Parameters Selection of SVM for Function Approximation Based on Differential Evolution

Shaowu Zhou [1] Lianghong Wu [1,2] Xiaofang Yuan [2] Wen Tan [1]

[1] College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, P. R. China

[2] College of Electrical and Information Engineering, Hunan University, Changsha 410082, P. R. China

Abstract

Support vector machines (SVM) is a new machine learning method, and it has the ability to approximate nonlinear functions with arbitrary accuracy. Right setting parameters are very crucial to learning results and generalization ability of SVM. In this paper, parameters selection is regarded as a compound optimization problem and a modified differential evolution (MDE) algorithm is applied to search the optimal parameters. The modified differential evolution adopts a time-varying crossover probability strategy, which can improve the global convergence ability and robustness of the algorithm. Various examples are simulated and the experiment results demonstrate that this proposed approach has better approximation performance than other approaches.

Keywords: Machine learning, Support vector machines, Artificial neural networks, Differential evolution, Function approximation

1. Introduction

Artificial neural networks (ANN) have the ability to approximate nonlinear functions with arbitrary accuracy and which is validated since later 1980s [1]-[2]. Nevertheless the selection of structures and types of ANN dependents on experience greatly, and the training of ANN is based on empirical risk minimization (ERM) principle [3], which aims at minimizing the training errors. So ANN faces some disadvantages such as over-fitting, local optimal and bad generalization ability.

Support vector machines (SVM) [4] is a new machine learning method deriving from statistical learning theory. Since later 1990s, SVM is becoming more and more popular and has been successfully applied to many areas ranging from handwritten digit recognition, speaker identification to function approximation and time series forecasting [5]-[7]. Established on the theory of structural risk minimization (SRM) [3] principle, compared with ANN, SVM has some distinct advantages such as globally optimal, small sample-size, good generalization ability and resistant to over-fitting problem [3,6,7].

It is well known that the generalization performance of SVM depends on a proper setting of the hyper-parameters C , ε and the kernel parameters [8]. However, there is not a thoroughly general way for parameters selection. In the [9], existing practical approaches to parameters setting are summarized and practical recommendations for setting C and ε directly from the training data and estimated noise level are described. Indeed all those approaches (including [9]) for setting parameters of SVM are all based on priori knowledge, user expertise or experimental trial, which can not ensure the parameters value are global.

In this paper, parameters selection is regarded as compound optimization problem and a modified differential evolution algorithm is proposed to select suitable parameters value. To improve the global convergence ability and robustness of the DE, a time-varying crossover probability strategy is proposed. The validity of the proposed approach is illustrated using several nonlinear functions. Simulations results show that the SVM whose parameters selected by modified differential evolution has better performance than ANN and the SVM whose parameters chosen in other ways.

The paper is organized as follows: Section 2 gives a brief introduction to the SVM approximation and parameters. Section 3 describes parameters selection based on modified differential evolution. In Section 4, simulations are illustrated for approximation performance. Finally, Section 5 gives summary and conclusions.

2. SVM approximation and parameters

2.1. SVM approximation

To introduce the subject we will begin by outlining SVM for function approximation. Let the given training data sets represented as

$$D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}. \quad (1)$$

Where $x_i \in R^d$ is an input vector, $y_i \in R$ is its corresponding desired output, and n is the number of training data. In SVM, the original input space is mapped into a high dimensional space called feature space by a nonlinear mapping $x \to g(x)$. Let $f(x)$ be the SVM outputs corresponding to input vector x. In the feature space, a linear function is construct:

$$f(x) = w^T g(x) + b. \quad (2)$$

Where w is a coefficient vector, b is a threshold.

