Journal of Network Intelligence
Taiwan Ubiquitous Information

©2018 ISSN 2414-8105 (Online)
Volume **3**, Number **4**, December **2018**

Optimization of Multi-scale Kernel Chaotic Time Series Prediction Method based on The Joint Parameters Were Optimized with Variable Particle Swarm

Baoxiang Du, Junhua Zhu, Qun Ding
Electronic Engineering College of Heilongjiang University
No. 74, Xuefu Road, Nangang District, Harbin, Heilongjiang,China
dubaoxiang@sina.com

Received January 2018; revised August 2018

Abstract. *Support vector regression (SVR) algorithm is widely used in chaotic time series prediction because of its good model sparse performance, but at present, most of the research work on SVR algorithm is focused on construction of mononuclear functions and optimization of parameters, however, in practical application, due to the complicated laws of various data, it is difficult to use a single kernel function to reflect the laws that have both steep and gentle changes. In addition, only limited measured data can be obtained in practice, it is difficult to use a single support vector regression to accurately establish a model with complex change patterns under a small number of measured data samples. In this paper, a hybrid kernel function based on linear weighting of multiple kernel functions is proposed to construct the multi-kernel support vector regression algorithm, and the parameters are optimized by the joint parameters are optimized by multi-scale escape particle swarm. Experimental results show that the proposed method improves the accuracy of chaotic prediction and has good generalization ability.*

Keywords: Chaotic time series; Multi-kernel support vector machine; Multi-scale escaping particle swarm

1. Introduction.

Chaos is a random movement in a deterministic system. The discrete situation of chaos is usually expressed as chaotic time series, chaotic time series is a time series which is based on chaotic model and has chaotic characteristics, chaotic time series contains rich dynamics information of chaotic system, how to find the dynamical characteristics of the chaotic system generated by the finite chaotic time series observations is a subject that scholars have been paying close attention to in time series analysis [1,2]. Support vector regression is a machine learning algorithm proposed by the scholar Vapnik on the basis of statistical learning theory, It uses two-time programming algorithms to solve, also known as quadratic programming support vector regression (QPSVR) [3,4], It solves the shortcoming of traditional learning theory such as neural network and is widely used in various functions and series prediction. The scholar Smola then propose a linear programming support vector regression (LPSVR) with linear programming algorithm [5,6], the research results show that LPSVR has better model sparsity than QPSVR and can use more general kernel functions. The above analysis shows that the SVR algorithm has been widely used in the prediction of chaotic sequences, however, the current research work on SVR algorithm mainly focuses on the construction of single kernel function and optimization of parameters. For the relatively complex distribution of data, the performance of the single kernel support vector regression algorithm has a great relationship with the kernel function and its corresponding parameter selection, the general Cross-validation method for determining parameters is time-consuming and arbitrary, so the number of predictive precision and support vectors is very susceptible. In practical applications, because of the complexity of the hidden laws of various data, it is difficult to use a single kernel function to reflect the law with both steep and gentle changes.

In addition, only limited measured data are usually obtained in practice, it is difficult to use single support vector regression to accurately establish a model with complex variation under a small number of measured data samples. Therefore, the problem of how to model accurately need to solve when only a small number of actual data samples in practical application [7].

In recent years, multi-kernel learning is a hotspot in the field of nuclear machine learning. It is a very flexible type in the current nuclear learning model. It has been proved that the multi-kernel substitution of single kernel can enhance the performance of decision function, and can obtain better performance than single core model or single core machine combination model. The hybrid kernel obtained through multi kernel learning can be more fully expressed in the feature space, and can reduce the number of support vectors and improve the prediction accuracy. It has been widely used in classification [8], regression [9], clustering [10], dimensionality reduction [11]. and it has excellent performance in image classification [12], pose estimation [13], visual tracking [14], face recognition [15] and so on. Literature 8-15 reviews the progress in the research of multi-core methods. Support vector regression algorithm can get functions from data learning and show better performance than other methods. In order to solve the problem of accurate modeling of complex laws, it is necessary to use linear weighting of multiple kernel functions to form a mixed kernel function to construct multi kernel support vector regression (SVR) algorithm.

2. **Multi-scale kernel method.**

The most classical method of constructing multi-core learning is to combine multiple admissible nuclei into linear convex combinations, it can be expressed as

$$k(x_i, x_j) = \sum_{l=1}^{M} \beta_l k_l(x_i, x_j) \tag{1}$$

Where $\beta_l > 0$; l=1,...,M; $\sum_{l=1}^{M} \beta_i = 1$

This method was first used by Pavlidis for the classification of genetic functions of heterogeneous data, due to the different characteristics of genetic data from heterogeneous sources, different nuclear matrix integration is needed to evaluate the contribution of different heterogeneous characteristics, this kind of multi-kernel method is by the linear weighted combination of multiple kernel functions. The framework of the proposed method is illustrated in figure.1.

As is shown in Fig.1, the multi-kernel learning method is similar to a neural network, and output is composed of linear combinations of several nodes in the middle, the decision function of multi-core learning is similar to a neural network with a higher level than the support vector machine, whose output is a linear combination of the output of the kernel function in the middle.

