Recurrent Least Squares Support Vector Machines *

J.A.K. Suykens and J. Vandewalle

Katholieke Universiteit Leuven

Department of Electrical Engineering, ESAT-SISTA

Kardinaal Mercierlaan 94, B-3001 Leuven (Heverlee), Belgium

Tel: 32/16/32 18 02 Fax: 32/16/32 19 70

Email: johan.suykens@esat.kuleuven.ac.be

Corresponding author: Johan Suykens

(Running title: Recurrent Support Vector Machines)

**IEEE Transactions on Circuits and Systems-I,
Vol.47, No.7, pp.1109-1114, Jul. 2000**

*This research work was carried out at the ESAT laboratory and the Interdisciplinary Center of Neural Networks ICNN of the Katholieke Universiteit Leuven, in the framework of the FWO project G.0262.97 *Learning and Optimization: an Interdisciplinary Approach*, the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian State, Prime Minister's Office for Science, Technology and Culture (IUAP P4-02 & IUAP P4-24) and the Concerted Action Project MIPS (*Modelbased Information Processing Systems*) of the Flemish Community. Johan Suykens is a postdoctoral researcher with the National Fund for Scientific Research FWO - Flanders.

Abstract

The method of support vector machines has been developed for solving classification and static function approximation problems. In this paper we introduce support vector machines within the context of recurrent neural networks. Instead of Vapnik's epsilon insensitive loss function, we consider a least squares version related to a cost function with equality constraints for a recurrent network. Essential features of support vector machines remain such as Mercer's condition and the fact that the output weights are a Lagrange multiplier weighted sum of the data points. The solution to recurrent least squares support vector machines is characterized by a set of nonlinear equations. Due to its high computational complexity, we focus on a limit case of assigning the squared error an infinitely large penalty factor with early stopping as a form of regularization. The effectiveness of the approach is demonstrated on trajectory learning of the double scroll attractor in Chua's circuit.

Keywords. Recurrent neural networks, Support vector machines, Radial basis functions, Double scroll.

1 Introduction

Recently, support vector machines (SVM's) have been introduced as a new method for solving classification and function estimation problems with many successful applications [24, 25, 26, 27]. SVM's are based on the structural risk minimization principle. The quality and complexity of the SVM solution does not depend directly on the dimensionality of the input space. The derivation of SVM's is based on constructing an optimal separating hyperplane after nonlinearly mapping the input space into a higher dimensional space. The explicit construction of this mapping is avoided by the application of Mercer's condition. Kernels that satisfy this condition and can be employed for SVM's are polynomials, splines, radial basis functions and multilayer perceptrons with one hidden layer. For classification problems the parameters which are related to these kernel functions are chosen so as to minimize an upper bound on the Vapnik-Chervonenkis (VC) dimension of the SVM. The training of SVM's with Vapnik's epsilon insensitive loss function is done by quadratic programming. The number of hidden units in the SVM is determined by the number of support vector data, which corresponds to the number of non-zero coefficients in the solution vector to the QP problem. A least squares version of SVM's for function estimation and classification has been investigated in [14, 23]. The training is done then by solving a set of linear equations. On the other hand, sparseness is lost in the least squares version.

In this paper we introduce a recurrent neural network version of least squares support vector machines (LS-SVM's). We investigate a class of nonlinear output error models for the autonomous case. In time-series prediction applications the difference between feedforward and recurrent is crucial, especially concerning chaotic systems [9, 20, 29]. Often the model is trained as a feedforward network (one step ahead predictor). On the other hand in order to make a further continuation of the given time-series, the identified feedforward model has to be iterated, i.e. it has to be used as a recurrent network for prediction. Due to the sensitivity of chaotic systems for small perturbations, replacing the feedforward model by its recurrent version (replacing the true signal values by the estimated ones as input of the network) may result into large deviations. In this paper we consider the problem of trajectory learning of the double scroll attractor [4, 5, 10] by recurrent neural nets which are parametrized by SVM's. In [21] a simple recurrent neural state space model has been trained for the double scroll. It has been observed that by using classical dynamic backpropagation [11, 12] this is a difficult task [21].

For the recurrent LS-SVM's essential features of support vector machines, such as Mercer's condition and the fact that the output weights are a Lagrange multiplier weighted sum of the data points, are still applicable. The solution to recurrent LS-SVM's is characterized by a set of nonlinear equations. Because of the considered least squares norm there seems to be some similarity at first sight with Widrow's Adaline [9, 29]. However, recurrent LS-SVM's and Widrow's Adaline are fundamentally different, e.g. due to the recurrent versus feedforward architecture respectively, with corresponding implications for the training procedure.