The learning of SVM can be obtained by minimization of the empirical risk on the training data. Where ε–intensive loss function is used for minimization of empirical risk. The loss function is defined as:

$$L^{\varepsilon}(x, y, f) = |y - f(x)|_e = \max(0, |y - f(x)| - \varepsilon) \quad (3)$$

Where ε is a positive parameter to allow approximation errors smaller than ε, the empirical risk is:

$$R_{emp}(w) = \frac{1}{n}\sum_{i=1}^{n} L^{\varepsilon}(y_i - f(x_i)) \quad (4)$$

Besides using ε–intensive loss, SVM tries to reduce model complexity by minimizing $\|w\|^2$, which can be described by slack variables. Introduce variables ξ_i and $\hat{\xi}_i$, then SVM approximation is obtained as the following optimization problem:

$$\text{Min } \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \hat{\xi}_i), \quad (5)$$

Subject to

$$y_i - f(x_i) \le \varepsilon + \xi_i, \quad (6)$$

$$f(x_i) - y_i \le \varepsilon + \hat{\xi}_i, \quad (7)$$

$$\xi_i, \hat{\xi}_i \ge 0. \quad (8)$$

Where C is a positive constant to be regulated. By using the Lagrange multiplier method [3], the minimization of (5) becomes the problem of maximizing the following dual optimization problem:

$$\max \sum_{i=1}^{n} y_i(\hat{\alpha}_i - \alpha_i) - \varepsilon\sum_{i=1}^{n}(\hat{\alpha}_i + \alpha_i) - \frac{1}{2}\sum_{i,j=1}^{n}(\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_j - \alpha_j)K(x_i, x_j), \quad (9)$$

subject to

$$\sum_{i=1}^{n}(\hat{\alpha}_i - \alpha_i) = 0, C \ge \hat{\alpha}_i, \alpha_i \ge 0. \quad (10)$$

Where $\hat{\alpha}_i$ and α_i are Lagrange multipliers, and kernel $K(x_i, x_j)$ is a symmetric function which is equivalent to the dot product in the feature space. The kernel $K(x_i, x_j)$ is defined as the following.

$$K(x_i, x_j) = g(x_i)^T g(x_j). \quad (11)$$

There are some kernels, i.e. polynomial kernel $K(x, y) = (x \cdot y + 1)^d$ and hyperbolic tangent kernel $K(x, y) = \tanh(c_1(x \cdot y) + c_2)$ can be used. Where the Gaussian function is used as the kernel:

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \quad (12)$$

Replacing $\beta_i = \hat{\alpha}_i - \alpha_i$ and relation $\hat{\alpha}_i \alpha = 0$, then the optimization of (6) to (7) is rewritten as:

$$\text{Max } \sum_{i=1}^{n} y_i\beta_i - \varepsilon\sum_{i=1}^{n}|\beta_i| - \frac{1}{2}\sum_{i,j=1}^{n}\beta_i\beta_j K(X_i, X_j), \quad (13)$$

Subject to

$$\sum_{i=1}^{n}\beta_i = 0, C \ge \beta_i \ge -C. \quad (14)$$

The learning results for training data set D can be derived from equation (13) to (14). Note that only some of coefficients β_i are not zeros and the corresponding vectors x are called support vectors (SV). That is, only those vectors whose corresponding coefficients $\hat{\alpha}_i - \alpha_i$ are not zero are SV. Then the approximation function is expressed as equation (15).

$$f(x) = \sum_{i=1}^{p}(\hat{\alpha}_i - \alpha_i)K(x_i, x_j) + b \quad (15)$$

It should be noted that p is the number of SV $f(x)$ is calculated only from SV, and the constant b is expressed as:

$$b = \frac{1}{2}\left\{\min\left(y_i - \sum_{i=1}^{p}(\hat{\alpha}_i - \alpha_i)K(x_i, x)\right) + \max\left(y_i - \sum_{i=1}^{p}(\hat{\alpha}_i - \alpha_i)K(x_i, x)\right)\right\} \quad (16)$$

2.2. SVM parameters

The quality of SVM models strongly depends on a proper setting of parameters and SVM approximation performance is sensitive to parameters. For Gaussian kernel, parameters to be regulated include hyper-parameters C, ε and kernel parameter σ. The values of C, σ and ε are relate to the actual function model and there are not fixed for different data set. So the problem of parameter selection is complicated.