The training methods of multi-kernel learning are mainly divided into two kinds:

(1) One-step learning method. The method is to train and study the whole of the weight of kernel function and the sample weight of synthetic learning machine, which is divided into serial training learning method and parallel training learning method. In the serial

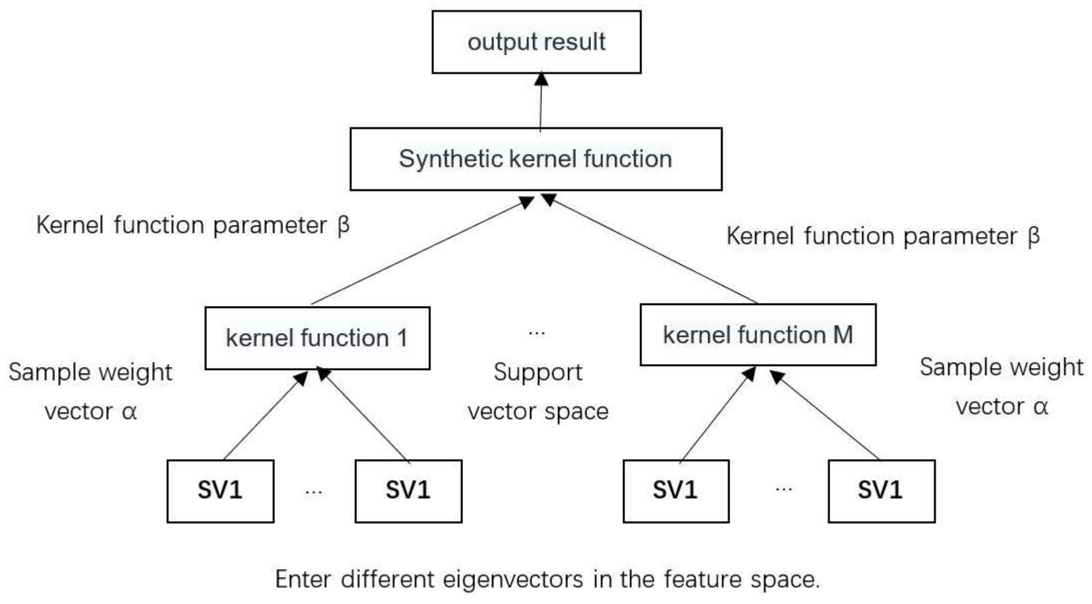

Figure 1. Schematic diagram of multi-core method for linear weighted synthesis

training learning method, the weight of kernel function is determined firstly, and then the parameters are obtained by using the kernel learning machine of the weighted kernel function; In parallel training learning methods, both of these parameters are trained at the same time.

(2) Two step learning method, also called iterative methods. In each iteration, the first fixed the parameters of the basic nuclear learning machine and then updated the weights of the synthesized nuclei, finally fixed the weights of the synthesized cores and updated the parameters of the basic nuclear learning machine with the basic training algorithm.

The intelligent optimization problem of multi-kernel learning method is to find some intelligent algorithms which are more mature and have good performance. First, set up an objective function and then find the extremum of the function, finding the extremum of a function is actually the process of synthesizing a kernel function, for example, the kernel function of polynomial and radial basis kernel can be selected as the kernel function of support vector machine, and support vector machine (SVM) is used for prediction. The order of the polynomial kernel, the scaling parameters of the radial base nucleus, the SVM adjusting parameters and the two weighting parameters of the synthesized kernel are used to form the parameter vectors as the particles, the parameters of the synthesized kernel are optimized by particle swarm algorithm, and the optimal prediction result is finally found.

3. **The joint parameter optimization algorithm based on multi-scale escape particle swarm.** Particle swarm optimization (PSO) was first proposed by Eberhart and Dr Kennedy in 1995, The basic idea comes from their early discovery of the simulation and modeling of the group behavior of birds, using social behavior instead of natural selection mechanism of evolutionary algorithm, and realizing the optimal solution search by mutual cooperation among populations [16-18]. Because the PSO algorithm is relatively simple, requires less parameters and can effectively solve the complex optimization tasks, it has good performance in pattern recognition, multi-objective function optimization, neural network weights training and image processing. However, as a new swarm intelligence

algorithm, PSO algorithm still has two difficult problems, such as slow convergence rate and premature (falling into local optimal point) [19-20]. Therefore, in order to prevent the PSO algorithm from falling into the local optimal solution, the researcher improves the global solution search ability by improving the population diversity. In the literature [21], the random multigenerational initialization of the population is proposed to solve the aggregation and conflict between particles to enhance the diversity of the population. Literature [22] improved the diversity of population and enhanced algorithm searching ability by introducing small probability random mutation mechanism. But due to the mutation rate is not easy to control, an excessive mutation rate, while increasing the diversity of the population, will also lead to mass chaos, this will delay the precise local search and delay the convergence rate of the algorithm.

Accordingly, a multi variant escape particle swarm optimization algorithm is applied in this chapter [23], the escaping ability of the algorithm includes uniform mutation operator and Gaussian mutation operator with different scale variance. Algorithm which USES the uniform mutation operator can guarantee whenever it has ability to escape local optimal solution, so as to enlarge the search of solution space, especially at the beginning of the algorithm, the mutation can make the algorithm has strong ability to space exploration; In order to make up for the deficiency of the uniform mutation operator cannot do local detailed search,, the Gaussian mutation operator with different scales is adopted, the operator can help the algorithm to conduct a distributed search in the search space, at the same time, the variation scale decreases with the increase of the adaptability, which can improve the accuracy of the optimal solution while guaranteeing the escaping ability, and ensure the convergence performance of the algorithm, especially in the later stage of the algorithm, which has strong local mining ability.

The algorithm is described as follows:

Setting the number of particles n, the number of Gaussian mutation operator scale is M, first sets the initial variance of multi-scale Gaussian mutation operator:

$$\sigma^{(0)} = (\sigma_1^{(0)}, \sigma_2^{(0)}, ..., \sigma_M^{(0),}) \tag{2}$$

