Nonlinear Prediction of Chaotic Time Series Using Support Vector Machines

Sayan Mukherjee **Edgar Osuna** **Federico Girosi**
sayan@ai.mit.edu eosuna@ai.mit.edu girosi@ai.mit.edu
Center for Biological and Computational Learning E25-201
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

A novel method for regression has been recently proposed by V. Vapnik et al. [8, 9]. The technique, called Support Vector Machine (SVM), is very well founded from the mathematical point of view and seems to provide a new insight in function approximation. We implemented the SVM and tested it on the same data base of chaotic time series that was used in [1] to compare the performances of different approximation techniques, including polynomial and rational approximation, local polynomial techniques, Radial Basis Functions, and Neural Networks. The SVM performs better than the approaches presented in [1]. We also study, for a particular time series, the variability in performance with respect to the few free parameters of SVM.

1 Introduction

n this paper we analyze the performance of a new regression technique called *Support Vector Machine* [8, 9]. This technique can be seen as a new way o train polynomial, neural network, or Radial Basis Functions regressors. The main difference between this technique and many conventional regression techniques is that it uses the *Structural Risk Minimization* and not the *Empirical Risk Minimization* induction principle. Since this is equivalent to minimizing an upper bound on the generalization error, rather than minimizing the training error, this technique is expected to perform better than onventional techniques. Our results show that SVM is a very promising egression technique, but in order to assess its reliability and performances nore extensive experimentation will need to be done in the future. We begin y applying SVM to several chaotic time series data sets that were used by asdagli [1] to test and compare the performances of different approximation echniques. The SVM is a technique with few free parameters. In absence of principled way to choose these parameters we performed an experimental tudy to examine the variability in performance as some of these parameters ary between reasonable limits. The paper is organized as follows. In the ext section we formulate the problem of time series prediction and see how is equivalent to a regression problem. In section 3 we briefly review the VM approach to the regression problem. In section 4, the chaotic time series

3-4256-9/97/$10.00 ©1997 IEEE

used to benchmark previous regression methods [1] are introduced. Section 5 contains the experimental results and the comparison with the techniques presented in [1]. Section 6 focuses on a particular series, the Mackey-Glass and examines the relation between parameters of the SVM and generalization error.

2 Time Series Prediction and Dynamical Systems

For the purpose of this paper a *dynamical system* is a smooth map $F : R \times \mathcal{S} \to \mathcal{S}$ where $\mathcal{S}$ is an open set of an Euclidean space. Writing $F(t, \mathbf{x}) = F_t(\mathbf{x})$ the map F has to satisfy the following conditions:

1. $F_0(\mathbf{x}) = \mathbf{x}$;
2. $F_t(F_s(\mathbf{x})) = F_{s+t}(\mathbf{x}) \quad \forall s, t \in R$

For any given initial condition $\mathbf{x}_0 = F_0(\mathbf{x})$ a dynamical system defines a trajectory $\mathbf{x}(t) = F_t(\mathbf{x}_0)$ in the set $\mathcal{S}$. The *direct* problem in dynamical systems consists in analyzing the behavior and the properties of the trajectories $\mathbf{x}(t)$ for different initial conditions $\mathbf{x}_0$. We are interested in a problem similar to the inverse of the problem stated above. We are given a finite portion of a time series $x(t)$, where x is a component of a vector $\mathbf{x}$ that represents a variable evolving according to some unknown dynamical system. We assume that the trajectory $\mathbf{x}(t)$ lies on a manifold with fractal dimension D (a "strange attractor"). Our goal is to be able to predict the future behavior of the time series $x(t)$. Remarkably, this can be done, at least in principle, without knowledge of the other components of the vector $\mathbf{x}(t)$. In fact, Takens embedding theorem [7] ensures that, under certain conditions, for almost all τ and for some $m \leq 2D+1$ there is a smooth map $f : \mathcal{R}^m \to \mathcal{R}$ such that

$$x(n\tau) = f\left(x((n-1)\tau),\ x((n-2)\tau), \ldots, x((n-m)\tau)\right) \qquad (1$$