The training problem for recurrent least squares SVM's is formulated as a non-convex constrained nonlinear optimization problem in the error variables and the Lagrange multipliers. The computational cost of the training process is relatively expensive. Due to this high computational complexity, we focus on a limit case of assigning the squared error an infinitely large penalty factor with early stopping as a form of regularization. A sequential quadratic programming [7] training process is behaving well for this problem. For the example of trajectory learning of the double scroll attractor, recurrent LS-SVM's with radial basis function (RBF) kernel yield a good generalization performance. Due to the study of the limit case, early stopping has to be applied which is equivalent to a form of regularization [1, 16]. The overfitting phenomenon can be detected on the training data itself by taking the first given data points in time as initial state for the recurrent LS-SVM, while in the case of standard feedforward models early stopping is usually based upon a test data set. Finally, part of the SVM theory that is related to upper bounds on the generalization error is not applicable because the input vectors to the recurrent architecture are not independent from each other.

This paper is organized as follows. In Section 2 we review work on classical SVM's and least squares SVM's. In Section 3 recurrent least squares SVM's are introduced. In Section 4 recurrent least squares SVM's with RBF kernel are applied to trajectory learning of the double scroll attractor.

2 Support vector machines and function estimation

Here we review basic ideas of the support vector method of static function estimation and least squares support vector machines. For detailed information about SVM's we refer to [3, 6, 15, 17, 18, 19, 24, 25, 26, 27].

Let us consider first the regression in the set of linear functions

$$\mathcal{F}(\mathcal{X}) = \mathcal{W}^T \mathcal{X} + \mathcal{B} \tag{1}$$

given training data $\{\mathcal{X}_i, \mathcal{Y}_i\}_{i=1}^{M}$ where M denotes the number of training data, $\mathcal{X}_i \in \mathbb{R}^m$ are the input data and $\mathcal{Y}_i \in \mathbb{R}$ are the output data and $\mathcal{W} \in \mathbb{R}^m$, $\mathcal{B} \in \mathbb{R}$. Originally, in the support vector method one aims at minimizing the empirical risk

$$\mathcal{R}_{emp}(\mathcal{W}, \mathcal{B}) = \frac{1}{M} \sum_{i=1}^{M} |\, \mathcal{Y}_i - \mathcal{W}^T \mathcal{X}_i - \mathcal{B} \,|_\epsilon \tag{2}$$

subject to elements of a structure $\mathcal{S}_n$, defined by the inequality $\mathcal{W}^T \mathcal{W} \leq c_n$. The loss function employs Vapnik's ϵ-insensitive model:

$$|\, \mathcal{Y}_i - \mathcal{F}(\mathcal{X}_i) \,|_\epsilon = \begin{cases} 0, & \text{if } |\, \mathcal{Y}_i - \mathcal{F}(\mathcal{X}_i) \,| \leq \epsilon \\ |\, \mathcal{Y}_i - \mathcal{F}(\mathcal{X}_i) \,| - \epsilon, & \text{otherwise.} \end{cases} \tag{3}$$

The function estimation problem is formulated then as

$$\min_{\mathcal{W},\mathcal{B},\xi^*,\xi} \mathcal{J}_\epsilon(\mathcal{W},\xi^*,\xi) = \frac{1}{2}\mathcal{W}^T\mathcal{W} + \gamma\,\{\sum_{i=1}^{M}\xi_i^* + \sum_{i=1}^{M}\xi_i\} \tag{4}$$

subject to the constraints

$$\begin{cases} \mathcal{Y}_i - \mathcal{W}^T\mathcal{X}_i - \mathcal{B} & \leq & \epsilon + \xi_i^*, & i = 1,..,M \\ -\mathcal{Y}_i + \mathcal{W}^T\mathcal{X}_i + \mathcal{B} & \leq & \epsilon + \xi_i, & i = 1,..,M \\ \xi_i^* & \geq & 0, & i = 1,..,M \\ \xi_i & \geq & 0, & i = 1,..,M \end{cases}$$

where ξ_i, ξ_i^* are slack variables and γ is a positive real constant. The solution is given by

$$\mathcal{W} = \sum_{i=1}^{M}(\alpha_i^* - \alpha_i)\mathcal{X}_i \tag{5}$$

where α_i^*, α_i are obtained by solving a quadratic program and are the Lagrange multipliers related to the first and second set of constraints. The data points corresponding to nonzero values for $(\alpha_i^* - \alpha_i)$ in (5) are called support vectors. Typically, many of these values are equal to zero. Another loss function which has been investigated is

$$\mathcal{J}_{\epsilon,p}(\mathcal{W},\xi^*,\xi) = \frac{1}{2}\mathcal{W}^T\mathcal{W} + \gamma\,\{\sum_{i=1}^{M}(\xi_i^*)^p + \sum_{i=1}^{M}(\xi_i)^p\} \tag{6}$$

where $p = 1$ corresponds to (4).