At the beginning, the Gaussian variance is generally set to the range of the optimal variable, with the increase of the number of iterations, the variance of the multi-scale Gaussian mutation operator is adjusted. the specific adjustment method is as follows: firstly, according to the size of the adaptive value, the particles in the population are sorted from small to large, and then combine it to form M subgroup, the number of particles per subgroup is $P = N/M$, K is the current iteration number, the fitness of each subgroup is calculated:

$$FitX_m^{(k)} = \sum_{i=1}^{p} Fit(x_i^m)/p \qquad m = 1, 2, ..., M \tag{3}$$

Each subgroup obtains different mutation abilities according to the different adaptive values, therefore, the standard deviation of the first m mutation operator is adjusted according to the variation of the adaptive value of the first m subgroup of the current iteration number k:

$$\sigma_m^{(k)} = \sigma_m^{(k-1)} exp(\frac{M * FitX_m^{(k)} - \sum_{m=1}^{M} FitX_m^{(k)}}{FitX_{max} - FitX_{min}}) \tag{4}$$

$$FitX_{max} = max(FitX_1^{(k)}, FitX_2^{(k)}, ..., FitX_M^{(k)}) \tag{5}$$

$$FitX_{min} = min(FitX_1^{(k)}, FitX_2^{(k)}, ..., FitX_M^{(k)}) \tag{6}$$

Due to the evolution of standard deviation of Gauss mutation operator is a recursive process, the Gaussian mutation operator in the back may be very large, therefore, the standard deviation of the Gaussian mutation operator is as follows: if $\sigma_i^{(k)} > W/4$,

$$\sigma_i^{(k)} = |\frac{W}{4} - \sigma_i^{(k)}| \tag{7}$$

W is the width of the variable space to be optimized, reuse the upper form until $\sigma_i^{(k)} < W/4$ is satisfied.

4. **Optimization of chaotic prediction with multi-kernel support vector machine , multi-scale escape PSO and Joint parameter.** If a time series with chaotic characteristics is considered to be produced by a nonlinear dynamical system, phase space reconstruction is the method of restoring and characterizing the original nonlinear dynamical system with this time series. Its basic idea is that the evolution of any component in the system is determined by the other components interacting with it. Therefore, the information of these relevant components is hidden in the development of any component. So in order to reconstruct an equivalent state space, just to look at one component, and the measurement of it at certain fixed time delay points is used as a new dimension, they determine a point in a multidimensional state space. Suppose that the time sequence of a single component produced by a chaotic system is $x(1), x(2), ..., x(n)$, so according to the Takens embedding theorem, By determining the appropriate time delay interval τ, and embedding dimension m, the chaotic time series is reconstructed to $X(n) = x(n), x(n + \tau), ..., x(n + (m - 1)\tau)$.

it is not difficult to find that the advantages and disadvantages of phase space reconstruction depend on the time delay interval and the quality of embedding dimension.

It is known from the above analysis, for SVM algorithm, the linear weighted weight, the penalty parameter and the parameters of gaussian kernel have great influence on the prediction performance. Moreover, there is a certain correlation between these parameters, which can not be optimized separately. For the time series prediction algorithm based on phase space, the time delay interval and the embedding dimension have great influence on the prediction result. However, the general method of chaotic time prediction based on phase space SVM usually separates the parameters of phase space reconstruction from the parameters of SVM algorithm to Optimize, this will not ensure that the parameters are optimal at the same time. In order to obtain the optimal chaotic time series prediction results, we need to combine the above parameters to make it optimal at the same time.

In order to optimize the above parameters at the same time, these variables need to be coded first. The real-coded method is used here. The particle expression of PSO algorithm is as follows:

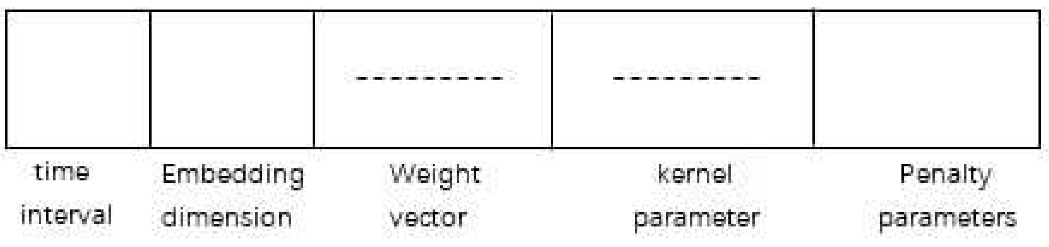

FIGURE 2. encoding of particles

Determination of individual fitness

Due to the goal of the final search based on Multiscale escape PSO is to get the optimal joint vectors, improve the accuracy of the chaotic time prediction of SVM algorithm and reduce the prediction error. Therefore, the individual fitness setting is related to the prediction ability of the final model. In this paper, the mean value of the predicted regularized mean square difference(NRMSE) is obtained by 5 cross validation:

$$\begin{cases} RMSE_i(T_i) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \\ NRMSE_i(T_i) = \frac{MSE_i(T_i)}{\sigma} \\ NRMSE = \frac{1}{k}\sum_{i=1}^{k} NRMSE_i(T_i) \end{cases} \tag{8}$$

σ is the standard deviation of the predicted chaotic time series.

Five cross-validation methods first divided the training sample set into five training sample subsets. the number of training samples for each subset is equal. Next, the five subsets are divided into four subset of training samples and a subset of evaluation samples, each time the four training samples were used to train SVM algorithm. Then the average square root difference of regularization is predicted and calculated by using the set of evaluation samples. Finally, the average value of the Prediction of regularization mean square root difference of the predicted five times is calculated as an individual adaptive value.

A combined optimization step of multi-kernel chaotic parameters based on multi-scale escape PSO:

(1). First, initialize the variable and set the initial value of the inertial weightW_{max},change step-size W_v, final inertia weight W_{min}, This setting W is linearly reduced with iteration in order to improve the convergence ability of the algorithm. The number of population $SwarmSize$, Scope of the optimization variable, the Learning factors of cognitive part (representing the particles ' learning of themselves) and social part (representing the collaboration of particles) C_1, C_2, K is the current iteration number, K_{max} is the maximum number of iterations of the algorithm, the initial thresholds for various dimensional velocities T_d, F_d records the number of times each dimension is less than the threshold. $K1$ is the threshold which Speed less than threshold time, $K2$ is the adjustment step-size of threshold T_d. Initializing the variance of multi-scale Gaussian mutation operators;

(2). Firstly, the particles are encoded and the particle population is initialized according to the range of the optimization variables, and the adaptive degree of the particles is calculated;

(3). Compare each particle fitness value with the fitness value of each particle experienced the best position of $pBEST$ and group experience best position $gBEST$;

(4). Enter the loop and use the linear formula to set the inertia weight W_i;

(5). The velocity and position of particle are updated according to the formula of PSO algorithm, and the legal range of various optimization variables is checked.