The value of m used is called the embedding dimension and the smallest value for which (1) is true is called the minimum embedding dimension, m^*. Therefore, if the map f were known, the value of x at time $n\tau$ is uniquely determined by its m values in the past. For simplicity of notation we define the m-dimensional vector

$$\tilde{\mathbf{x}}_{n-1} \equiv \left(x((n-1)\tau),\ x((n-2)\tau), \ldots, x((n-m)\tau)\right)$$

in such a way that eq. (1) can be written simply as $x(n\tau) = f(\tilde{\mathbf{x}}_{n-1})$. If N observations $\{x(n\tau)\}_{n=1}^{N}$ of the time series $x(t)$ are known, then one also knows $N - m$ values of the function f, and the problem of learning the dynamical system becomes equivalent to the problem of estimating the unknown function f from a set of $N - m$ sparse data points in $\mathcal{R}^m$. Man

regression techniques can be used to solve problems of this type. In this paper we concentrate on the Support Vector algorithm, a novel regression technique developed by V. Vapnik et al. [9].

3 Support Vectors Machines for Regression

In this section we sketch the ideas behind the Support Vectors Machines (SVM) for regression, a more detailed description can be found in [9] and [8]. In a regression problem we are given a data set $G = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, obtained sampling, with noise, some unknown function $g(\mathbf{x})$ and we are asked to determine a function f that approximates $g(\mathbf{x})$, based on the knowledge of G. The SVM considers approximating functions of the form:

$$f(\mathbf{x}, \mathbf{c}) = \sum_{i=1}^{D} c_i \phi_i(\mathbf{x}) + b \tag{2}$$

where the functions $\{\phi_i(\mathbf{x})\}_{i=1}^{D}$ are called *features*, and b and $\{c_i\}_{i=1}^{\infty}$ are coefficients that have to be estimated from the data. This form of approximation can be considered as an hyperplane in the D-dimensional feature space defined by the functions $\phi_i(\mathbf{x})$. The dimensionality of the feature space is not necessarily finite, and we will present examples in which it is infinite. The unknown coefficients are estimated by minimizing the following functional:

$$R(\mathbf{c}) = \frac{1}{N} \sum_{i=1}^{N} \mid y_i - f(\mathbf{x}_i, \mathbf{c}) \mid_{\epsilon} + \lambda \|\mathbf{c}\|^2 \tag{3}$$

where λ is a constant and the following *robust* error function has been defined:

$$\mid y_i - f(\mathbf{x}_i, \mathbf{c}) \mid_{\epsilon} = \begin{cases} 0 & \text{if } \mid y_i - f(\mathbf{x}_i, \mathbf{c}) \mid < \epsilon \\ \mid y_i - f(\mathbf{x}_i, \mathbf{c}) \mid & \text{otherwise.} \end{cases} \tag{4}$$

Vapnik showed in [8] that the function that minimizes the functional in eq. 3) depends on a finite number of parameters, and has the following form:

$$f(\mathbf{x}, \alpha, \alpha^*) = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) K(\mathbf{x}, \mathbf{x}_i) + b, \tag{5}$$

where $\alpha_i^* \alpha_i = 0$, $\alpha_i, \alpha_i^* \geq 0$ $i = 1, \ldots, N$, and $K(\mathbf{x}, \mathbf{y})$ is the so called *kernel* function, and describes the inner product in the D-dimensional feature space:

$$K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{D} \phi_i(\mathbf{x}) \phi_i(\mathbf{y})$$

The interesting fact is that for many choices of the set $\{\phi_i(\mathbf{x})\}_{i=1}^{D}$, including infinite dimensional sets, the form of K is analytically known and very

simple, and the features ϕ_i never need to be computed in practice becaus the algorithm relies only on computation of scalar products in the featur space. Several choices for the kernel K are available, including gaussians tensor product B-splines and trigonometric polynomials. The coefficients and α^* are obtained by maximizing the following quadratic form:

$$R(\alpha^*, \alpha) = -\epsilon \sum_{i=1}^{N}(\alpha_i^*+\alpha_i)+\sum_{i=1}^{N} y_i(\alpha_i^*-\alpha_i)-\frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i^*+\alpha_i)(\alpha_i^*-\alpha_i)K(\mathbf{x}_i, \mathbf{x}_j$$

(6

subject to the constraints $0 \leq \alpha_i^*, \alpha_i \leq C$ and $\sum_{i=1}^{N}(\alpha_i^* - \alpha_i) = 0$. Due to th nature of this quadratic programming problem, only a number of coefficient $\alpha_i^* - \alpha_i$ will be different from zero, and the data points associated to ther are called *support vectors*. The parameters C and ϵ are two free parameter of the theory, and their choice is left to the user. They both control th VC-dimension of the approximation scheme, but in different ways. A clea theoretical understanding is still missing and we plan to conduct experimenta work to understand their role.