The values of parameter C, σ and ε affect model complexity in a different way. The parameter C determines the trade-off between model complexity and the tolerance degree of deviations larger than ε. The parameter ε controls the width of the ε–intensive

zone and can affect the number of SV in optimization problem. The kernel parameter σ determines the kernel width and relates to the input range of the training data set.

3. Modified differential evolution

3.1. Basic differential evolution

In 1995, R. Storn and K. Price first introduced the differential evolution algorithm. It is one of the optimization techniques and a kind of evolutionary computation technique. The method has been found to be an effective and robust in solving problems with nonlinearity, non-differentiability, multiple optima, and high dimensionality [11]. There are several variants of DE [10]. In this paper, we use the DE scheme classified using notation as DE/rand/1/bin strategy [10]. This strategy is the most often used in practice.

A set of *D* optimization parameters is called an individual. It is represent by a *D*-dimensional parameter vector. A population consists of *NP* parameter vectors x_{ij}^t , i=1,2,…,*NP*, j=1,2,…,*D*. t denotes one generation. *NP* is the number of members in a population. It is not changed during the evolution process. The initial population is chosen randomly with uniform distribution in the search space.

DE has three operations: mutation, crossover and selection. The crucial idea behind DE is a scheme for generating trial vectors. Mutation and crossover are used to generate trial vectors, and selection then determines which of the vectors will survive into the next generation.

3.1.1.Mutation

For each target vector x_{ij}^t , a mutant vector v is generated according to

$$v_i^{t+1} = x_{r1}^t + F \cdot (x_{r2}^t - x_{r3}^t). \quad (17)$$

The r_1, r_2 and r_3 are randomly chosen indexes and $r_1, r_2, r_3 \in \{1,2,..., NP\}$. Note that indexes must be different from each other and from the running index i so that *NP* must be a least four. F is a real number to control the amplification of the difference vector $(x_{r2}^t - x_{r3}^t)$. According to Storn and Price [9], the range of F is in (0,2]. If a component of a mutant vector goes off the search space, then this component is set to bound value.

3.1.2.Crossover

The target vector is mixed with the mutated vector, using the following scheme, to yield the trial vector u.

$$u_{ij}^{t+1} = \begin{cases} v_{ij}^{t+1}, & rand(j) \le CR \quad or \quad j = randn(i) \\ x_{ij}^t, & rand(j) > CR \quad and \quad j \ne randn(i) \end{cases} . \quad (18)$$

Where j=1,2, …, D, $rand(j) \in [0,1]$ is the jth evaluation of a uniform random generator number. $CR \in [0,1]$ is the crossover probability constant, which has to be determined previously by the user. $randn(i) \in (1,2,...,D)$ is a randomly chosen index which ensures that u_i^{t+1} gets at least one element from v_i^{t+1} . Otherwise, no new parent vector would be produced and the population would not alter.

3.1.3.Selection

DE adapts greedy selection strategy. If and only if, the trial vector u_i^{t+1} yields a better fitness function value than x_i^t, then u_i^{t+1} is set to x_i^{t+1}. Otherwise, the old value x_i^t is retained. In this paper the minimization optimization is considered. The selection operator is as following.

$$x_i^{t+1} = \begin{cases} u_i^{t+1}, & f(u_i^{t+1}) < f(x_i^t) \\ x_i^t, & f(u_i^{t+1}) \ge f(x_i^t) \end{cases} . \quad (19)$$

3.2. Time-varying crossover probability strategy

From the crossover operator equation (15) we can see that if the crossover probability constant *CR* is larger then the v_i^{t+1} has more contribution to the u_i^{t+1} and which is propitious to speed up the convergence speed and search the local optima, on the contrary, if the *CR* is smaller then the x_i^t has more contribution to the u_i^{t+1} and which is in favor of keeping the population diversity and exploring the global optima. Good search strategy should keep the population diversity to emphasize the global optima exploration in the beginning searching stage, and enhance the local optima searching ability to improve the optimization precision in the later stage. Based on this idea, a time-varying crossover probability constant strategy is proposed in this paper. That is, with the increasing of iteration times the *CR* becomes larger accordingly from a littler value, then the x_i^t contribute more to the u_i^{t+1} in the beginning stage and v_i^{t+1} has more contribution to the u_i^{t+1} in the later stage. As a result, the algorithm has good global exploration ability in the beginning stage and has good local searching ability in the later stage.