(6). Compared with the corresponding adaptive value of the particle position before and after the update, the particle position updating formula of PSO algorithm is used to produce better particles.

(7). Determine whether the velocity of each dimension meets the threshold T_d, the multi-scale Gaussian variation and uniform variation are done, and the adaptive values of various modified particles are computed, and the particle adaptation values before and after the mutation are compared, and the optimal particles are updated;

(8). Updating variance of Multi-scale Gaussian mutation operator, and record The Times of meeting the speed threshold F_d, and determine whether F_d satisfies $K1$, And update T_d;

(9). If the termination condition (up to the maximum number of iterations) is reached, the operation is closed, otherwise the Loop 4 operation.

In order to make use of multi-scale kernel SVM algorithm to predict chaotic time series and improve prediction precision, we need to optimize the joint vectors of time interval and embedding dimension, weight, kernel parameter and penalty parameter, therefore, the Multi-scale mutation PSO algorithm is used to optimize it. Where the initialization range of the union vector is [110][110][01][01][01][01][02][24][48][816][501000]. T_d is set to $[220.001, 0.001, 0.001, 0.0010.010.010.020.022]$. Because the time interval and the embedded dimension are integers, it is meaningless to set them to a smaller size, so the speed thresholds for these two dimensions are set to be large, in order to prevent the mutation from being too frequent, the speed of other dimensions should be smaller, this can effectively adjust the algorithm exploration and mining capabilities.

The flow chart of the algorithm is shown in figure.3.

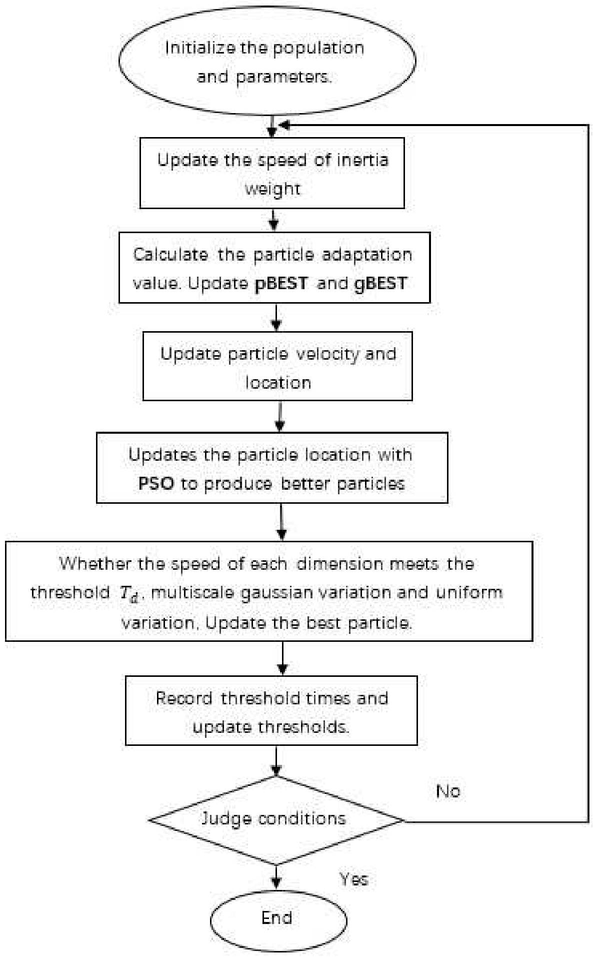
Initialize the population and parameters.
Update the speed of inertia weight
Calculate the particle adaptation value. Update pBEST and gBEST
Update particle velocity and location
Updates the particle location with PSO to produce better particles
Whether the speed of each dimension meets the threshold T_d, multiscale gaussian variation and uniform variation. Update the best particle.
Record threshold times and update thresholds.
Judge conditions
No
Yes
End

FIGURE 3. Optimization flow chart base on multi-scale variant PSO and Joint parameter

5. Analysis and comparison of testing.

5.1. Lorenz chaotic time series prediction.

In order to verify the predictive performance of this algorithm, we first use the Lorenz attractor, one of the most famous

3-dimensional autonomous nonlinear dynamical systems, It is mainly used for fluid turbulence modeling. Its dynamic equation model is described below:

$$\begin{cases} \frac{dx(t)}{dt} = -\sigma x(t) + \sigma y(t) \\ \frac{dy(t)}{dt} = rx(t) - y(t) - x(t)z(t) \\ \frac{dz(t)}{dt} = x(t)y(t) - bz(t)/3 \end{cases} \tag{9}$$

The parameters are set to $(\sigma, r, b) = (10, 28, 8)$, at this time the system presents a chaotic state. We first make a single step prediction of the chaotic time series of one of the state variables constituting the system, the state variable is $x(t)$. This sets the system's initial state variable X, Y, Z to $[1, 2, 3]$, Using the four-order five-level Runge-Kutta method to solve the problem, set step-size to $[0, 0.01, 100]$, getting a numerical solution of a time series $x(n)$. Then use the Multi-scale escape PSO algorithm to optimize these joint parameter vectors. The multi-scale Gaussian mutation SVM algorithm of predictive model adopts 4 RBF kernel, the parameters are set as follows: the number of population is 10, the number of iterations is 500,$C_1 = 1.4, C_2 = 1.4, W_start = 0.95, W_end = 0.4, w_vary = 0.15; K1 = 5, K2 = 10$. Basic PSO algorithm parameter ditto. The change of the optimal individual adaptive value and the number of iterations is as follows, and compare with the basic PSO algorithm. The results are shown in figure.4, figure.5 and figure.6.