4 Benchmark Time Series

We tested the SVM regression technique on the same set of chaotic tim series that has been used in [1] to test and compare several approximatio techniques.

4.1 The Mackey-Glass time series

We considered two time series generated by the Mackey-Glass delay-differenti equation [4]:

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-\Delta)}{1+x(t-\Delta)^{10}}, \qquad (7$$

with parameters $\Delta = 17, 30$ and embedding dimensions $m = 4, 6$ respectivel We denote these two time-series by MG_{17} and MG_{30}. In order to be co sistent with [1] the initial condition for the above equation was $x(t) = 0$ for $0 \leq t \leq \Delta$, and the sampling rate $\tau = 6$. The series were generated b numerical integration using a fourth order Runge-Kutta method.

4.2 The Ikeda map

The Ikeda map [2] is a two dimensional time series which is generated iterati the following map:

$$f(x_1, x_2) = (1 + \mu(x_1 \cos\omega - x_2 \sin\omega), \mu(x_1 \sin\omega + x_2 \cos\omega)), \qquad ($$

where $\omega = 0.4 - 6.0/(1 + x_1^2 + x_2^2)$. In [1] Casdagli considered both this time series, that we will denote by $Ikeda_1$, and the one generated by the fourth iterate of this map, which has a more complicated dynamic, and that will be denoted by $Ikeda_4$.

4.3 The Lorenz time series

We also considered the time series associated to the variable x of the Lorenz differential equation [3]:

$$\dot{x} = \sigma(y - x), \quad \dot{y} = rx - y - xz, \quad \dot{z} = xy - bz \tag{9}$$

where $\sigma = 10$, $b = \frac{8}{3}$, and $r = 28$. We considered two different sampling rates, $\tau = 0.05$ and $\tau = 0.20$, generating the two time series $Lorenz_{0.05}$ and $Lorenz_{0.20}$. The series were generated by numerical integration using a fourth order Runge-Kutta method.

5 Comparison with Other Techniques

In this section we report the results of the SVM on the time series presented above, and compare them with the results reported in [1] about different approximation techniques (polynomial, rational, local polynomial, Radial Basis Functions with multiquadrics as basis function, and Neural Networks). In all cases a time series $\{x(n\tau)\}_{n=1}^{N+M}$, was generated: the first N points were used for training and the remaining M points were used for testing. In all cases N was set to 500, except for the $Ikeda_4$, for which $N = 100$, while M was always set to 1000. The data sets we used were the same that were used in [1]. Following [1], denoting by $\tilde{f}_N$ the predictor built using N data points, the following quantity was used as a measure of the generalization error of $\tilde{f}_N$:

$$\sigma^2(\tilde{f}_N) = \frac{1}{M} \sum_{n=N+1}^{M} \frac{(x(n\tau) - \tilde{f}_N(\tilde{\mathbf{x}}_{n-1}))^2}{\text{Var}} \tag{10}$$

where Var is the variance of the time series. We implemented the SVM using MINOS 5.4 [5] as the solver for the Quadratic Programming problem of eq. (6). Details of our implementation can be found in [6]. For each series we choose the kernel, K, and parameters of the kernel that gave us the smallest generalization error. This is consistent with the strategy adopted in [1]. The results are reported in table (1). The last column of the table contains the results of our experiments, while the rest of the table is from [1] with parameters and kernels set as in the remaining part of this section.

Mackey-Glass time-series: the kernel K was chosen to have the following form:

$$K(\mathbf{x}, \mathbf{x}_i) = \sum_{d=1}^{m} \frac{\sin((\nu + 1/2)(x^{(d)} - x_i^{(d)}))}{\sin(0.5(x^{(d)} - x_i^{(d)}))} \tag{11}$$

where $x^{(d)}$ is the d-th component of the m-dimensional vector $\mathbf{x}$, and a ν is an integer. This kernel generates an approximating function that is an additive trigonometric polynomial of degree ν, and correspond to features ϕ_i that are trigonometric monomials up to degree ν. We tried various values for ϵ and C. The embedding dimension for the series MG_{17} and MG_{30} were $m = 4$ and $m = 6$ in accordance with the work by Casdagli, and we used $\nu = 200$ and $\epsilon = 10^{-3}$.

