, and it is an effective method for predicting the dynamic measurement errors of sensors.

**Keywords:** sensors; dynamic measurement errors; prediction; improved PSO; support vector machine

## 1. Introduction

Today, sensors are widely used in the real world [1–4], and sensor errors are one of the keys to evaluate the measurement quality of the sensor results. With the development of modern measurement technology, dynamic measurements have gradually become the mainstream of modern measurement techniques.

As an effective way to improve measurement accuracy and reduce measurement errors, real-time error correction of sensors has been widely used in dynamic measurements. Predicting the dynamic measurement error is useful to correct the errors of sensors. Dynamic measurement errors of sensors are difficult to model with traditional mathematics because they have four features [5]: time-varying, randomness, correlation and dynamic. Because of its complexity, predicting the dynamic errors has been a popular research field [6–9].

In recent years, several modeling methods were used to predict dynamic errors like the gray theory, Bayesian networks and neural network. Every method has its own advantages and drawbacks. The harmonic analysis method is suitable for modeling periodic sequences, but it is not suitable for random sequences [10]. Bayesian networks is a prediction modeling method, however, it requires

the prior distribution and independent samples, which is difficult to achieve in real systems [11]. Grey theory model can be constructed by a few samples, but it only depicts a monotonically increasing or decreasing process [12,13]. Artificial neural network has a good non-linear mapping performance, however, it has disadvantages, such as over-fitting and easily falling into a local minimum [14–16].

Support vector machine (SVM) adopts structural risk minimization to improve its generalization ability [17]. It can better solve the problems of nonlinear data and small samples. SVM has been widely applied to solve function fitting problems. However, the generalization ability of SVM depends heavily on the appropriate parameters, and the model's parameters have a huge influence on the precision of the model predictions [18]. Thus, many optimization algorithms have been adopted to optimize the SVM parameters [19–21], like the particle swarm optimization (PSO) algorithm, genetic algorithm (GA) and glowworm swarm optimization (GSO) algorithm. These methods have limitations, as the particle swarm optimization and genetic algorithm fall into local extremes easily [22], and the glowworm swarm optimization algorithm has low convergence precision and slow convergence speed [23]. The NAPSO algorithm is an improved particle swarm optimization algorithm based on the natural selection strategy and simulated annealing mechanism. These two methods are used to improve the global search ability and convergence speed. In this study, a method of dynamic measurement error prediction for sensors based on NAPSO optimize support vector machine is proposed.

The rest of the paper is organized as follows: in Section 2, a detailed overview of the SVM algorithm is provided. Then, in Section 3, the PSO and NAPSO algorithms and the process of optimization are described briefly. Section 4 reports on a simulation of the dynamic measurement error prediction model. The results of experiments are discussed in Section 5. Conclusions are drawn in the last section.

## 2. SVM Algorithm

### 2.1. SVM

SVM is a machine learning method based on the statistical learning theory developed in mid-1990s. The basic idea of SVM is that the data of input space $R^n$ are mapped to a high dimensional feature space $F$ by a nonlinear mapping, then the linear regression operations are finished in the high dimensional feature space [24].

For a given training dataset $\{(x_i,y_i), i = 1,2, \ldots ,n\}$, $x_i$ is a n-dimensional input vector and $y_i$ is the corresponding output value, $\varphi(x)$ is the nonlinear mapping from input space $R^n$ to high dimensional feature space $F$.

$$R^n \to F : x \to \varphi(x) \tag{1}$$

The regression function of SVM is formulated as follows:

$$f(x) = [\omega \times \varphi(x)] + b \quad \omega \in R^m, b \in R \tag{2}$$

where $\omega$ is the weight vector and $b$ is the threshold, the main goal of the SVM is to find the optimal $\omega$, the weight vector $\omega$ can be expressed by a linear combination of the training examples. Thus, the Equation (2) takes the form:

$$f(x) = \sum_{i=1}^{n} \alpha_i {x_i}^T x + b \quad b \in R \tag{3}$$

where $\alpha_i$ are Lagrangian multipliers. In the feature space, Equation (4) can be expressed as follows:

$$f(x) = \sum_{i=1}^{n} \alpha_i \varphi({x_i}^T) \varphi(x_j) + b \quad b \in R \tag{4}$$

Introduce the Kernel function $K(x_i,y_j)$:

$$K(x_i, x_j) = \varphi(x_i{}^T)\varphi(x_j) \tag{5}$$

Finnally, the SVM regression function is formulated as:

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x_j) + b \tag{6}$$

### 2.2. Kernel Function

Kernel function is a key concept of SVM, the performance of SVM mainly depends on the kernel function. As shown in the Equation (1), the kernel function establishes a relation between the input space $R^n$ and the high dimensional feature space $F$. Different selection of kernel functions will construct different regression models:

$$K(x_i, x_j) = \phi(x_i)^T \times \phi(x_j) \tag{7}$$

The common kernel functions include the polynomial kernel function, linear kernel function, fourier kernel function and radial basis function (RBF) kernel function. The kernel function parameters has a directly influence on the complexity of the function, RBF kernel function has the advantages of fewer parameters and good performance. Thus, RBF kernel function is used in this paper.

The RBF kernel function is expressed as follows:

$$K(x_i, x_j) = \exp\left\{-\frac{|x_i - x_j|^2}{2\sigma^2}\right\} \tag{8}$$

where $\sigma$ is the width coefficient of the kernel function.

The SVM parameters determine both its generalization ability and learning ability, the punishment coefficient $C$ and RBF kernel function width $\sigma$ have a direct impact on the accuracy and efficiency of the SVM prediction model. $C$ adjusts the balance between generalization and empirical error. When $C$ is greater, the model's complexity will be increased and it will fall into the "over-fitting" phenomenon easily, but if $C$ is too small, the model's complexity will be reduced and it will fall into the "under-fitting" phenomenon easily. The value of $\sigma$ affects the complexity of the sample data distribution in feature space. In this paper, NAPSO algorithm is used to optimize the two parameters to achieve better prediction results.