The work in this paper is related to a least squares version of SVM's. This version has been investigated in [14] for function estimation and in [23] for classification problems. For function estimation it corresponds to the following form of ridge regression

$$\min_{\mathcal{W},\mathcal{B},\xi} \mathcal{J}_{LS}(\mathcal{W},\mathcal{B},\xi) = \frac{1}{2}\mathcal{W}^T\mathcal{W} + \gamma\,\frac{1}{2}\sum_{i=1}^{M}\xi_i^2 \tag{7}$$

subject to the equality constraints

$$\mathcal{Y}_i = \mathcal{W}^T\mathcal{X}_i + \mathcal{B} + \xi_i,\ \ i = 1,...,M.$$

One defines the Lagrangian

$$\mathcal{L}_{LS}(\mathcal{W},\mathcal{B},\xi;\alpha) = \mathcal{J}_{LS}(\mathcal{W},\mathcal{B},\xi) - \sum_{i=1}^{M}\alpha_i\,(\,\mathcal{W}^T\mathcal{X}_i + \mathcal{B} + \xi_i - \mathcal{Y}_i\,) \tag{8}$$

where α_i are Lagrange multipliers (which can be either positive or negative due to the equality constraints as follows from the Kuhn-Tucker conditions [7]). The conditions for optimality

$$\begin{cases} \frac{\partial \mathcal{L}_{LS}}{\partial \mathcal{W}} = 0 & \rightarrow \quad \mathcal{W} = \sum_{i=1}^{M} \alpha_i \mathcal{X}_i \\ \\ \frac{\partial \mathcal{L}_{LS}}{\partial \mathcal{B}} = 0 & \rightarrow \quad \sum_{i=1}^{M} \alpha_i = 0 \\ \\ \frac{\partial \mathcal{L}_{LS}}{\partial \xi_i} = 0 & \rightarrow \quad \alpha_i = \gamma \xi_i, \qquad\qquad\qquad i = 1, ..., M \\ \\ \frac{\partial \mathcal{L}_{LS}}{\partial \alpha_i} = 0 & \rightarrow \quad \mathcal{W}^T \mathcal{X}_i + \mathcal{B} + \xi_i - \mathcal{Y}_i = 0, \quad i = 1, ..., M \end{cases} \tag{9}$$

can be written immediately as the solution to the following set of linear equations

$$\left[\begin{array}{ccc|c} I & 0 & 0 & -\mathcal{X} \\ 0 & 0 & 0 & -\vec{1}^T \\ 0 & 0 & \gamma I & -I \\ \hline \mathcal{X}^T & \vec{1} & I & 0 \end{array}\right] \left[\begin{array}{c} \mathcal{W} \\ \mathcal{B} \\ \xi \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ 0 \\ \hline \mathcal{Y} \end{array}\right] \tag{10}$$

where $\mathcal{X} = [\mathcal{X}_1 \dots \mathcal{X}_M]$, $\mathcal{Y} = [\mathcal{Y}_1; ...; \mathcal{Y}_M]$, $\vec{1} = [1; ...; 1]$, $\xi = [\xi_1; ...; \xi_M]$, $\alpha = [\alpha_1; ...; \alpha_M]$. The solution is finally given by

$$\left[\begin{array}{c|c} 0 & \vec{1}^T \\ \hline \vec{1} & \mathcal{X}^T \mathcal{X} + \gamma^{-1} I \end{array}\right] \left[\begin{array}{c} \mathcal{B} \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline \mathcal{Y} \end{array}\right] \tag{11}$$

with $\mathcal{W} = \sum_i \alpha_i \mathcal{X}_i$, $\xi_i = \alpha_i / \gamma$. The support values α_i are proportional now to the errors at the data points (9).