Lorenz time-series: For the Lorenz$_0$.05 and Lorenz$_0$.20 series the polynomial kernels of order 6 and 10 were used. The embedding dimensions used were 6 and 10 respectively. The value of ϵ used was 10^{-3}.

Ikeda map: a B-spline of order 3 was used as the kernel, and the value of ϵ was 10^{-3}.

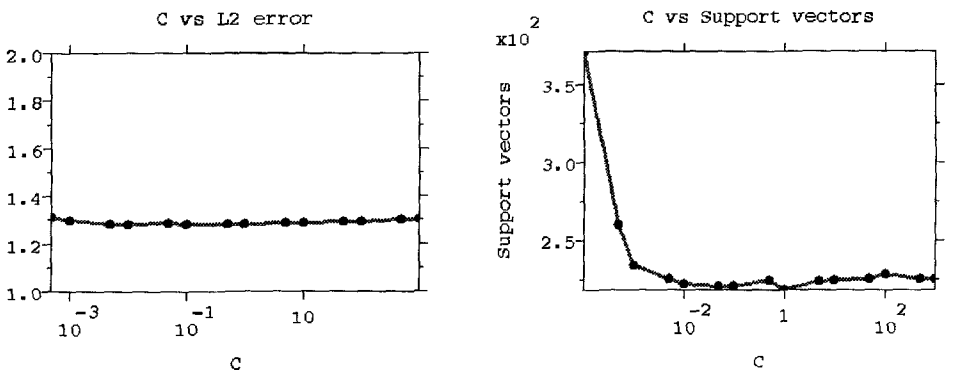
	Poly	Rational	Loc$^{d=1}$	Loc$^{d=2}$	RBF	N.Net	SVM
MG_{17}	-1.95$^{(7)}$	-1.14$^{(2)}$	-1.48	-1.89	-1.97	-2.00	-2.36 (258)
MG_{30}	-1.40$^{(4)}$	-1.33$^{(2)}$	-1.24	-1.42	-1.60	-1.5	-1.87 (341)
$Ikeda_1$	-5.57$^{(12)}$	-8.01$^{(8)}$	-1.71	-2.34	-2.95	-	-6.21 (374)
$Ikeda_4$	-1.05$^{(14)}$	-1.39$^{(14)}$	-1.26	-1.60	-2.10	-	-2.31 (427)
$Lor_{0.05}$	-4.62$^{(6)}$	-4.30$^{(3)}$	-2.00	-3.48	-3.54	-	-4.76 (389)
$Lor_{0.20}$	-1.05$^{(5)}$	-1.39$^{(6)}$	-1.26	-1.60	-2.10	-	-2.21 (448)

Table 1: Estimated values of $\log_{10} \sigma(\tilde{f}_n)$ for the SVM algorithm and for various regression algorithms, as reported in [1]. The degrees used for the best rational and polynomial regressors are in superscripts beside the estimates. Loc$^{d=1}$ and Loc$^{d=2}$ refer to local approximation with polynomials of degree 1 and 2 respectively. The numbers in parenthesis near the SVM estimates are the number of support vectors obtained by the algorithm. The Neural Networks results which are missing were also missing in [1].

6 Sensitivity of SVM to Parameters and Embedding Dimension

In this section we report our observations on how the generalization error and the number of support vectors vary with respect to the free parameters of the SVM and to the choice of the embedding dimension. The parameters

we analyze are therefore C, ϵ, the dimensionality of the feature space D, and the embedding dimension m. All of these results are for the MG_{17} series. Figure 1a demonstrates that C has little effect on the generalization error (the plot spans over 7 orders of magnitude). The parameter C has also little effect on the number of support vector, as shown in figure 1b, which remains almost constant in the range $10^{-2} - 10^{2}$. The results were similar for kernels with low ($D = 2$), high ($D = 802$) and infinite dimensionality of the feature spaces.



(a) (b)