## 3. SVM Parameters Optimization Based on NAPSO

### 3.1. PSO

Particle swarm optimization was proposed by Eberhart and Kennedy in 1995 [25], PSO was derived from research on bird flocks' preying behavior. When a flock of birds is looking for food in an area randomly, if there is only one piece of food in the area being searched, the most effective and simple method to find the food is to follow the bird that is closest to the food.

In a PSO algorithm, every single solution is a particle in the search space. Each particle has a fitness value, which is determined by an optimization function, each particle has its own velocity and position. The velocity and position of each particle will be changed by the particle best position and global best position. The update equations of the velocity and position are shown by the following expression:

$$v_{i.d}(t+1) = \omega v_{i.d}(t) + c_1 r_1 [p_{best} - x_{i.d}(t))] + c_2 r_2 [g_{best} - x_{i.d}(t))] \tag{9}$$

$$x_{i.d}(t+1) = x_{i.d}(t) + v_{i.d}(t+1) \tag{10}$$

In the D-dimensional space, $t$ is the iteration number, $v_{i,d}(t)$ is the velocity of particle $i$ at iteration $t$, $v_{i,d}(t+1)$ is the velocity of particle $i$ at iteration $t+1$, $x_{i,d}(t)$ is the position of particle $i$ at iteration $t$, $x_{i,d}(t+1)$ is the position of particle $t$ at iteration $t+1$, $\omega$ is the inertia weight. $c_1$ is the cognition learning factor, $c_2$ is the social learning factor, $r_1$ and $r_2$ are random numbers that are uniformly distributed in [0,1], $p_{best}$ is the particle best position for the individual variable of particle $i$, $g_{best}$ is the global best position variable of the particle swarm.

The initial position and velocity of each particle are randomly generated and will be updated based on the Equations (9) and (10) until a satisfactory solution is found. In the PSO algorithm, a single particle moves to its $p_{best}$ and $g_{best}$, each particle's movement generates a fast convergence, thus the PSO algorithm converges rapidly. However, the fast convergence also makes the update of each particle depend too much on its $p_{best}$ and $g_{best}$, which makes the algorithm fall into local optima and easily converge prematurely. Therefore, in this paper, an improved PSO algorithm (NAPSO) is used to optimize the parameters of SVM.

### *3.2. NAPSO*

The NAPSO algorithm is an improved PSO algorithm based on the methods of natural selection and simulated annealing. In the NAPSO algorithm, the simulated annealing mechanism is used to improve the ability of the algorithm to jump out of a local optimum trap, while the natural selection method is employed to accelerate the rate of convergence.

The NAPSO algorithm starts with a set of random velocities and positions. Before the iteration, each particle's personal best position and global best position are calculated by the fitness function. Each particles update its velocity and position by Equations (9) and (10) at each iteration.

After updating a particle's speed, position $l$ and fitness value $f'$, the particle moves to a random position $l_1'$ in its neighborhood and computes its new fitness value $f_1'$. The movement formula is expressed as follows:

$$l_1' = l + r_3 \times [v_{max} - v_{min}] \times r_1 \tag{11}$$

where $r_3$ is the normally distribution random numbers of D-dimension that are distributed in [0,1], $v_{max}$ is the maximum value of the velocity, and $v_{min}$ is the minimum value of the velocity.

When $f_1' > g_{best}$, we keep the position $l$. When $f_1' < g_{best}$, if $f_1' > f'$, use the new position $l_1'$ to replace the position $l$ by the simulated annealing operation, where the operation of simulated annealing is expressed as follows:

$$l = \left\{ \begin{array}{ll} l & if\ \exp((-1) \times (f_1' - f')/T > r_4) \\ l_1' & if\ \exp((-1) \times (f_1' - f')/T <= r_4) \end{array} \right\} \tag{12}$$

where $r_4$ is a random number that is uniformly distributed in [0,1], $T$ is the simulated annealing temperature.

Each particle uses the simulated annealing operation to determine whether to accept the new position, and then updates the particle's $p_{best}$ and $g_{best}$ by its position. The simulated annealing operation can significantly enhance the ability of the algorithm to jump out of the local optimum trap. At the end of each iteration, all particles have been ranked by their fitness values, from best to worst, and using the better half to replace the other half. In this way, the stronger adaptability particles are saved. Finally, the NAPSO algorithm is terminated by the satisfaction of a termination criterion.

The pseudo code of the NAPSO algorithm is presented as the Algorithm 1.

**Algorithm 1:** NAPSO

**Input** $\omega$, $c_1$, $c_2$, $T$
**Output** $g_{best}$
**Initialization:** $x$, $p_{best}$, $g_{best}$
**while** $t$ < maximum number of iterations and $g_{best}$ > minimum fitness **do**
  **for** each particle **do**
    update the velocity $v$, position $l$, and fitness $f'$
    find a new position $l'_1$ in the neighborhood and calculate its fitness value $f'_1$
    **if1** ($f'_1 < g_{best}$) **then**
      **if2** ($f'_1 - f' < 0$) **then**
        accept the new position $l'_1$
      **else if2**
        accept the new position $l'_1$ by the simulated annealing operation
      **end if2**
    **else if1**
      accept the old position $l$
    **end if1**
    update the $p_{best}$, $g_{best}$ and Simulated temperature $T$
  **end for**
  rank all particles by their fitness value, use the better half to replace the other half.
  $t = t + 1$
**end while**
return the $g_{best}$

The simulated annealing operation will slow the rate of convergence, thus increasing the convergence time. The natural selection operation will reduce the diversity of samples. However, these two operations can compensate for each other, as the simulated annealing operation can increase sample diversity, and the natural selection operation can speed up the convergence rate. These two operations are used to both ensure the convergence rate of the algorithm and guarantee that the ability of the algorithm to jump out of the local optimal trap can be enhanced.

