Support Vector Machines: a Nonlinear Modelling and Control Perspective *

Johan A.K. Suykens

Katholieke Universiteit Leuven
Department of Electrical Engineering, ESAT-SISTA
Kasteelpark Arenberg 10, B-3001 Leuven (Heverlee), Belgium
Tel: 32/16/32 18 02 Fax: 32/16/32 19 70
Email: johan.suykens@esat.kuleuven.ac.be

April 2001

*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 Belgian Program 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 MEFISTO of the Flemish Community. Johan Suykens is a postdoctoral researcher with the National Fund for Scientific Research FWO - Flanders.

Abstract

In recent years neural networks as multilayer perceptrons and radial basis function networks have been frequently used in a wide range of fields, including control theory, signal processing and nonlinear modelling. A promising new methodology is Support Vector Machines (SVM), which has been originally introduced by Vapnik within the area of statistical learning theory and structural risk minimization. SVM approaches to classification, nonlinear function and density estimation lead to convex optimization problems, typically quadratic programming. However, due to their non-parametric nature, the present SVM methods were basically restricted to static problems. We discuss a method of least squares support vector machines (LS-SVM), which has been extended to recurrent models and use in optimal control problems. We explain how robust nonlinear estimation and sparse approximation can be done by means of this kernel based technique. A short overview of hyperparameter tuning methods is given. SVM methods are able to learn and generalize well in large dimensional input spaces and have outperformed many existing methods on benchmark data sets. Its full potential in a dynamical systems and control context remains to be explored.

Keywords. Statistical learning theory, support vector machines, kernel based methods, robust estimation, ridge regression, regularization, feedforward and recurrent neural networks, neural control.

1 Introduction

In the last decade, neural networks have proven to be a powerful methodology in many areas [1, 5, 13, 27, 33, 34]. Although initially neural nets were often presented as a kind of miracle approach, reliable training methods exist nowadays mainly thanks to interdisciplinary studies and insights from several fields including statistics, circuit-, systems and control theory, signal processing, information theory, physics and others. Despite many of these advances, there still remain a number of weak points for these classical neural net approaches, namely the existence of many local minima solutions and the issue of how to choose the number of hidden units (Fig.1). Major breakthroughs have been obtained at this point by means of a new class of neural networks called Support Vector Machines (SVMs), originally introduced by Vapnik within the area of statistical learning theory and structural risk minimization [45, 46, 49]. SVM solutions are characterized by convex optimization problems. One starts formulating the problem in a primal weight space, where one can work in a so-called infinite dimensional feature space, but one solves the resulting problem (usually a quadratic programming (QP) problem) in the dual space. The model complexity (e.g. number of hidden units) follows from solving this convex optimization problem due to the fact that the obtained solution is sparse. SVM models also scale very well to high dimensional input spaces. They are less black-box than multilayer perceptrons (MLP) in the sense that the support vectors (which constitute the model) correspond to informative data points.

Presently, there is a lot of interest in this new class of kernel based techniques [7, 25] especially in the areas of neural networks and machine learning. Taking into account the fact that SVM methods have outperformed many existing methods on several classification and nonlinear function estimation problems, this can be relevant for certain problems studied in the area of control theory as well. However, due to the fact that SVMs are basically non-parametric techniques, its use in a dynamical systems and control context is more complicated. While for standard MLPs one can think of model structures and/or controllers parameterized by such architectures, this is no longer true for SVM methods. This also explains why standard SVM methods have been developed mainly for static problems like classification, static nonlinear function and density estimation. However, recently a method of Least Squares SVMs (LS-SVM) has been investigated [36] which has been extended to dynamical problems of recurrent neural networks [39] and use in optimal control [40].