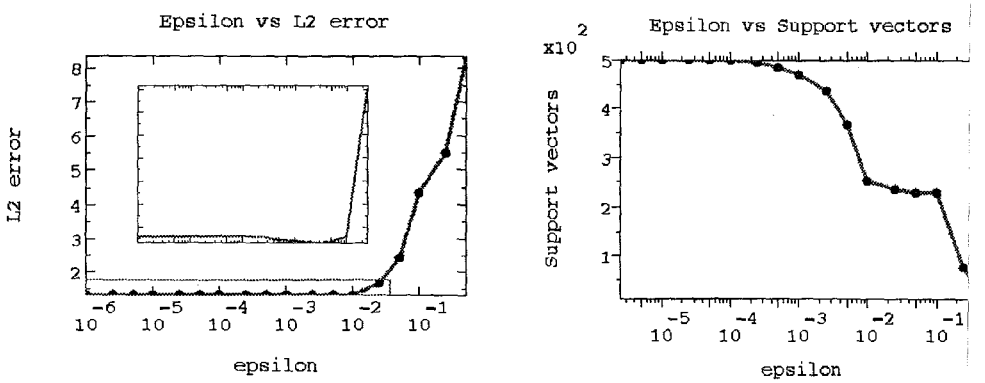
Figure 1: (a) The L^2 generalization error versus C for the MG_{17} series. (b) The number of support vectors versus C for the same series. The kernel was an additive trigonometric polynomial with $\nu = 200$.

The parameter ϵ has a strong effect on the number of support vectors and on the generalization error, and its relevance is related to D. In order to see why this happens, remember that if $I[\mathbf{c}]$ and $I_{\text{emp}}[\mathbf{c}]$ are respectively the expected risk and empirical risk, with probability $1 - \eta$:

$$I[\mathbf{c}] \leq I_{\text{emp}}[\mathbf{c}] + \tau \sqrt{\frac{h(\log \frac{2N}{h} + 1) - \log \frac{\eta}{4}}{N}}, \tag{12}$$

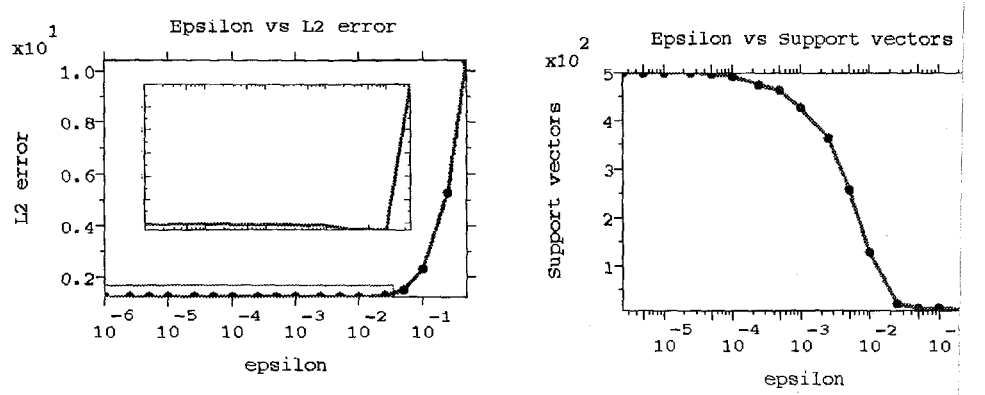
where τ is a bound on the cost function used to define the expected risk and h is the VC-dimension of the approximation scheme. It is known that the VC-dimension satisfies $h \leq \min(\frac{R^2(A+1)^2}{\epsilon^2}, D) + 1$[10], where R is the radius of the smallest sphere that contains all the data points in the feature space, A is a bound on the norm of the vector of coefficients. When D is small, the VC-dimension h is not dependent on ϵ and the second term on the bound of the generalization error is constant and therefore a very small ϵ does not cause overfitting. For the same reason when D is large the term is very sensitive to ϵ and overfitting occurs for small ϵ. Numerical results confirm this. For example, figures 2 and 3 which correspond to feature spaces of 802

and infinite dimensions, respectively, show overfitting. (The kernels used were the additive trigonometric polynomial with $\nu = 200$ and a B-spline of order 3, respectively.) Figure 4 corresponds to a feature space of 10 dimensions and there is no overfitting. (The kernel used was the additive trigonometric polynomial with $\nu = 2$.)



(a) (b)

Figure 2: (a) The L^2 generalization error versus ϵ with a 802 dimensional feature space. The inset magnifies the boxed region in the lower left section of the plot. Note that overfitting occurs. (b) The number of support vectors versus ϵ for the same feature space.



(a) (b)

Figure 3: (a) The L^2 generalization error versus ϵ with an infinite dimensional feature space. The inset magnifies the boxed region in the lower left section of the plot. Note that ovefitting occurs (b) The number of support vectors versus ϵ for the same feature space.