So far we explained the linear case. Support vector machines with polynomials, splines, radial basis function networks or multilayer perceptrons as kernels are obtained after mapping the input data into a higher dimensional space by $\varphi(\mathcal{X}_i)$, where $\varphi(\cdot) : \mathbb{R}^m \rightarrow \mathbb{R}^{n_h}$. The number n_h doesn't have to be specified because of the application of Mercer's condition, which means that

$$K(\mathcal{X}_i, \mathcal{X}_j) = \varphi(\mathcal{X}_i)^T \varphi(\mathcal{X}_j) \tag{12}$$

can be imposed for these kernels. For least squares SVM's the nonlinear function can then be represented as

$$\mathcal{Y}_i = \sum_{j=1}^{M} \alpha_j K(\mathcal{X}_j, \mathcal{X}_i) + \mathcal{B} \tag{13}$$

where the parameters $\alpha_j, \mathcal{B}$ follow from (11) after replacing $\mathcal{X}_i^T \mathcal{X}_j$ by $K(\mathcal{X}_i, \mathcal{X}_j)$. For RBF kernels one has

$$K(\mathcal{X}_i, \mathcal{X}_j) = \exp(-\nu \|\mathcal{X}_i - \mathcal{X}_j\|_2^2) \tag{14}$$

where ν is a positive real constant. The number of hidden units in the LS-SVM is equal to the number of data points, while in standard SVM's this is equal to the number of support vectors. In other words, sparseness is lost in the LS-SVM case.

3 Recurrent least squares SVM's

Given a deterministic nonlinear dynamical system with input $u_k \in \mathbb{R}$ and output $y_k \in \mathbb{R}$, we consider nonlinear models of the form

$$\hat{y}_k = f(\hat{y}_{k-1}, \hat{y}_{k-2}, ..., \hat{y}_{k-p}, u_{k-1}, u_{k-2}, ..., u_{k-p}) \tag{15}$$

where $\hat{y}_k$ denotes the estimated output and f is a smooth nonlinear mapping. Depending on the field, such models are called recurrent input/output models [9, 20, 29], nonlinear output error models (NOE) [16] or parallel models [11]. These are opposed to

$$\hat{y}_k = f(y_{k-1}, y_{k-2}, ..., y_{k-p}, u_{k-1}, u_{k-2}, ..., u_{k-p}) \tag{16}$$

which are called feedforward models, NARX models or series-parallel models. Models of this form can be trained by means of the standard SVM methods discussed in Section 2. The parametrization of f by SVM's is static because there is no recursion in the variable $\hat{y}_k$.

Without loss of generality concerning the methods, we will further discuss the autonomous case of the recurrent model (15)

$$\hat{y}_k = f(\hat{y}_{k-1}, \hat{y}_{k-2}, ..., \hat{y}_{k-p}). \tag{17}$$

We take the following parametrization

$$\hat{y}_k = w^T \varphi(\begin{bmatrix} \hat{y}_{k-1} \\ \hat{y}_{k-2} \\ \vdots \\ \hat{y}_{k-p} \end{bmatrix}) + b. \tag{18}$$

We express this model in terms of the given data and the error variables:

$$y_k - e_k = w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) + b \tag{19}$$

where $e_k = y_k - \hat{y}_k$, $x_{k-1|k-p} = [y_{k-1}; y_{k-2}; ...; y_{k-p}]$, $\xi_{k-1|k-p} = [e_{k-1}; e_{k-2}; ...; e_{k-p}]$ by definition. The nonlinear mapping $\varphi(\cdot) : \mathbb{R}^p \to \mathbb{R}^{n_h}$ will be related to Mercer's condition. The output weight vector and bias term are denoted by $w \in \mathbb{R}^{n_h}$ and $b \in \mathbb{R}$. Recurrent neural networks (15) and (18) are classically trained by dynamic backpropagation or backpropagation through time [11, 12, 28]. The goal of this Section is to develop a SVM approach for the recurrent model.

We formulate the training of the network (19) as

$$\min_{w,b,e} \mathcal{J}(w, b, e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=p+1}^{N+p} e_k^2 \tag{20}$$

subject to the equality constraints

$$y_k - e_k = w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) + b, \quad k = p+1, ..., N+p. \tag{21}$$

We define the Lagrangian

$$\mathcal{L}(w, b, e; \alpha) = \mathcal{J}(w, b, e) + \sum_{k=p+1}^{N+p} \alpha_{k-p} \, [y_k - e_k - w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) - b]. \tag{22}$$

The conditions for optimality are given by

$$\left\{ \begin{array}{lcl} \frac{\partial \mathcal{L}}{\partial w} & = & w - \sum_{k=p+1}^{N+p} \alpha_{k-p} \, \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) = 0 \\ \\ \frac{\partial \mathcal{L}}{\partial b} & = & \sum_{k=p+1}^{N+p} \alpha_{k-p} = 0 \\ \\ \frac{\partial \mathcal{L}}{\partial e_k} & = & \gamma e_k - \alpha_{k-p} - \sum_{i=1}^{p} \alpha_{k-p+i} \frac{\partial}{\partial e_{k-i}} [w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p})] = 0, \;\; k = p+1, ..., N \\ \\ \frac{\partial \mathcal{L}}{\partial \alpha_{k-p}} & = & y_k - e_k - w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) - b = 0, \;\; k = p+1, ..., N+p. \end{array} \right. \tag{23}$$