### *3.3. Optimization Process*

The NAPSO algorithm is applied to optimize the SVM parameters $C$ and $\sigma$ as follows:

Step 1: Initialize the NAPSO algorithm, set the number of particles velocity, particles positions and the other parameters. Because the search space is 2-dimensional, the position of each particle contains two variables. Set $T$ to be the simulated temperature; the initial $T$ is 5000 °C, and the lower limit of $T$ is 1 °C. Calculate the fitness value of each particle. The fitness evaluation function is defined as follows:

$$J = \sum_{i=1}^{n} (Y_i - Y'_i)^2 / n \tag{13}$$

where $Y_i$ is the actual value, $Y'_i$ is the predicted value and $n$ is the number of training samples.

Step 2: According to the fitness value of each particle to set the personal best position $p_{best}$ and global best position $g_{best}$.

Step 3: Update the position $l$ and velocity of each particle. Evaluate the fitness value $f'$. Then, randomly find a new position $l'_1$ in the neighborhood of the particle, calculate the new fitness value ($f'_1$) of the new position.

Step 4: Calculate the difference between the fitness value $f'$ and the new fitness value $f'_1$, $\Delta f = f'_1 - f'$.

Step 5: When $f'_1 \geq g_{best}$, keep the original position $l$. When $\Delta f > 0$ and $f'_1 < g_{best}$, according the Equation (12) accept the new position $l'_1$, if $\Delta f < 0$ and $f'_1 < g_{best}$, replace the original position with the new position. Then, update the $p_{best}$ and $g_{best}$.

Step 6: When the updates of each particle has completed, then rank all of the particles according to the each particle's fitness value, employ the better half particles' information to replace the other half particles' information and update the temperature $T = T \times 0.9$.

Step 7: If the number of iterations is equal to the maximum iterations or the $g_{best}$ is less than or equal to the least fitness, output the two variables of the $g_{best}$; otherwise, return to Step 2.

## 4. Experiments

### *4.1. Data Description*

In this paper, two cases have been considered to illustrate the effectiveness of the proposed method. The data of case 1 is a dynamic error sequence, which is derived from the measuring error of an angular instrument with anticlockwise rotation (speed 2r/min) based on standard value interpolation under room temperature, the error sequence contains a total of 240 samples. In case 2, the measuring error sequence of the length grating contains a total 334 samples. A high precision Hewlett Packard 5529A laser interferometer was used as calibration system, A-quad-B pulse from the encoder of grating length gauge is used to trigger the laser interferometer and carry out dynamic data acquisition. The software package of the HP laser interferometer is used to implement data acquisition. The speed of the target mirror is 21.1 mm/s, the experiment has been repeated five times.

### *4.2. Preprocessing*

The two datasets are both one-dimensional data, so in order to achieve the better predict results and get more information from the data, these two one-dimensional data must be converted to multi-dimensional data [26,27]. Assuming $p$ is the dimension of the input vector, the reconstructed samples are listed in Table 1.

According to the reconstructed method listed in the Table 1, in case 1,the dimension number $p$ is 16, the number of restructured sample is 224, selecting the first 124 samples for training and the final 100 samples for testing. The proportion of training samples to testing samples is 1.24:1, in case 2, the dimension $p$ is 20, the number of restructured sample is 315, the first 200 samples are selected as training data and the rest are used as testing data. The proportion of training samples to testing samples is 1.73:1.

**Table 1.** Reconstructed samples.

| Input | Output |
|---|---|
| $X(1), X(2), \cdots, X(p)$ | $X(p+1)$ |
| $X(2), X(3), \cdots, X(p+1)$ | $X(p+2)$ |
| $\cdots$ | $\cdots$ |
| $X(n-p), X(n-p+1), \cdots, X(n-1)$ | $X(n)$ |

Preprocess the data by the normalized method, then perform parameter optimization and train the model.

### *4.3. Valuation Index*

To further evaluate the prediction of the NAPSO-SVM model, the root mean square error (RMSE) and mean absolute percent error (MAPE) are used as evaluation indices. The definitions of MAPE and RMSE are as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - Y_i')^2} \tag{14}$$

$$\mathrm{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - Y_i'}{Y_i}\right| \tag{15}$$

where $Y_i$ is the actual value, $Y_i'$ is the prediction value and $n$ is the number of training samples.

The NAPSO algorithm is used to determine the punishment coefficient $C$ and RBF kernel function width $\sigma$. The SVM model is built based on the training samples and optimal parameters. To show the performance of the proposed method, the particles swarm optimization and glowworm swarm optimization are also implemented.

### *4.4. GSO Algorithm*

Glowworm swarm optimization (GSO) is a new swarm intelligence optimization algorithm, proposed by Krishnanand and Ghose [28] in 2006. GSO has some apparent differences with PSO, GSO is appropriate for multiple optima of multimodal functions. It simulates the luciferin properties of glowworms, by comparing the luciferin value to exchange information.

Like the PSO algorithm, in the GSO algorithm, each single solution is a glowworm in the search space. Each glowworm has its luciferin and neighborhood range. The value of luciferin is used to measure the quality of the solution. The value of the neighborhood range is used to determine the search scope of the glowworm. According a probabilistic mechanism, each glowworm move toward a neighbor that has a luciferin higher than its own.