Suppose t is the current iteration and T is the maximum iteration times, the time-varying crossover probability constant *CR* is determined as the following equation (17).

$$CR = CR_{\min} + \frac{t*(CR_{\max} - CR_{\min})}{T} . \quad (20)$$

Where the $CR_{\min}$, $CR_{\max}$ is the minimum crossover probability constant and the maximum crossover probability constant respectively.

3.3. Objective function

For SVM approximation, the objective of parameters selection is to minimize deviations between the outputs of training data and the outputs of SVM. In this paper, the mean square error (MSE) is used as the performance criterion.

$$MSE = \left(\frac{1}{p} \sum_{k=1}^{p} (y_k - f(x_k, w))^2 \right)^{\frac{1}{2}} . \quad (21)$$

Where p is the number of training data, y_k is the output of the kth training data, and $f(x_k, w)$ is the output of SVM as input x_k . Then the objective of the differential evolution is to search optimal parameter C , σ and ε to minimize MSE:

$$\min f(C, \delta, \varepsilon) = \min MSE . \quad (22)$$

Generally, the search range of these parameters is $C \in$ [0.5,100], $\sigma \in$ [0.1,100], $\varepsilon \in$ [0.01,0.2]. For special problem, the search range is changeable.

3.4. Optimization procedures of modified differential evolution

The searching procedures of the modified differential evolution (MDE) were shown as below.

Step1: Specify the number of population *NP*, the difference vector scale factor *F*, the minimum and maximum crossover probability constant $CR_{\min}$ and $CR_{\max}$, and the maximum number of generations. Initialize randomly the individuals of the population and the trial vector in the given searching space.

Step2: Calculate the fitness value of each individual in the population using the objective function given by equation (19).

Step3: Compare each individual's fitness value and get the best fitness and best individual.

Step4: Generate a mutant vector according to equation (14) for each individual.

Step5: According to equation (15), do the crossover operation and yield a trial vector.

Step6: Do the selection operation in terms of equation (16) and generate a new population.

Step7: t=t+1, return to Step2 until to the maximum number of generations.

4. Simulation research

4.1. SVM sensitive to parameters C , σ and ε

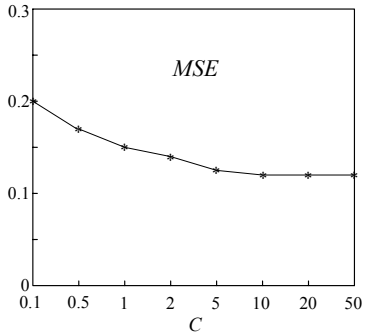
Hermite function (Equ.(23)) is chosen to demonstrate SVM sensitive to parameters. In the range of x, 100 pairs training data $(x_i, y_i)_{i=1}^{100}$ are randomly selected.

$$y = 1.1*(1 - x + 2x^2)e^{-\frac{x^2}{2}} \quad x \in [0, 6] \quad (23)$$

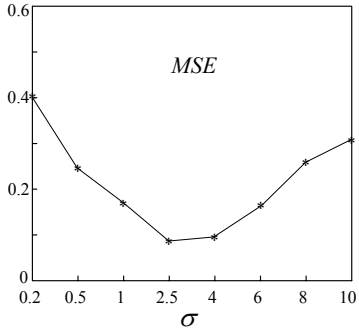
In this section, three experiments are done to demonstrate the influence of parameters value on the performance of SVM. In each experiment, two parameters of C , σ and ε are fixed and the other is changeable. For various C , σ and ε , SVM approximation results are different. The MSE in equation (21) illustrates deviations between $f(x_k, w)$ and y_k . Both Table 1 and Fig.1 show the MSE of SVM approximation. It can be concluded that the performance of SVM is sensitive to parameters value, and for various parameters value, the performance of SVM is somewhat different.