By replacing w into the last two conditions and applying Mercer's condition by letting $\mathcal{X}_i$ in (1) correspond to $z_{k-1|k-p} = x_{k-1|k-p} - \xi_{k-1|k-p}$:

$$K(z_{k-1|k-p}, z_{l-1|l-p}) = \varphi(z_{k-1|k-p})^T \varphi(z_{l-1|l-p}) \tag{24}$$

one obtains the following conditions for optimality

$$\left\{\begin{array}{l} \sum_{k=p+1}^{N+p} \alpha_{k-p} = 0 \\ \\ \gamma e_k - \alpha_{k-p} - \sum_{i=1}^{p} \alpha_{k-p+i} \frac{\partial}{\partial e_{k-i}} [\sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{k-1|k-p}, z_{l-1|l-p})] = 0, \quad k = p+1, ..., N \\ \\ y_k - e_k - \sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{k-1|k-p}, z_{l-1|l-p}) - b = 0, \quad k = p+1, ..., N+p. \end{array}\right. \tag{25}$$

Hence (23) which is of the form $F_1(w, b, e, \alpha) = 0$ has been represented as $F_2(b, e, \alpha) = 0$ in (25) by elimination of w. This is similar to the elimination of w in (11) for the static SVM case where w itself is never explicitly calculated. Furthermore the Mercer condition (24) is implicitly defining the nonlinear mapping $\varphi(\cdot)$.

Finding a solution to the equations (25) is computationally very expensive. Therefore, we further consider the case $\gamma \to \infty$ which corresponds to

$$\min_{\alpha, e, b} \frac{1}{2} \sum_{k=p+1}^{N+p} e_k^2 \tag{26}$$

subject to

$$\left\{\begin{array}{l} y_k - e_k = \sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{l-1|l-p}, z_{k-1|k-p}) + b, \quad k = p+1, ..., N+p \\ \\ \sum_{k=p+1}^{N+p} \alpha_{k-p} = 0. \end{array}\right. \tag{27}$$

At this point it is important to note that the vectors z depend on the error variables e. Hence the cost function is subject to a set of nonlinear constraints in e. Hence although (26)(27) might seem similar to Widrow's Adaline at first sight, due to the least squares norm, the recurrent LS-SVM is fundamentally different.

The resulting recurrent simulation model is given by

$$\hat{y}_k = \sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{l-1|l-p}, \begin{bmatrix} \hat{y}_{k-1} \\ \hat{y}_{k-2} \\ \vdots \\ \hat{y}_{k-p} \end{bmatrix}) + b \tag{28}$$

with given initial condition $\hat{y}_i = y_i$ for $i = 1, 2, .., p$. For RBF kernels one employs

$$K(z_{k-1|k-p}, z_{l-1|l-p}) = \exp(-\nu \, \|z_{k-1|k-p} - z_{l-1|l-p}\|_2^2) \tag{29}$$

where ν is a positive real constant.

No centers have to be determined in SVM models, in contrast with most of the classical RBF approaches [2, 13, 15]. For the recurrent SVM case the parameter estimation problem becomes non-convex. The constrained nonlinear optimization problem (26)(27) can be solved e.g. by sequential quadratic programming (SQP) [7]. For large data sets special methods for large scale nonlinear optimization have to be applied. Because the results are based on the limit case $\gamma \to \infty$ and hence partially neglecting the regularization term $\frac{1}{2}w^T w$ in (26) (the form of the solution is still derived by taking into account a regularization term based upon (20)) there is a danger for overfitting. Therefore, instead of minimizing the cost function to its local minimum one has to apply early stopping, which is equivalent to a form of regularization [1, 3, 16]. In standard SVM theory, which is applicable to feedforward models, the parameter ν in (29) could be determined by minimizing upper bounds on the generalization error. However, this theory doesn't apply to the recurrent SVM case because the input arguments z of $K(\cdot, \cdot)$ are not independent from each other. Hence ν should be either chosen as part of the unknown parameter vector for the optimization (26)(27) or ad hoc.

4 Example: double scroll trajectory learning

In this example we consider Chua's circuit [4, 5, 10]

$$\left\{ \begin{array}{rcl} \dot{x} & = & \alpha \, [y - h(x)] \\ \dot{y} & = & x - y + z \\ \dot{z} & = & -\beta y \end{array} \right. \tag{30}$$

with piecewise linear characteristic

$$h(x) = m_1 x + \frac{1}{2}(m_0 - m_1)\,(|x+1| - |x-1|). \tag{31}$$

A double scroll attractor is generated by taking $\alpha = 9$, $\beta = 14.286$, $m_0 = -1/7$, $m_1 = 2/7$. A trajectory has been generated for initial condition $[0.1; 0; -0.1]$ by using a Runge-Kutta integration rule (ode23 in Matlab).