Except initialization, GSO algorithm has three phases. In the first phase, each glowworm updates its luciferin, the luciferin update rule has the following equation:

$$l_i(t+1) = (1-\rho)l_i(t) + \gamma J_i(t+1) \tag{16}$$

where $l_i(t)$ is the luciferin level for glowworm $i$ at iteration $t$, $\rho$ is the luciferin decay constant. $\gamma$ is the luciferin enhancement constant and $J_i(t+l)$ is the fitness value of the objective function at glowworm $i$ at iteration $t+l$.

In the second phase, each glowworm finds neighbors that have a luciferin level higher than its own. The set of neighbors of glowworm $i$ at iteration $t$ is expressed as follows:

$$N_i(t) = \left\{ j : d_{ij}(t) < r_d^i(t); l_i(t) < l_j(t) \right\} \tag{17}$$

where $d_{ij}(t)$ is the Euclidean distance between glowworms $i$ and $j$ at iteration $t$, $r_d^i(t)$ is the variable neighborhood range at glowworm $i$ at iteration $t$. The probability of glowworm $t$ moving toward its neighbor $j$ is calculated by the following equation:

$$p_{ij}(t) = \frac{l_j(t) - l_i(t)}{\sum_{k \in N_i(t)} l_k(t) - l_i(t)} \tag{18}$$

Then, the rule of the glowworm movements is given by Equation (19):

$$x_i(t+1) = x_i(t) + s\left[ \frac{x_j(t) - x_i(t)}{\|x_j(t) - x_i(t)\|} \right] \tag{19}$$

where $x_i(t)$ is the position of glowworm $i$ at iteration $t$, $\|x_j(t) - x_i(t)\|$ is the distance between glowworms $i$ and $j$ at iteration $t$, s represents the step size.

In the third phase, each glowworm updates its neighborhood range by the following equation:

$$r_d^i(t+1) = min\left\{ r_s, max\left\{ 0, r_d^i(t) + \beta(n_t - |N_i(t)|) \right\} \right\} \tag{20}$$

where $\beta$ and $n_t$ are constant parameters. $r_s$ is the neighborhood range bound ($r_d^i(t) \leq r_s$).

## 5. Results

Our study used the Libsvm toolbox v3.21 (written by Chih-Jen Lin), and our algorithm are coded using Matlab 2010a and all experiments are implemented on a 4 GB RAM, 3.20 GHz Intel core i5 machine equipped with the Windows 7 operating system. In case 1, the prediction results of the three models are shown in Figures 1–3, respectively.