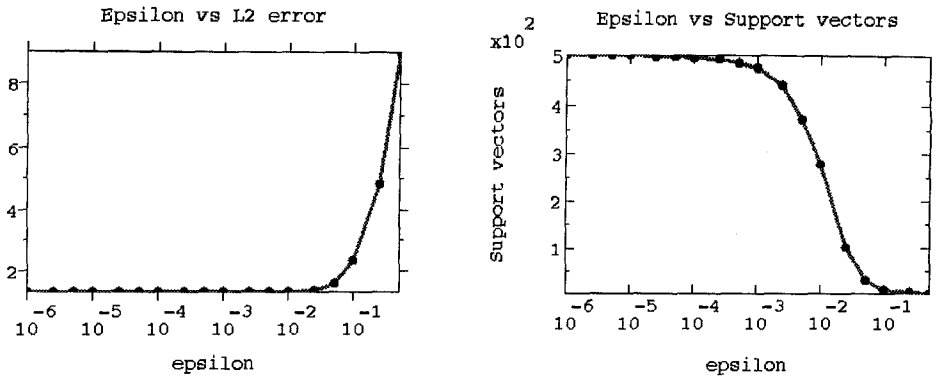


Figure 4: (a) The L^2 generalization error versus ϵ with a 10 dimensional feature space. Note that there is no overfitting (b) The number of support vectors versus ϵ for the same feature space.

The effect of the embedding dimension m on generalization error was also examined. According to Takens theorem the generalization error should decrease as m approaches the minimum embedding dimension, m^*. Above m^* there should be no decrease in the generalization error. However, if the regression algorithm is sensitive to overfitting the generalization error can increase for $m > m^*$. The minimal embedding dimension of the MG_{17} series is 4. Our numerical results demonstrate the SVM does not overfit for the case of a low dimensional kernel and overfits slightly for high dimensional kernels, see figure 5. The additive trigonometric polynomial with $\nu = 200$ and 2 were used for this figure.

7 Discussion

The SVM algorithm showed excellent performances on the data base of chaotic time series, outperforming the other techniques in the benchmark in all but one case. The generalization error is not sensitive to the choice of C, and very stable with respect to ϵ in a wide range. The variability of the performances with ϵ and D seems consistent with the theory of VC bounds.

Acknowledgements

Figures in this paper were displayed with the Signiscope^R.

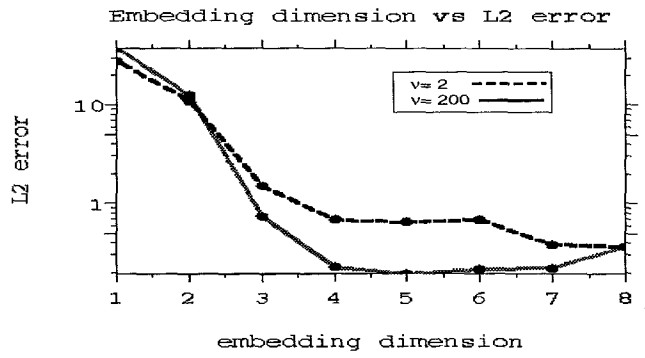


Figure 5: The L^2 generalization error versus the embedding dimension. The solid line is for a 802 dimensional feature space and the dashed line is for a 10 dimensional feature space.

References

[1] M. Casdagli. Nonlinear prediction of chaotic time-series. *Physica D*, 35:335–356, 1989.

[2] K. Ikeda. Multiple-valued stationary state and its instability of light by a ring cavity system. *Opt. Commun.*, 30:257, 1979.

[3] E.N. Lorenz. Deterministic non-periodic flow. *J. Atoms. Sci.*, 26:636, 1969.

[4] M.C. Mackey and J. Glass. Oscillation and chaos in physiological contol systems. *Science*, 197:287, 1977.

[5] B.A. Murtagh and M. Saunders. Large-scale linearly constrained optimization. *Mathematical Programming*, 14:41–72, 1978.

[6] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. A.I. Memo 1602, MIT A. I. Lab., 1997.

[7] F. Takens. Detecting strange attractors in fluid turbulence. In D. Rand and L.S. Young, editors, *Dynamical Systems and Turbulence*. Springer-Verlag, Berlin, 1981.

[8] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[9] V. Vapnik, S.E. Golowich, and A. Smola. Support vector method for function approximation, regression estimation, and signal processing. In *Advances in Neural information processings systems 8*, San Mateo, CA, 1996. Morgan Kaufmann Publishers. (in press).

[10] V. N. Vapnik. *Statistical learning theory*. J. Wiley, 1997.