In Fig.1 the first $N = 300$ data points were used for trajectory learning of the double scroll by a recurrent SVM with RBF kernel. For the model structure $p = 12$ has been taken and $\nu = 1$ for the RBF kernel. In order to solve the constrained nonlinear optimization problem (26)(27), sequential quadratic programming (SQP) has been applied (constr in Matlab with a specification of the number of equality constraints). The model (28) has been simulated in C by using Matlab's cmex facility. In all simulations, for the initial unknown parameter vector α_k, e_k have been chosen randomly according to a Gaussian distribution with zero mean and standard deviation 0.1 and $b = 0$. Although no further optimization of the model structure has been done, it has been observed that small values of p (that are chosen in accordance with Takens' embedding theorem) are slowing down the training process. The value of ν in (29) has to be chosen relative to the scaling of the data set. On Fig.1 the simulation result after training is shown in dashed line based on (28)(29) with as initial condition data points $k = 1$ to 12. Early stopping has been decided based upon the given training data (not on an additional test set as is usual for feedforward models) by evaluating the cost function in (26) for the simulation obtained from (28) with $p = 12$ and $N = 300$ for the given initial state. Early stopping was applied at the moment when this performance index degraded which occurred after about 400 iteration steps of SQP for this example. The resulting recurrent SVM is generalizing to a double scroll attractor, also for small perturbations on the initial state. Fig.2 shows a further optimization to the local minimum with the result of Fig.1 as initial parameter vector, which leads to overfitting. The overfitting phenomenon can be detected on the training data itself as illustrated on Fig.2. Finally note that for the recurrent SVM's it is important to simulate the form (28) because when solving the constrained nonlinear optimization problem (26)(27) the constraints will usually not hold exactly but with a certain tolerance. For chaotic systems such small perturbation could cause significant differences between the solution vector to (26)(27) and the recurrent simulation model (28).

5 Conclusions

In this paper we introduced recurrent least squares support vector machines, in addition to existing SVM solutions for classification and static nonlinear function estimation problems. Some essential features of support vector machines as Mercer's condition and the fact that the output weights are a Lagrange multiplier weighted sum of the data points, are still applicable. For computational reasons we focused on the limit case of assigning the squared error an infinitely large penalty factor. The training has been formulated as a non-convex constrained nonlinear optimization problem in the error variables and the Lagrange multipliers. Training recurrent neural networks by dynamic backpropagation to follow trajectories of chaotic systems is well-known to be a hard problem. By trajectory learning of a double scroll we illustrated that recurrent least squares SVM's can generalize well, even on relatively small given training data sets. For large data sets efficient methods for large scale constrained nonlinear optimization have to be used. The work of recurrent SVM's opens new perspectives with respect to time-series prediction and nonlinear modeling in general.

References

[1] Bishop C.M., *Neural networks for pattern recognition*, Oxford University Press, 1995.

[2] Chen S., Cowan C., Grant P., "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Transactions on Neural Networks*, Vol.2, No.2, pp.302-309, 1991.

[3] Cherkassky V., Mulier F., *Learning from data: concepts, theory and methods*, John Wiley and Sons, 1998.

[4] Chua L.O., Komuro M., Matsumoto T., "The Double Scroll Family," *IEEE Trans. Circuits and Systems-I*, **33**(11), 1072-1118, 1986.

[5] Chua L.O., "Chua's Circuit: an overview ten years later," *Journal of Circuits, Systems, and Computers*, **4**(2), 117-159, 1994.

[6] Cortes C., Vapnik V., "Support vector networks," *Machine Learning*, Vol.20, pp.273-297, 1995.

[7] Fletcher R., *Practical methods of optimization*, Chichester and New York: John Wiley and Sons, 1987.

[8] Golub G.H., Van Loan C.F., *Matrix Computations*, Baltimore MD: Johns Hopkins University Press, 1989.

[9] Haykin S., *Neural Networks: a Comprehensive Foundation*, Macmillan College Publishing Company: Englewood Cliffs, 1994.

[10] Madan R.N. (Guest Editor), *Chua's Circuit: A Paradigm for Chaos*, Signapore: World Scientific Publishing Co. Pte. Ltd, 1993.

[11] Narendra K.S., Parthasarathy K., "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, Vol.1, No.1, pp.4-27, 1990.

[12] Narendra K.S., Parthasarathy K., "Gradient methods for the optimization of dynamical systems containing neural networks," *IEEE Transactions on Neural Networks*, Vol.2, No.2, pp.252-262, 1991.