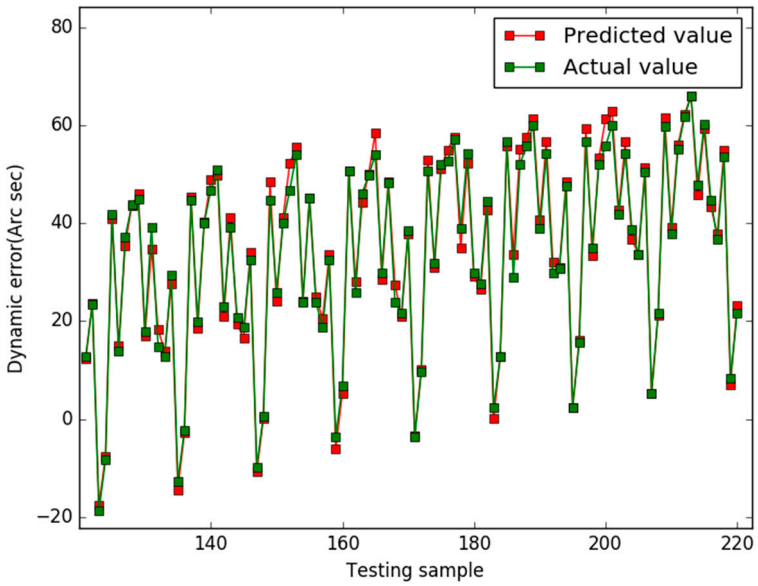


**Figure 1.** Predicted results of the NAPSO-SVM (case 1).

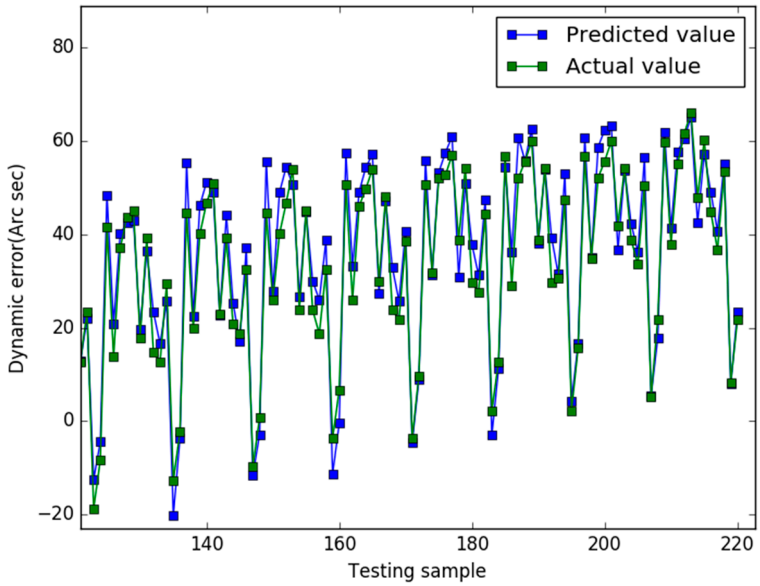


**Figure 2.** Predicted results of the PSO-SVM (case 1).

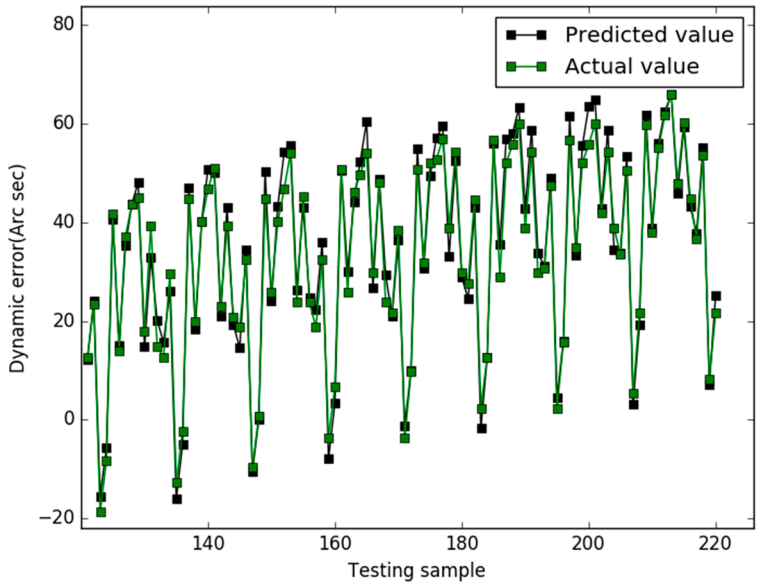


**Figure 3.** Predicted results of the GSO-SVM (case 1).

To make a fair comparison, the maximum generation, population size, minimum fitness value, range of gains, dimension of search space and initial positions are identical for all the algorithms. The maximum number of generations is 100, the minimum fitness value is 0.1, the size of the population is 100, and the dimension of the search space is 2. The parameters for NAPSO were set as follows: the inertia weight $w$ = 0.9, the acceleration constants $c_1$ and $c_2$ are 2, the initial temperature is 10,000 °C, the lower limit of temperature is 1 °C, the maximum value of velocity is 1, the minimum value of velocity is −1. In GSO algorithm, the light absorption coefficient is 50, the minimum value of attractiveness is 0.8, the maximum value of attractiveness is 1.0, the value of initial step size factor is 0.5. The PSO algorithm has the same inertia weight and acceleration constant as the NAPSO algorithm.

Figure 4 presents the comparison results of predicted residuals by the three models. The MAPE value and RMSE value of the three models are listed in Table 2.

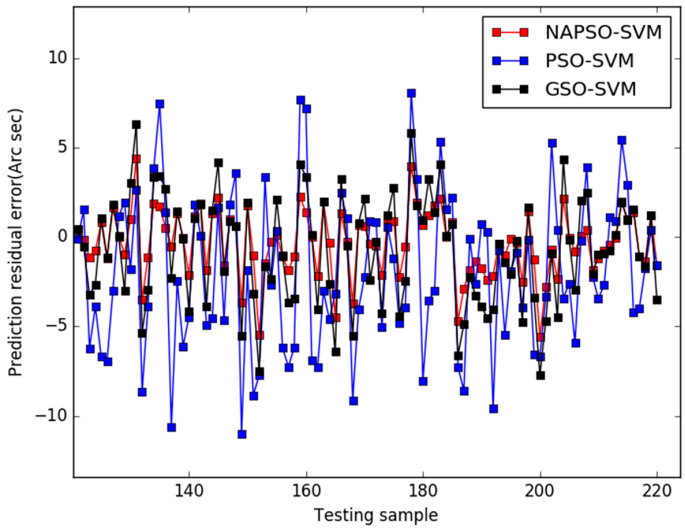


**Figure 4.** Comparison of three models for predicted residuals (case 1).

In Figure 1, for the NAPSO-SVM model, the curve of predicted value is very close to the curve of the actual value. In Figure 2, the predicted value curve often lags the actual value curve. In Figure 3,

the predicted value curve is close to the actual value curve, but there are many points are still outside the range of the actual curve. We could find that compared with the obvious hysteresis of the PSO-SVM and GSO-SVM models, there is a negligible hysteresis between the predicted and actual value curves with the NAPSO-SVM model, and it is closer to the actual value. We find that the NAPSO-SVM model outperforms the PSO-SVM and GSO-SVM model. The prediction performance of NAPSO-SVM is better than GSO-SVM model and accuracy much better than PSO-SVM.

The residual curves of the three models are shown in the Figure 4. The prediction residual curve of the PSO-SVM model is large, ranging from −11″ to 8″, and the prediction residual of the GSO-SVM model is smaller than the PSO-SVM model. But it is still relatively large, ranging from −8″ to 6″. The predicted residual of the NAPSO-SVM is smaller than the others and tends to more gentle, ranging from −5″ to 4″.The results prove that dynamic measurement error prediction ability of NAPSO-SVM model is better than PSO-SVM and GSO-SVM model, and the NAPSO algorithm is an effective method for parameters optimization. To further verify the ability of the three models. Table 2 lists the comparison results between the three models for prediction accuracy indexes.

**Table 2.** Comparison of the index value among the three models (case 1).

| MODEL | MAPE | RMSE |
|---|---|---|
| NAPSO-SVM | 0.0744 | 0.1879 |
| PSO-SVM | 0.2423 | 0.4710 |
| GSO-SVM | 0.1493 | 0.3128 |

In Table 2, the MAPE value and RMSE value of the NAPSO-SVM model are smaller than those of the PSO-SVM and GSO-SVM models. The MAPE value is approximately 0.0744 for NAPSO-SVM model, compared with approximately 0.2423 and 0.1493 for the PSO-SVM and GSO-SVM models, respectively. Furthermore, the RMSE value is 0.1876 in the case of the NAPSO-SVM model. Compared with the NAPSO-SVM model, the RMSE values of the GSO-SVM model and PSO-SVM model are 0.4710 and 0.3128, respectively. In summary, the results of the Table 2 are in accord with Figure 4, and the NAPSO-SVM model has the best dynamic measurement error prediction ability among the three methods.