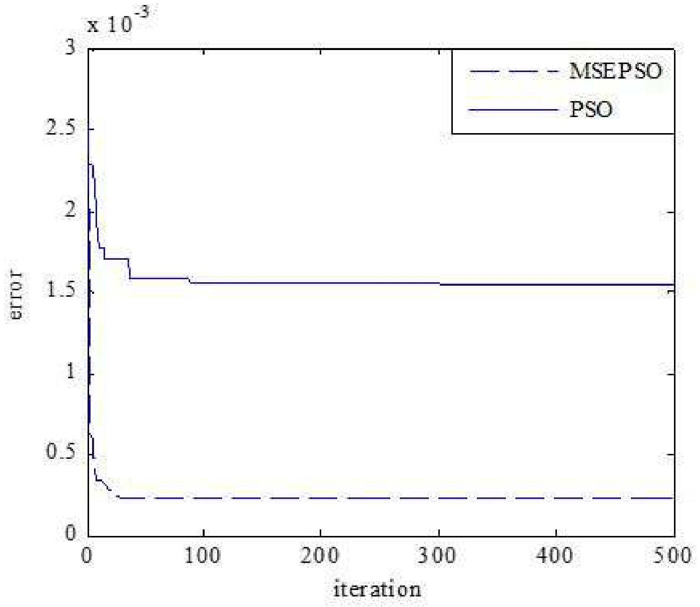

Figure 4. Comparison diagram of our algorithm and PSO algorithm in the optimal joint vector-valued contrast of Lorenz chaotic time series $x(n)$

It can be seen from the above figure that the basic PSO algorithm can easily fall into the local optimal solution, as the number of iterations increases, it is still impossible to escape the local optimal solution. But in this paper, we can quickly escape these local extremum points by using multi-scale mutation to realize the accurate location of the optimal solution region, the algorithm can find the correct search direction in a short time, and has a fast descent speed. The search for the optimal value can be achieved with few iterations.

The optimal vectors obtained by the two algorithms are shown in table 1.

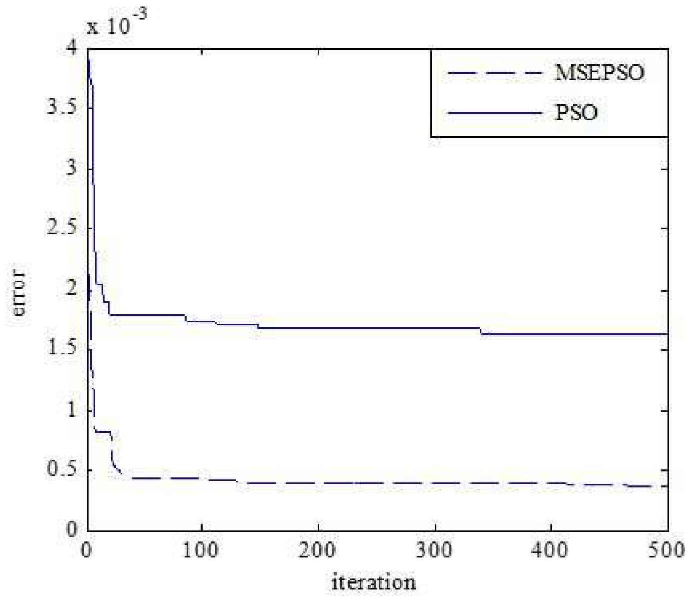

FIGURE 5. Comparison diagram of our algorithm and PSO algorithm in the optimal joint vector-valued contrast of Lorenz chaotic time series $y(n)$

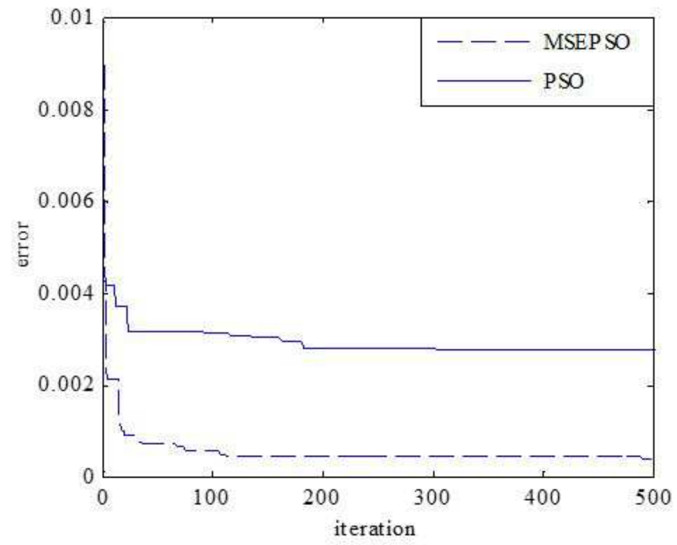

FIGURE 6. Comparison diagram of our algorithm and PSO algorithm in the optimal joint vector-valued contrast of Lorenz chaotic time series $z(n)$

In order to facilitate the comparison of the prediction results of various algorithms, set phase space $m = 2, \tau = 2$. In order to eliminate the transient effects, we start with the 5001 time series. We select 500 of the training samples, that is [5005,5504], the test sample is 1500 sequence values. The least Squares support vector machine (LS-SVM) parameter is set as follows in this algorithm. The kernel function is the RBF Gaussian kernel, the nuclear width is $\sigma =\sqrt{3}$, Penalty factor $\gamma = 8500$. For performance comparisons, this paper adopts the current popular algorithm gaussian model GP. least squares support vector machines (LS-SVM), non-sensitive support vector machines ε, Support vector machine ν, and RBF neural network algorithm. Three support vector