[13] Poggio T., Girosi F., "Networks for approximation and learning," *Proceedings of the IEEE*, Vol.78, No.9, pp.1481-1497, 1990.

[14] Saunders C., Gammerman A., Vovk V., "Ridge Regression Learning Algorithm in Dual Variables," Proceedings of the 15th International Conference on Machine Learning ICML-98, Madison-Wisconsin, 1998.

[15] Schölkopf B., Sung K.-K., Burges C., Girosi F., Niyogi P., Poggio T., Vapnik V., "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," *IEEE Transactions on Signal Processing,* Vol.45, No.11, pp.2758-2765, 1997.

[16] Sjöberg J., Zhang Q., Ljung L., Benveniste A., Delyon B., Glorennec P., Hjalmarsson H., Juditsky A., "Nonlinear black-box modelling in system identification : a unified overview," *Automatica*, 31(12): 1691-1724, 1995.

[17] Smola A., Schölkopf B., "On a kernel-based method for pattern recognition, regression, approximation and operator inversion," *Algorithmica*, Vol.22, pp.211-231, 1998.

[18] Smola A., Schölkopf B., Müller K.-R., "The connection between regularization operators and support vector kernels," *Neural Networks*, Vol.11, No.4, 1998.

[19] Smola A., *Learning with Kernels*, PhD Thesis, published by: GMD, Birlinghoven, 1999.

[20] Suykens J.A.K., Vandewalle J., De Moor B., *Artificial neural networks for modelling and control of non-linear systems*, Kluwer Academic Publishers: Boston, 1996.

[21] Suykens J.A.K., Vandewalle J., "Learning a simple recurrent neural state space model to behave like Chua's double scroll," *IEEE Transactions on Circuits and Systems-I*, Vol.42, No.8, pp.499-502, 1995.

[22] Suykens J.A.K., Vandewalle J. (Eds.), *Nonlinear Modeling: advanced black-box techniques*, Kluwer Academic Publishers: Boston, 1998.

[23] Suykens J.A.K., Vandewalle J., “Least squares support vector machine classifiers,” *Neural Processing Letters*, Vol.9, No.3, pp.293-300, June 1999.

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

[25] Vapnik V., *Statistical learning theory,* John Wiley, New-York, 1998.

[26] Vapnik V., Golowich S., Smola A., “Support vector method for function approximation, regression estimation and signal processing”, *In Advances in Neural Information Processing Systems, 9*, MIT Press, 1997.

[27] Vapnik V., “The support vector method of function estimation,” In *Nonlinear Modeling: advanced black-box techniques*, Suykens J.A.K., Vandewalle J. (Eds.), Kluwer Academic Publishers, Boston, pp.55-85, 1998.

[28] Werbos P., “Backpropagation through time: what it does and how to do it,” *Proceedings of the IEEE*, 78 (10), pp.1150-1560, 1990.

[29] Zurada J.M., *Introduction to Artificial Neural Systems*, West Publishing Company, 1992.

Captions of Figures

Figure 1: Trajectory learning of the double scroll (full line) by a least squares SVM with RBF kernel. The simulation result after training (dashed line) on $N = 300$ data points is shown with as initial condition data points $k = 1$ to 12. For the model structure $p = 12$ and $\nu = 1$ is taken. Early stopping is done in order to avoid overfitting.

Figure 2: Overfitting of the data obtained by taking the result of Fig.1 as initial parameter vector for a further optimization to the local minimum.