In case 2, the parameters of each algorithm are essentially the same as the previous case, and the prediction results of three models are shown in Figures 5–7. Figure 8 shows the comparison results of the residuals predicted by the three models. The MAPE value and RMSE values of the three models are listed in the Table 3.

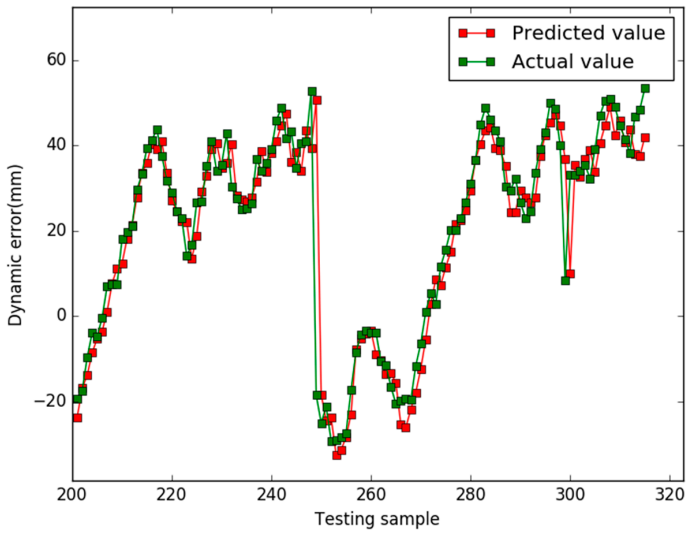


**Figure 5.** Predicted results of the NAPSO-SVM (case 2).

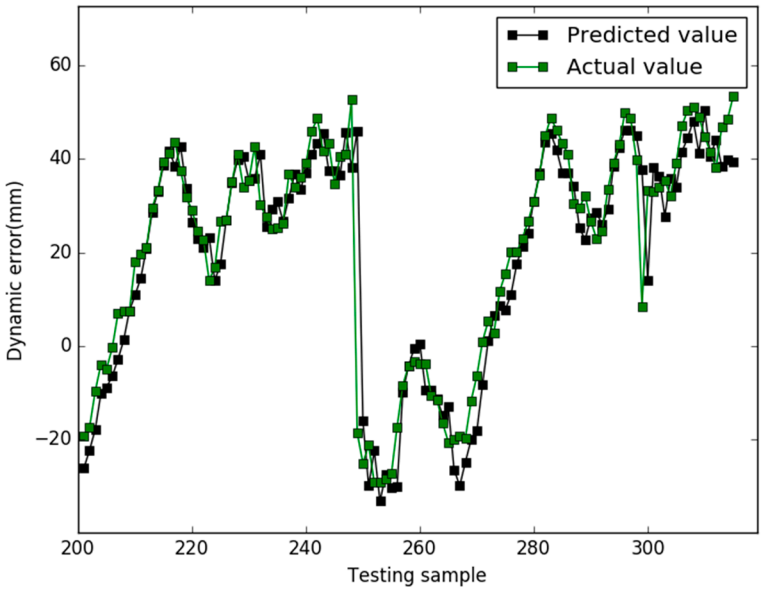


**Figure 6.** Predicted results of the PSO-SVM (case 2).

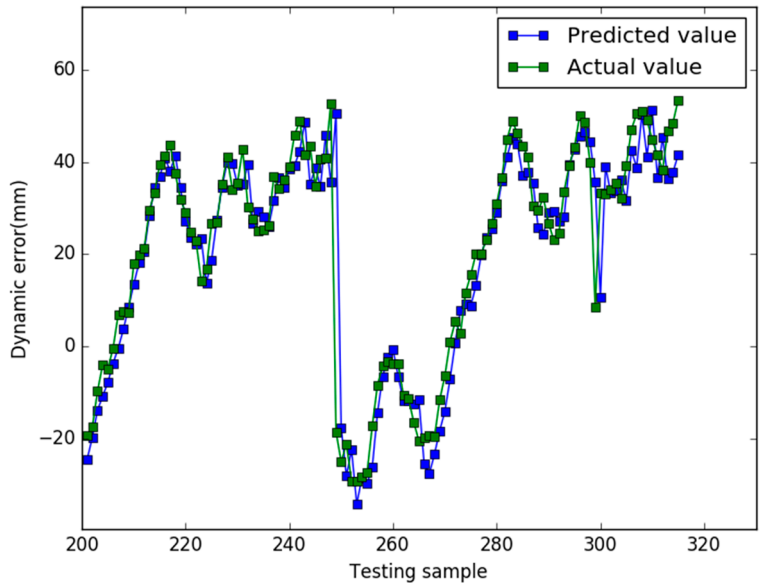


**Figure 7.** Predicted results of the GSO-SVM (case 2).

When the ratio of training samples and testing samples is approximately 1.73, in Figures 5–7, for all models, there is hysteresis between the predicted and actual value curves on the four points (testing samples 248, 249, 299, 300). Except for the four points, the prediction curve of the NAPSO-SVM model is closest to the actual value curve, and the prediction curve of the NAPSO-SVM model is approximately the same as the actual value curve. However, unlike in case 1, the prediction results of the PSO-SVM model is nearly the same as the GSO-SVM model, but the prediction curves of these two models still lag behind the actual value curve.