TABLE 1. The optimal joint vector-valued contrast of Lorenz series

variable	algorithm	interval	dimension -ality	weight	kernel parameters	penalty value
x(n)	PSO	2	3	[0.340.210.085 0.36]	[6.6 8.2 7.3 3]	10
	Algorithm in this paper	2	2	[0.01 0.3 0.3 0.39]	[1.9 3.8 7 10]	949
y(n)	PSO	2	2	[0.0037 0.33 0.21 0.46]	[1.8 3.7 4.9 9]	10
	Algorithm in this paper	2	2	[0.21 0.3 0.23 0.27]	[4.3 7.8 3.2 6.1]	913
z(n)	PSO	2	2	[0.24 0.28 0.31 0.17]	[3.9 9.3 6.4 4.9]	10
	Algorithm in this paper	2	2	[0.0049 0.18 0.28 0.53]	[1.9 3.5 6.8 12]	873

machine algorithms use all gaussian kernel, and the kernel parameters $\varepsilon - SVR$ is$\sqrt{2.2}$, Penalty factor $C = 3$,$\varepsilon = 0.01$, the $v = 0.5$ of the $\varepsilon - SVR$ algorithm, the other is the same as the $\varepsilon - SVR$ algorithm, the least squares support vector machine is the same as the algorithm set in this paper. RBF neural network employs 150 hidden-layer centers, the GP model uses a conventional Gaussian kernel function.

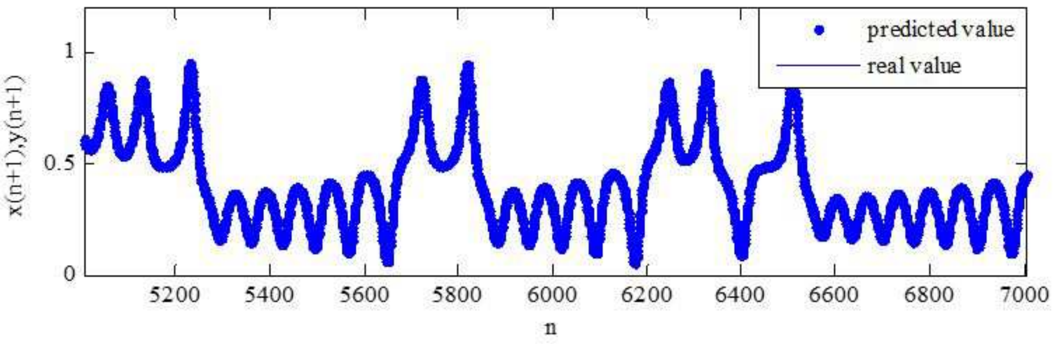

FIGURE 7. n=5005 to 7004 Lorenz chaos time series real output(solid-line) and single-step prediction output(dot-line)

figure.7 is a comparison between the predicted value and the real value of the chaotic time series of the Lorenz sequence by using algorithm of this paper, in order to facilitate display, reduce the range of values to [5005,7004] . The absolute prediction error distribution of each point is shown in figure.8. It is not difficult to find that, whether it is training data or test data, their single step prediction results and real values are well matched.

Table 2 shows the comparison between the methods in this paper and other chaotic time series prediction methods, from the results of comparison, we can see that this method shows successful prediction and robustness in training set and test set. This is because multi-scale kernel functions are more flexible than single kernel functions, large-scale kernel function can be adapted to smooth region of fitting decision function. The

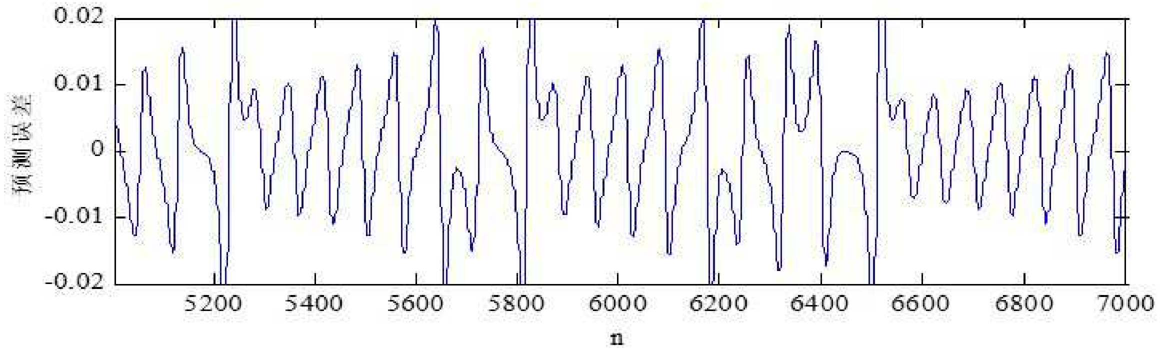

FIGURE 8. n=5005 to 7004 Lorenz chaos time series absolute prediction error distribution

TABLE 2. Performance comparison of Lorenz chaos time series $x(n)$ prediction with other methods

Prediction modle	Training RMSE	Training NRMSE	Prediction RMSE	Prediction NRMSE
GD	0.00706	10157e-04	0.0345	5.659e-04
$\varepsilon - SVR$	0.00702	10149e-04	0.07223	0.001182
$\nu - SVR$	0.03230	5.288e-04	0.1265	0.002072
LS-SVM	0.01164	1.9068e-04	0.05638	9.2307e-04
RBF-NN	0.02545	4.1655e-04	0.07332	0.00120
PSO-SVM	0.03297	5.3983e-04	0.06986	0.001143
MSEPSO-LSSVM	0.00679	1.1130e-04	0.0233	3.8217e-04

small-scale kernel function is more suitable for smoothing the region with more drastic change of the decision function. The method of single kernel function can only use one scale to fit the smooth part of the decision function and change the violent part. If the parameters of the single kernel function are set too high, the large-scale kernel is apt to cause the lack of fitting to the decision function. And the setting is too low, it is easy to have the phenomenon of over fitting. Multi-scale kernel can be better to neutralize the disadvantages of too large scale and too small scale of single nucleus, so it is better to solve the problem of fitting and over fitting, and it is not easy to deviate.