In this paper we give first a short introduction to standard support vector machines for classification and nonlinear function estimation, as originally introduced by Vapnik. Then we will focus on LS-SVMs for classification and nonlinear function estimation problems [23, 36]. When no bias term is used, the LS-SVM system takes the same form in dual space as regularization networks [8, 19] and Gaussian processes [7, 51] (Fig.2). In regularization theory one takes a viewpoint from functional analysis and in Gaussian processes one immediately formulates problems in the dual space. On the other hand, the fact that LS-SVMs have clear and explicit primal-dual formulations has a number of advantages. Concerning classification problems one has regression interpretations and direct links to work in statistics. Furthermore, it is easier to formulate modified versions such as weighted least squares for robust nonlinear estimation and establish links with convex optimization theory such as interior point methods. On the other hand, the use of a least squares cost function has potential drawbacks like the lack of sparseness in the solution vector and optimality under Gaussian assumptions. However, one can overcome these two problems [41]. Because of the primal-dual LS-SVM formulation one still has a notion of support values. In standard SVMs from quadratic programming, many elements in the solution vector will be exactly equal to zero (the non-zero values correspond to data points which are called support vectors). For LS-SVMs one rather has a spectrum of sorted support values. The physical meaning of these parameters can nevertheless be further employed in order to prune the model. While pruning of classical multilayer perceptrons (optimal brain damage, optimal brain surgeon) [1, 12, 15] involves computing a Hessian matrix or its inverse, in LS-SVMs all necessary pruning information follows from the solution vector itself. The second potential drawback is circumvented by applying a weighted least squares version. In this way, the solution can be easily robustified in order to cope with outliers and non-Gaussian error distributions with heavy tails, by incorporated ideas from robust statistics [14, 21] into LS-SVMs. An important issue for SVMs (and modelling in general) is model selection. For nonlinear modelling one often employs an RBF kernel. In this case, the kernel width and regularization parameter need to be selected. At this point one has several possible choices, ranging from VC bounds in statistical learning theory to cross-validation, bootstrap and Bayesian methods.

This paper is organized as follows. In Section 2 we review Vapnik's SVM classifier. In Section 3 we discuss the basic LS-SVM classifier formulation. In Section 4 SVMs for nonlinear function estimation and weighted LS-SVMs for robust estimation are explained. In Section 5 we give a short overview of possible hyperparameter selection methods. In Section 6 we discuss sparseness issues. In Section 7 a recurrent LS-SVM extension is presented. Finally, in Section 8 LS-SVMs for optimal control are explained.

2 Vapnik's SVM classifier

In this Section we review some basic ideas of support vector machines. For further details about SVMs for classification and nonlinear function estimation we refer to [6, 7, 24, 25, 28, 29, 30, 45, 46, 47].

Given a training set $\{x_k, y_k\}_{k=1}^N$ with input data $x_k \in \mathbb{R}^n$ and corresponding binary class labels $y_k \in \{-1, +1\}$, the SVM classifier formulation starts from the following assumption:

$$\begin{cases} w^T \varphi(x_k) + b \geq +1 \quad , & \text{if} \quad y_k = +1 \\ w^T \varphi(x_k) + b \leq -1 \quad , & \text{if} \quad y_k = -1 \end{cases} \tag{1}$$

which is equivalent to

$$y_k[w^T \varphi(x_k) + b] \geq 1, \qquad k = 1, ..., N. \tag{2}$$

Here the nonlinear function $\varphi(\cdot) : \mathbb{R}^n \to \mathbb{R}^{n_h}$ maps the input space to a so-called higher dimensional feature space (Fig.4). It is important to note that the dimension n_h of this space is only defined in an implicit way and it can be infinite dimensional. The term b denotes a bias term. In primal weight space the classifier takes the form

$$y(x) = \text{sign}[w^T \varphi(x) + b], \tag{3}$$

but, on the other hand, is never evaluated in this form.

One defines the optimization problem:

$$\min_{w,b,\xi} \mathcal{J}(w, \xi) = \frac{1}{2} w^T w + c \sum_{k=1}^N \xi_k \tag{4}$$

subject to

$$\begin{cases} y_k[w^T \varphi(x_k) + b] \geq 1 - \xi_k, \qquad k = 1, ..., N \\ \qquad \xi_k \geq 0, \qquad k = 1, ..., N. \end{cases} \tag{5}$$

The variables ξ_k are slack variables which are needed in order to allow misclassifications in the set of inequalities (e.g. due to overlapping distributions). The minimization of $\|w\|_2$

corresponds to a maximization of the margin between the two classes (Fig.3). c is a positive real constant and should be considered as a tuning parameter in the algorithm.

The Lagrangian for this problem is given by

$$\mathcal{L}(w,b,\xi;\alpha,\nu) = \mathcal{J}(w,\xi) - \sum_{k=1}^{N} \alpha_k \{y_k [w^T \varphi(x_k) + b] - 1 + \xi_k\} - \sum_{k=1}^{N} \nu_k \xi_k \tag{6}$$

with Lagrange multipliers $\alpha_k \geq 0$, $\nu_k \geq 0$ $(k = 1, ..., N)$. It is well-known from optimization theory [9] that the solution is characterized by the saddle point of the Lagrangian:

$$\max_{\alpha,\nu} \min_{w,b,\xi} \mathcal{L}(w,b,\xi;\alpha,\nu). \tag{7}$$

One obtains

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \;\; \rightarrow \;\; w = \sum_{k=1}^{N} \alpha_k y_k \varphi(x_k) \\ \frac{\partial \mathcal{L}}{\partial b} = 0 \;\; \rightarrow \;\; \sum_{k=1}^{N} \alpha_k y_k = 0 \\ \frac{\partial \mathcal{L}}{\partial \xi_k} = 0 \;\; \rightarrow \;\; 0 \leq \alpha_k \leq c, \;\; k = 1, ..., N. \end{cases} \tag{8}$$