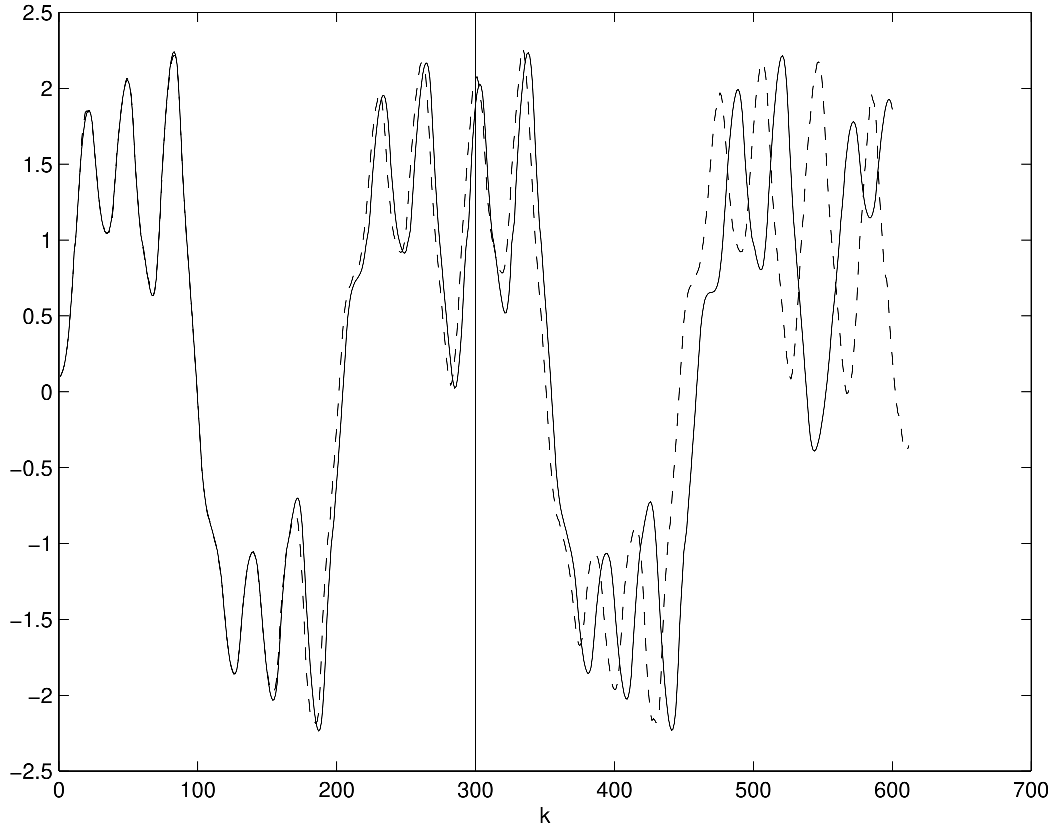

Fig. 1.

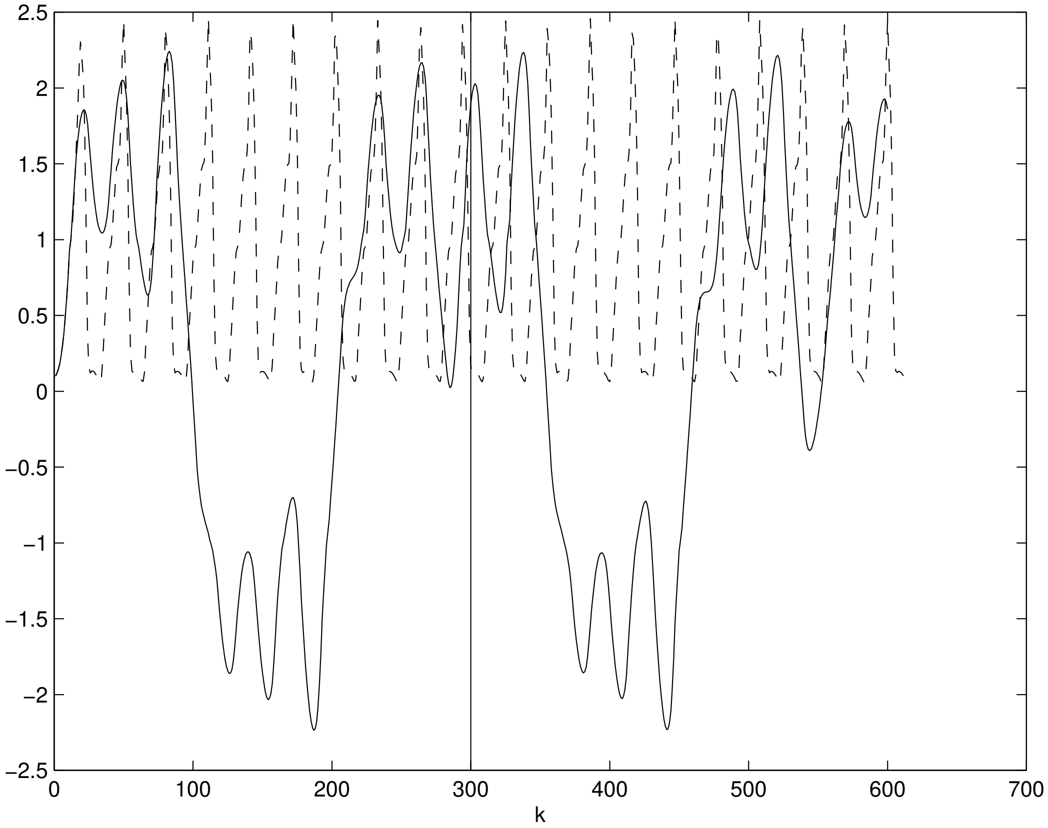

Fig. 2.