5.2. **Prediction of chaotic time series of sunspot.** The sunspot number is an exponent used to indicate the level of the sun's total activity. Previous studies have shown that the monthly average and annual mean of sunspot are a low-dimensional chaotic time series. This paper uses the data sequence of sunspot data from NASA: http://solarscience.msfc.nasa.gov/greenwch/spot_num.txt. The chaotic time series prediction of the monthly average sunspot number from 1749-2014 is carried out. Take the first 200 time series values as training samples and the last 1000 as test samples. The embedding dimension of the reconstructed phase space obtained after optimization of the multi-scale escape PSO optimization algorithm in this paper is $M = 1$, and the time delay is $\tau = 3$, Here take $q = 3$ to generate training samples and test sample sets. figure.9 is the iterative optimal joint vector result comparison chart of the Multi-scale escape PSO algorithm and the basic PSO optimization algorithm, Table 3 compares the results of the combined vector optimization with two different optimization algorithms.

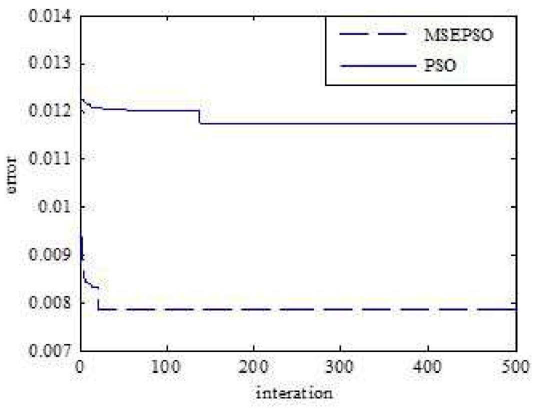

FIGURE 9. Comparison diagram of our algorithm and PSO algorithm in the optimal joint vector-valued contrast of sunspot number sequences 1-step prediction

TABLE 3. The optimal vector-valued of sunspot number sequences by different optimal algorithm

algorithm	interval	dimension -ality	weight	kernel parameters	penalty value
PSO	5	4	[0.79 0.092 0.0019 0.12]	[4.5 5.7 6 8.3]	10
Algorithm in this paper	3	1	[0.32 0.29 0.0062 0.38]	[1.2 3.1 6.1 15]	407

For ease of display, only the predicted results of the 500 test samples in the range [503 1002] are shown here. figure.10 is the fitting relation between the single step prediction result and the actual output value of the sunspot average number in this algorithm, figure.11 shows the distribution of absolute errors. It can be seen from the graph that the algorithm has a good coincidence effect on the single step prediction of the monthly average number of sunspots. Table 4 compares the predictive performance of different prediction algorithms to the monthly average chaotic sequence of sunspots, It's not hard to see the optimal predictive performance of this algorithm. PSO-LSSVM optimization algorithm is easy to get into local optimal solution because of PSO algorithm own defects, So the joint vectors obtained after optimization may not be the optimal solution, It is very important for the optimization ability of the optimization algorithm in the solution model. This is also the key joint parameter vector optimization algorithm can improve the predictive performance of multi-kernel SVM algorithm.

6. **conclusion.** In this paper, a chaotic time series prediction method based on Multi-scale kernel is proposed for the disadvantage of the single kernel SVM chaotic prediction model. This method is a combined vector optimization method based on PSO intelligent optimization algorithm, phase space parameter, kernel parameter vector and penalty function, it makes up for the shortcoming of previous experience selection and step optimization parameters, the joint optimization vector obtained by this method can make the predictive model get better prediction precision, and improve the expression ability and stability of the decision function.

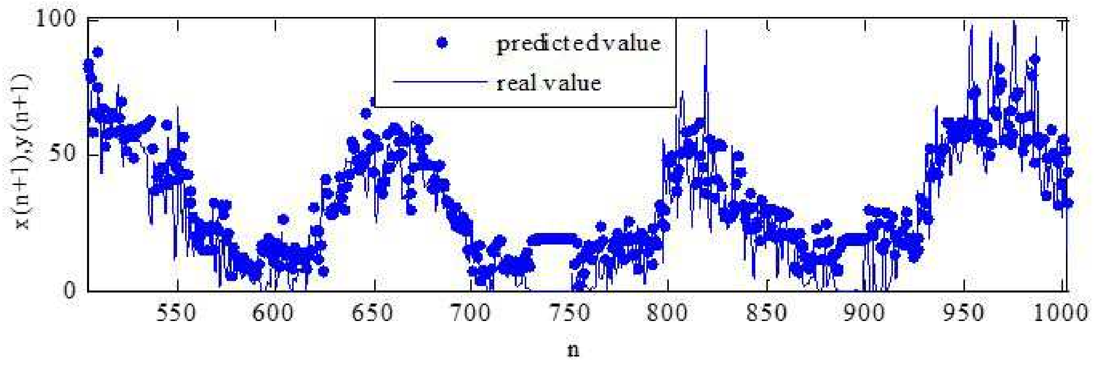

FIGURE 10. n=503 to 1002 Sun Spot chaos time series real output(solid-line) and single-step prediction output(dot-line)

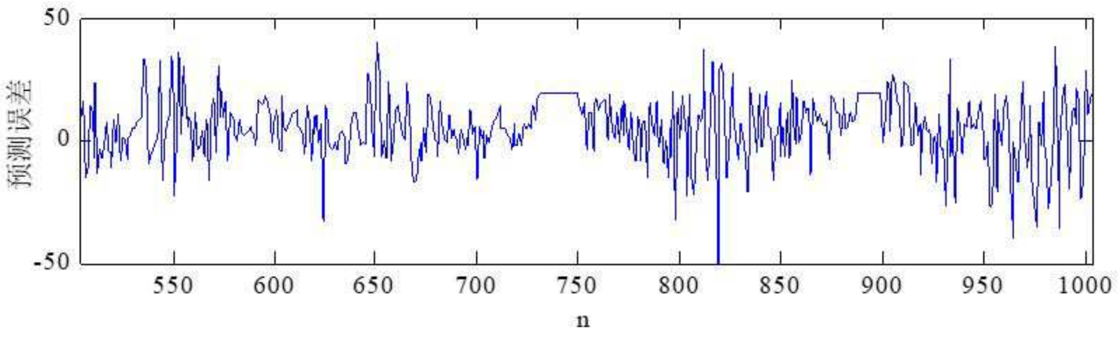

FIGURE 11. n=503 to 1002 sun spot chaos time series prediction error distribution