No.	Expt.1 (As Fig. 1 (a)) (δ=3.6, =0.05)		Expt.2 (As Fig.1 (b)) (C=5, ε=0.05)		Expt.3 (As Fig. 1 (c)) (C=5, δ=3.6)	
	C	MSE	δ	MSE	ε	MSE
1	0.1	0.2013	0.2	0.4035	0.002	0.0092
2	0.5	0.1767	0.5	0.2507	0.005	0.0094
3	1	0.1525	1	0.1824	0.01	0.0092
4	2	0.1494	2.5	0.0091	0.02	0.0093
5	5	0.1348	4	0.1057	0.05	0.0090
6	10	0.1310	6	0.1743	0.075	0.0093
7	20	0.1293	8	0.2790	0.1	0.0091
8	50	0.1292	10	0.3116	0.2	0.0092

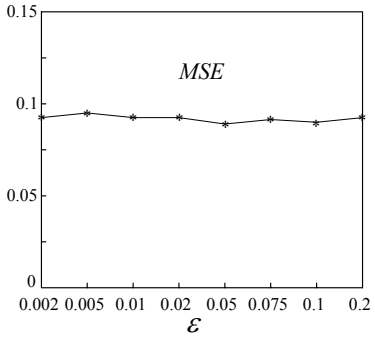
Table 1: MSE comparison in various parameter C, δ and ε.



(a) MSE in various C (δ=3.6, ε=0.05)



(b) MSE in various δ (C=5, ε=0.05)



(c) MSE in various ε (C=5, δ=3.6)

Fig.1: MSE comparison in various parameter C, δ and ε.

4.2. SVM approximation of non-linear function

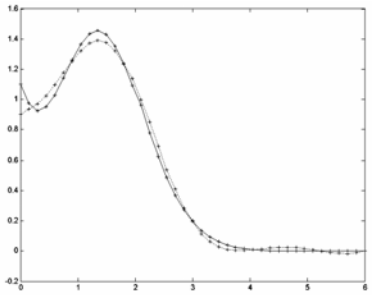
Here the Hermite function is also considered as the test function. 100 pairs training data $(x_i, y_i)_{i=1}^{100}$ and 40 pairs test data $(x_i, y_i)_{i=1}^{40}$ are randomly selected. The MDE algorithm is applied to optimize the parameter C, σ and ε. The objective of optimization is to minimize MSE. In the experiment, the control parameters are as following. The maximum iteration T =500, the difference vector scale factor F=0.5, the $CR_{\min} = 0.1$, the $CR_{\max} = 0.9$, the number of population NP=30, and the search ranges are $C \in [0.5, 15]$, $\sigma \in [0.5, 5]$ and $\varepsilon \in [0.01, 0.2]$, initial values of parameter C, σ and ε are randomly generated in the search ranges. After optimizing using the modified differential evolution, the optimal parameters value are $C = 6.4$, $\sigma = 3.9$ and $\varepsilon = 0.026$.

The performance comparison was done between the proposed approach and other two approximation approaches, the ANN approximation (RBF neural network) and the SVM [(1)] approximation (parameters selected as in [9]). The SVM based on MDE is called SVM [(2)].

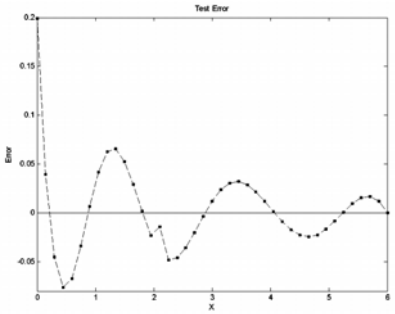
The Table 2 is the statistical results of the comparison. The Fig. 2 to Fig.4 illustrates the test results of these three approximation approaches respectively. In the Fig. 2(a), 3(a) and 4(a), solid line shows 40 pairs test data, whereas dotted line shows approximation outputs of these three approaches respectively. The Fig. 2(b), 3(b) and 4(b) illustrates the actual errors of these three approaches respectively. From these simulations, it can be seen that the SVM [(2)] whose parameters selected based on MDE has better approximation performance than other two approximation approaches.