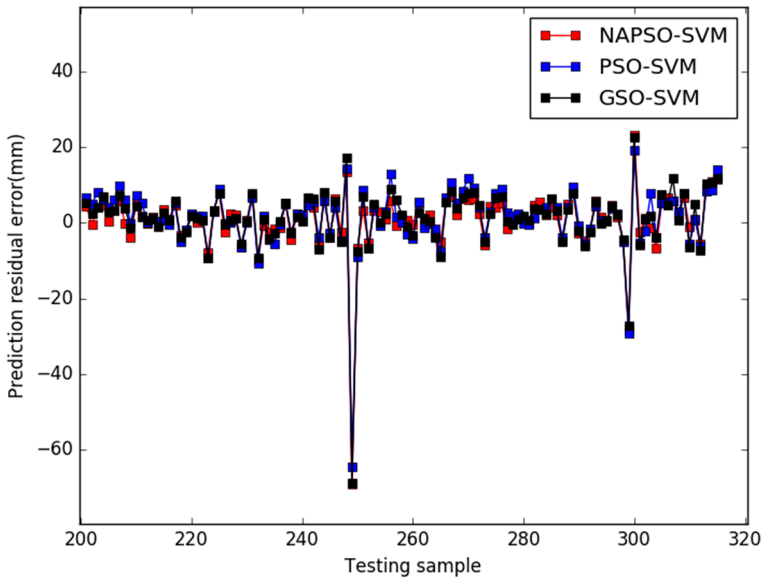


**Figure 8.** Comparison of the predicted residuals of the three models (case 2).

In Figure 8, it is easy to see the distribution of the four hysteresis points. The prediction residual errors are very big on the four hysteresis points. Except for the four points, the prediction residual of the NAPSO-SVM model is smallest among the three models, ranging from −9.9086 to 11.3979 mm. The prediction residual of the GSO-SVM model is ranging from −9.3916 to 11.8487 mm, and the prediction residual of the PSO-SVM model is ranging from −10.7265 to 14.0340 mm. The prediction residual range of the NAPSO-SVM model is almost equal to the GSO-SVM model, and better than the PSO-SVM model.

**Table 3.** Comparison of the index value among the three models (case 2).

| MODEL | MAPE | RMSE |
|---|---|---|
| NAPSO-SVM | 0.3840 | 0.8015 |
| PSO-SVM | 0.5377 | 0.8209 |
| GSO-SVM | 0.4403 | 0.8356 |

As Table 3 shows, the MAPE value is approximately 0.3840 for the NAPSO-SVM model compared with approximately 0.5377 and 0.4403 for the PSO-SVM model and GSO-SVM model. NAPSO-SVM model has the smallest RMSE value of the three algorithms, acquiring RMSE value of 0.8015 and the RMSE values of the PSO-SVM model and GSO-SVM model are 0.8209 and 0.8356, respectively. The prediction ability of the NAPSO-SVM model is better than the other models, and GSO-SVM model has the worst RMSE value, PSO-SVM model has the worst MAPE value.

Although the data of cases is the actual data, a Gaussian noise (SNR = 15 dB) are added into the data of case 2 to prove the ability of NAPSO-SVM model, the MAPE values of the NAPSO-SVM model, PSO-SVM model and GSO-SVM model are 0.4239, 0.6035 and 0.5839, respectively. The RMSE values of the NAPSO-SVM model, PSO-SVM model and GSO-SVM model are 0.9839, 0.9868 and 0.9867. NAPSO-SVM model still has a better performance.

The results of the two cases show that the NAPSO-SVM model has the best prediction accuracy among the three methods. This indicates that the NAPSO algorithm has a better global search capability than the other two algorithms, the reason being that the updating of the position and velocity of the particles in the PSO algorithm are too dependent on the current best particle. Compared with the PSO algorithm, the NAPSO algorithm uses simulated annealing and a natural selection mechanism, so it is easier jump out of the local trap and search for the global optimal solution in the global space.

## 6. Conclusions

Dynamic measurement has been a hot area of research for several years, and dynamic measurement error prediction is a useful method to improve sensor measurement accuracy. In this study, a method of dynamic measurement error prediction based on NAPSO-optimized SVM parameters is proposed. To improve the prediction accuracy, the NAPSO algorithm is used to optimize the SVM parameters to avoid the problems of "over-fitting" and "under-fitting" of SVM. The results of the two cases show that compared with the PSO-SVM and GSO-SVM models, the NAPSO-SVM model has the better prediction accuracy. The proposed method provides a new way for predicting sensors' dynamic measurement errors and has definite value for application in dynamic measurement. However, like the standard PSO, NAPSO has intrinsic property randomness. In the future, we plan to study which is the more effective method for improving the prediction results.

**Acknowledgments:** The work was supported in part by the National Natural Science Foundation of China (Nos. 51305407, 61571104, and 61071124), the General Project of Scientific Research of the Education Department of Liaoning Province (No. L20150174), the Program for New Century Excellent Talents in University (No. NCET-11-0075), the Fundamental Research Funds for the Central Universities (Nos. ZYGX2017KYQD170, N150402003, N120804004 and N130504003), and the State Scholarship Fund (201208210013).

**Author Contributions:** M.J. and L.J. were the main designers of the proposed scheme and wrote the paper. F.L. focused on the research, and part of the paper writing, D.J. and H.S. revised the paper and analyzed the results.