TABLE 4. Performance comparison of Lorenz chaos time series $x(n)$ prediction with other methods

Prediction modle	Training RMSE	Training NRMSE	Prediction RMSE	Prediction NRMSE
GD	0.1207	6.201e-05	35.57	0.01827
$\varepsilon - SVR$	22.209	0.01141	44.043	0.02263
$\nu - SVR$	24.4433	0.01256	43.357	0.02228
LS-SVM	0.1251	6.4317e-05	43.4456	0.02233
RBF-NN	0.367	1.886e-04	43.897	0.02255
PSO-SVM	2.23358	0.001148	43.4436	0.02232
MSEPSO-LSSVM	0.1110	5.7056e-05	32.0223	0.01645

Acknowledgment. This work is supported by Science and technology research project of Heilongjiang Provincial Education Department (NO.12541637).

REFERENCES

[1] H. L. Zhang, R. G. Li, W. H. Fan and Y. Wang, Full-parameter continuous fractional chaotic time series prediction based on quantum particle swarm *Control and decision*, vol. 31, no. 1, pp. 52-58,China, 2016.
[2] M. L. Xu,M. Hang, A predictive model of factor echo State network for multivariate chaotic time series, *Journal of automation*, vol. 41, no. 5, pp. 1042-1046, 2015.

[3] Vapnik V N, The Nature of Statistical Learning Theory, *Berlin:Springer- Verlag*, 1995.

[4] X. G. Zhang, About statistical learning theory and support vector machine, *Journal of automation*, vol. 26,no. 1 pp. 32-42, China, 2000.

[5] A. J. Smola,B. Scholkopf, A tutorial on support vector regression, *Statistics and Computing*, vol. 14,no. 3 pp. 199-222, 2004.

[6] A. Smola, B. Scholkopf and G. Ratsch, Linear programs for automatic accuracy control in regression, *Proceedings of the 9th International Conference on Artificial Neural Networks. Edinburgh, UK: IEEE*, pp. 575-580, 1999.

[7] G. R. G. Lanckriet, N. Cristianini, P. Bartlett,L. E. Ghaoui and M. I. Jordan, Learning the kernel matrix with semidefinite programming, *Journal of Machine Learning Research*, vol. 11,no. 5 pp. 27-72, 2004.

[8] C. V. Gustavo, G. C. Luis, M. M. Jordi, V. F. Joan V F and C. M. Javier, Composite kernels for hyperspectral image classification, *IEEE Transactions on Geoscience and Remote Sensing Letters*, vol. 3,no. 1 pp. 93-97, 2006.

[9] D. N. Zheng, J. X. Wang, Y. N. Zhao, Non-flat function estimation with a multi-scale support vector regession, *Neurocomputing*, vol. 7,no. 1-3 pp. 420-429, 2006.

[10] A. Nazarpour,P. Adibi, Two-stage multiple kernel learning for supervised dimensionality reduction, *Pattern Oecognition*, vol. 48,no. 5 pp. 1854-1862, 2015.

[11] D. Sahoo,S. C. H. Hoi,B. Li, Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 293-302, 2014.

[12] J. H. Yu, F. Sun, W. Y. Li, 3D human pose estimation based on multi-kernel sparse coding, *Acta Electronica Sinica*, vol. 44,no. 8 pp. 1899-1908, 2016

[13] J. SU,B. H. Wang,D. Z. Liu, A multi-kernel tracking algorithm based on topology constraint, *Proc of IEEE Int Conf on Neural Networks. Piscataway,NJ:IEEE Press*, vol. 43,no. 2 pp. 353-357, 2015

[14] S. Yang,C. X. Zhao,F. Liu, Fusion of local and global features using multiple kernel learning for face recognition , *Acta Electronica Sinica*, vol. 44,no. 10 pp. 2344-2350, 2016

[15] G. Mehment,A. Ethem, Particle swarm optimization, *Proc of IEEE Int Conf on Neural Networks. Piscataway,NJ:IEEE Press*, vol. 12 pp. 2211-2268, 2011

[16] J. Kennedy, R. Eberhart, Particle swarm optimization, *Proc of IEEE Int Conf on Neural Networks. Piscataway,NJ:IEEE Press*, 1995.

[17] Y. H. Shi, R. Eberhzrt, A modified particle swarm optimizer, *Proc of IEEE Int Conf on Evolutionary Computation. Piscataway,NJ,IEEE Press*, pp. 67-73, 1998.

[18] K. E. Parsopoulos,M. N. Vrahatis, On the computation of all global minimizers through particle swarm optimization, *IEEE Trans. on Evolutionary Computation*, vol. 8,no. 3 pp. 211-224, 2004.

[19] I. C. Trelea, The particle swarm optimization algorithm:convergence analysis and parameter selection, *Information Processing Letters*, vol. 85,no. 9 pp. 317-325, 2003.

[20] R. Mendes,J. Kennedy and J. Neves, The Fully Informed Particle Swarm:Simpler, Maybe Better, *IEEE Trans on Evolut ionary Computation*, vol. 8,no. 6 pp. 204-210, 2004.

[21] F. Bergh,T. A. P. Engelbrech, A cooperative approach to particle swarm optimization, *IEEE Trans on Evolutionary Computation*, vol. 8,no. 3 pp. 1-15, 2004.

[22] X. F. Xie, W. J. Zhang,Z. L. Yang, A dissipative particle swarm optimization, *Proceeding of the IEEE International Conference on Evolutionary Computation, Honolulu,IEEE*, pp. 1456-1461, 2002.

[23] X. M. Tao, F. R. Liu, Y. Liu, Z. J. Tong, A particle swarm optimization algorithm for multi-scale collaborative variation, *Journal of software*, vol. 23,no. 7 pp. 1805-1815, 2012.