Approximation approaches	MSE	Max. positive error	Max. negative error	Test curve	Test error curve
ANN	0.064	0.197	-0.084	Fig. 2(a)	Fig. 2(b)
SVM[(1)]	0.052	0.171	-0.061	Fig. 3(a)	Fig. 3(b)
SVM[(2)]	0.032	0.075	-0.041	Fig. 4(a)	Fig. 4(b)

Table 2 Approximation results of different approximation approaches

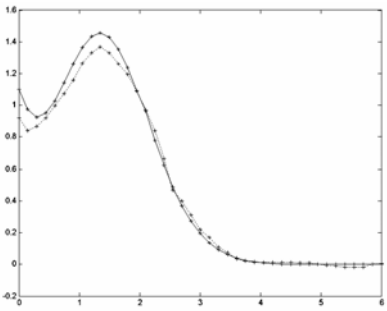


(a) Test data and ANN approximation

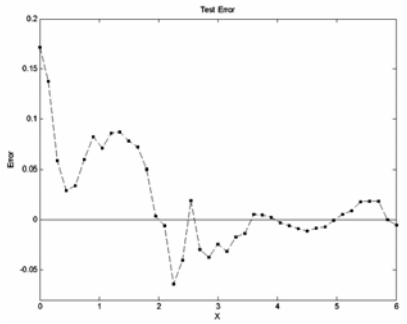


(b) Test error of ANN approximation.

Fig.2: Simulations of ANN approximation.

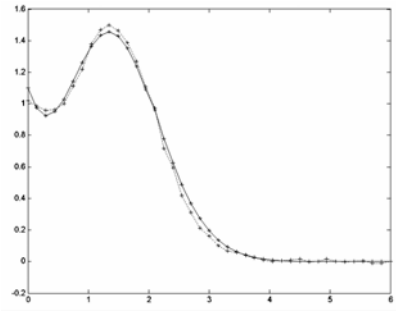


(a) Test data and SVM[(1)] approximation.

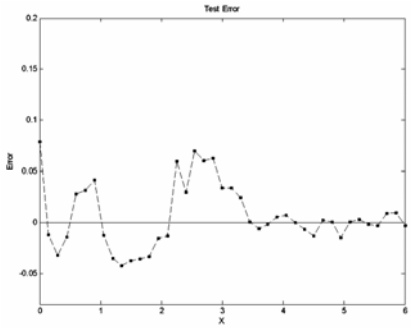


(b) Test error of SVM$^{(1)}$ approximation

Fig.3: Simulations of SVM$^{(1)}$ approximation.



(a) Test data and SVM$^{(2)}$ approximation.



(b) Test error of SVM$^{(2)}$ approximation

Fig.4: Simulations of SVM$^{(2)}$ approximation.

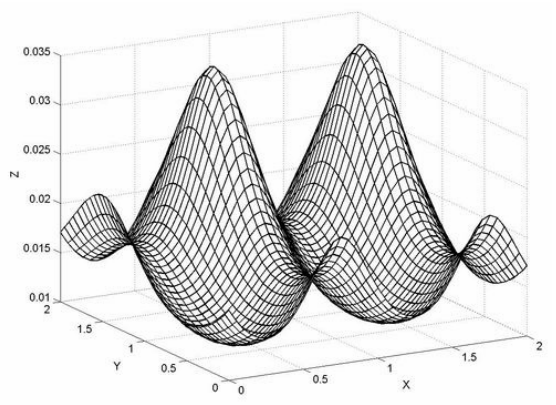
4.3. SVM approximation of two-dimensional function