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

## References

1. Cheng, S.; Cai, Z.; Li, J.; Gao, H. Extracting Kernel Dataset from Big Sensory Data in Wireless Sensor Networks. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 813–827. [CrossRef]
2. Jiang, D.; Li, W.; Lv, H. An energy-efficient cooperative multicast routing in multi-hop wireless networks for smart medical applications. *Neurocomputing* **2017**, *220*, 160–169. [CrossRef]
3. Jiang, D.; Wang, Y.; Han, Y.; Lv, H. Maximum connectivity-based channel allocation algorithm in cognitive wireless networks for medical applications. *Neurocomputing* **2017**, *220*, 41–51. [CrossRef]
4. Jiang, D.; Xu, Z.; Li, W.; Yao, C.; Lv, Z.; Li, T. An energy-efficient multicast algorithm with maximum network throughput in multi-hop wireless networks. *J. Commun. Netw.* **2016**, *18*, 713–724.
5. Yakovlev, V.T. Method of Determining Dynamic Measurement Error Due to Oscillographs. *Meas. Tech.* **1987**, *30*, 331–334. [CrossRef]
6. Chen, Y. Area-Efficient Fixed-Width Squarer with Dynamic Error-Compensation Circuit. *IEEE Trans. Circuits Syst. II* **2015**, *62*, 851–855. [CrossRef]
7. Jiang, D.; Nie, L.; Lv, Z.; Song, H. Spatio-temporal Kronecker compressive sensing for traffic matrix recovery. *IEEE Access* **2016**, *4*, 3046–3053. [CrossRef]
8. Yakovlev, V.T.; Collins, E.; Flores, K.; Pershad, P.; Stemkovski, M.; Stephenson, L. Statistical Error Model Comparison for Logistic Growth of Green Algae (Raphidocelis subcapitata). *Appl. Math. Lett.* **2017**, *64*, 213–222.
9. Cheng, S.; Cai, Z.; Li, J.; Fang, X. Drawing Dominant Dataset from Big Sensory Data in Wireless Sensor Networks. In Proceedings of the 34th Annual IEEE International Conference on Computer Communications, Kowloon, Hong Kong, China, 26 April–1 May 2015; pp. 531–539.
10. Trapero, J.; Kourentzes, N.; Martin, A. Short-term solar irradiation forecasting based on dynamic harmonic regression. *Energy* **2015**, *84*, 289–295. [CrossRef]
11. Yang, H.; Zhang, L.; Zhou, J.; Fei, Y.; Peng, D. Modelling of dynamic measurement error for parasitic time grating sensor based on Bayesian principle. *Opt. Precis. Eng.* **2015**, *24*, 2523–2531. [CrossRef]
12. Ge, L.; Zhao, W.; Zhao, S.; Zhou, J. Novel error prediction method of dynamic measurement lacking information. *J. Test. Eval.* **2012**, *40*. [CrossRef]
13. Jiang, D.; Xu, Z.; Lv, Z. A multicast delivery approach with minimum energy consumption for wireless multi-hop networks. *Telecommun. Syst.* **2016**, *62*, 771–782. [CrossRef]

14. He, Z.; Cai, Z.; Cheng, S.; Wang, X. Approximate Aggregation for Tracking Quantiles and Range Countings in Wireless Sensor Networks. *Theor. Comput. Sci.* **2015**, *607*, 381–390. [CrossRef]
15. Barbounis, T.G.; Theocharis, J.B. A locally recurrent fuzzy neural network with application to the wind speed prediction using spatial correlation. *Neurocomputing* **2007**, *70*, 1525–1542. [CrossRef]
16. Cheng, S.; Cai, Z.; Li, J. Curve Query Processing in Wireless Sensor Networks. *IEEE Trans. Veh. Technol.* **2015**, *64*, 5198–5209. [CrossRef]
17. Sasikala, S.; Balamurugan, S.; Geetha, S. A Novel Memetic Algorithm for Discovering Knowledge in Binary and Multi Class Predictions Based on Support Vector Machine. *Appl. Soft Comput.* **2016**, *49*, 407–422.
18. Malvoni, M.; De Giorgi, M.G.; Congedo, P.M. Photovoltaic Forecast Based on Hybrid PCA–LSSVM Using Dimensionality Reduced Data. *Neurocomputing* **2016**, *211*, 72–83. [CrossRef]
19. Jiang, D.; Xu, Z.; Liu, J.; Zhao, W. An optimization-based robust routing algorithm to energy-efficient networks for cloud computing. *Telecommun. Syst.* **2016**, *63*, 89–98. [CrossRef]
20. Jiang, D.; Zhang, P.; Lv, Z.; Song, H. Energy-efficient multi-constraint routing algorithm with load balancing for smart city applications. *IEEE Intern. Things J.* **2016**, *3*, 1437–1447. [CrossRef]
21. Jiang, D.; Huo, L.; Lv, Z.; Song, H.; Qin, W. A joint multi-criteria utility-based network selection approach for vehicle-to-infrastructure networking. *IEEE Trans. Intell. Transp. Syst.* **2017**. [CrossRef]
22. Zhong, Y.; Ning, J.; Zhang, H. Multi-agent Simulated Annealing Algorithm based on Particle Swarm Optimisation Algorithm. *Int. J. Comput. Appl. Technol.* **2012**, *43*, 335–342. [CrossRef]
23. Zainal, N.; Zain, A.; Radzi, N. Glowworm Swarm Optimization (GSO) for optimization of machining parameters. *J. Intell. Manuf.* **2016**, *27*, 998–1006. [CrossRef]
24. Lentka, Ł.; Smulko, J.M.; Ionescu, R.; Granqvist, C.G.; Kish, L.B. Determination of gas mixture components using fluctuation enhanced sensing and the LS-SVM regression algorithm. *Metrol. Meas. Syst.* **2015**, *22*, 341–350. [CrossRef]
25. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the First IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; pp. 1942–1948.
26. Li, J.; Cheng, S.; Cai, Z.; Yu, J.; Wang, C.; Li, Y. Approximate Holistic Aggregation in Wireless Sensor Networks. *ACM Trans. Sens. Netw.* **2017**, *13*, 11:1–11:24. [CrossRef]
27. Jiang, M.; Luo, J.; Jiang, D.; Xiong, J.; Song, H.; Shen, J. A Cuckoo Search-support Vector Machine Model for Predicting Dynamic Measurement Errors of Sensors. *IEEE Access* **2016** , *4*, 5030–5037. [CrossRef]
28. Krishnanand, K.N.; Ghose, D. Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions. *Swarm Intell.* **2009**, *3*, 87–124. [CrossRef]