A two-dimensional nonlinear function was also used to test the approximation performance of SVM based on the MDE. The function is defined as:

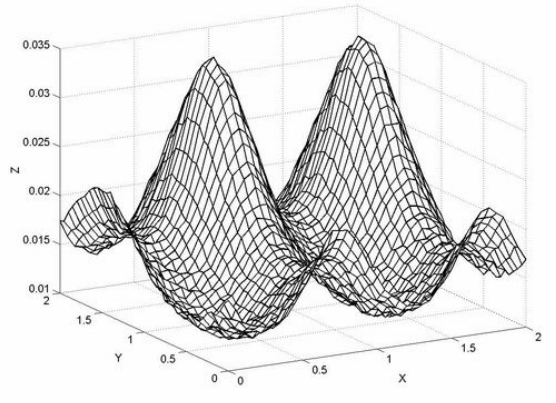
$$z = \frac{1+\sin xy}{4+\sin 2\pi x+\sin \pi y}, \qquad (20)$$

where $x \in [0, 2]$, $y \in [0, 2]$.

The parameters value of SVM is optimized by the MDE. The search range of variables are $C \in [0.5,25]$, $\sigma \in [0.25,\ 10]$ and $\varepsilon \in [0.01,0.2]$. The maximum number of iterations T=600, other control parameters are same as the above. After optimizing, the MSE is 0.0495. The Fig.5 (a) and (b) illustrate the actual model and SVM approximation model. The experiment results show that the SVM based on the MDE also has good approximation performance for two-dimensional nonlinear functions.



(a) Actual model



(b) SVM approximation model

Fig.5 Actual model and SVM approximation model.

5. Conclusions

Good setting parameters are very crucial to SVM learning results and generalization ability. In this paper, the parameters selection of SVM is considered as a compound optimization problem, and a modified differential evolution with time-varying crossover probability constant is proposed to optimize the parameters of SVM. Various examples are simulated to demonstrate the superiority of this proposed approach. The experiment results demonstrate that the SVM based on modified differential evolution has better approximation performance than the RBF ANN and the SVM whose parameters chosen in other ways. The modified differential evolution has good global convergence ability and high optimization precision, and it can widely used in other application areas.

Acknowledgement

This work is partially supported by National Nature Science Foundation of China (Grant No. 60375001) and the Scientific Research Funds of Hunan Province Education Department (Grant No. 05B016).

References

[1] K. Funahashi, On the approximate realization of continuous mappings by neural networks, *Neural Networks*, 2:183-192,1989.

[2] K. Hornik, M. Stinchombe and H. White, Multilayer feedforward networks are universal approximators, *Neural Networks*, 5: 359-366, 1989.

[3] V. Vapnik, An overview of statistical learning theory, *IEEE Transactions on Neural Networks*, 5: 988-999, 1999.

[4] V. Vapnik, *The nature of statistical learning theory*, Springer-verlag, New York, 1995.

[5] C. J. C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, 2: 121-167, 1998.

[6] W. C. Chan, K. C. Cheung and C. J. Harris, On the modelling of nonlinear dynamic system using support vector neural networks, *Engineering Applications of Artificial Intelligence*, 14: 105-113, 2001.

[7] G. Q. Zhu, S. R. Liu and J. S. Yu, Support vector machine and its applications to function approximation, *Journal of East China University of Science and Technology*, 5: 555-559, 2002.

[8] W. J. Wang, Z. B. Xu and W. Z. Lu, Determination of the spread parameter in the Gaussian kernel for classification and regression , *Neurocomputing*, 55: 643-663, 2003.

[9] V. Cherkassky and Y. Ma, Practical selection of SVM parameters and noise estimation for SVM regression, *Neural Networks*, 17: 113-126, 2004.

[10] R. Storn and K. Price, Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces, *Technical report, International Computer Science Institute*, Berkley, 1995.

[11] L. H. Wu, Y. N. Wang and X. F. Yuan, Differential evolution algorithm with adaptive second mutation, *Control and Decision*, 8: 898-902, 2006.