By replacing w in the Lagrangian, one obtains the following dual problem (in the Lagrange multipliers α), which is the quadratic programming problem:

$$\max_{\alpha} \mathcal{Q}(\alpha) = -\frac{1}{2} \sum_{k,l=1}^{N} y_k y_l \, K(x_k, x_l) \, \alpha_k \alpha_l + \sum_{k=1}^{N} \alpha_k \tag{9}$$

such that

$$\begin{cases} \sum_{k=1}^{N} \alpha_k y_k = 0 \\ \\ 0 \leq \alpha_k \leq c, \;\; k = 1, ..., N. \end{cases} \tag{10}$$

Motivated by the Mercer condition one can choose a kernel

$$K(x_k, x_l) = \varphi(x_k)^T \, \varphi(x_l) \tag{11}$$

such that there is no need to compute w and $\varphi(x_k)$ (which can be infinite dimensional).

Finally, in dual space the nonlinear SVM classifier becomes

$$y(x) = \mathrm{sign}[\sum_{k=1}^{N} \alpha_k \, y_k \, K(x, x_k) + b] \tag{12}$$

with α_k positive real constants and b a real constant that follow from the QP problem. The non-zero Lagrange multipliers α_k are called support values. The corresponding data points are called support vectors and are located close to the decision boundary. These are the data points that contribute to the classifier model. The bias term b follows from the KKT conditions which is not further discussed here.

Several choices for the kernel $K(\cdot,\cdot)$ are possible:

- $K(x,x_k) = x_k^T x$ (linear SVM)
- $K(x,x_k) = (x_k^T x + 1)^d$ (polynomial SVM of degree d)
- $K(x,x_k) = \exp\{-\|x - x_k\|_2^2/\sigma^2\}$ (RBF kernel)
- $K(x,x_k) = \tanh(\kappa\, x_k^T x + \theta)$ (MLP SVM) .

The Mercer condition holds for all σ values in the RBF case, but not for all possible choices of κ, θ in the MLP case (hence MLP kernels are less interesting). In the case of RBF or MLP kernels, the number of hidden units corresponds to the number of support vectors. Eventually, one might also train parametric models using SVM methodology. This has been illustrated in [35] on the training of MLP classifiers by minimizing an upper bound on the VC dimension.

Note that the size of the matrix in the QP problem is directly proportional to the number of training data, which is an important drawback in the case of large given data sets. Interior point methods (Vanderbei, Smola) have been applied to solve moderate size problems. For larger scale problems so-called chunking procedures (Osuna) have been studied where the problem is decomposed into smaller subproblems. Platt's SMO (Sequential Minimal Optimization) is an extreme form of chunking. Another large scale method is successive overrelaxation (SOR) (Mangasarian), which has been applied to massive data sets.

3 Least Squares Support Vector Machines

In [36] Vapnik's SVM classifier formulation has been modified as follows:

$$\min_{w,b,e} \mathcal{J}(w,e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=1}^{N} e_k^2 \tag{13}$$

subject to the equality constraints

$$y_k \left[w^T \varphi(x_k) + b \right] = 1 - e_k,\ \ k = 1, ..., N. \tag{14}$$

Important differences with standard SVMs are the equality constraints and the sum squared error term, which greatly simplifies the problem.

The solution is obtained after constructing the Lagrangian:

$$\mathcal{L}(w,b,e;\alpha) = \mathcal{J}(w,b,e) - \sum_{k=1}^{N} \alpha_k \{y_k[w^T \varphi(x_k) + b] - 1 + e_k\} \tag{15}$$

where α_k are the Lagrange multipliers. The conditions for optimality are given by:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 & \rightarrow \quad w = \sum_{k=1}^{N} \alpha_k y_k \varphi(x_k) \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \rightarrow \quad \sum_{k=1}^{N} \alpha_k y_k = 0 \\ \frac{\partial \mathcal{L}}{\partial e_k} = 0 & \rightarrow \quad \alpha_k = \gamma e_k, \qquad\qquad\qquad\qquad k = 1, ..., N \\ \frac{\partial \mathcal{L}}{\partial \alpha_k} = 0 & \rightarrow \quad y_k[w^T\varphi(x_k) + b] - 1 + e_k = 0, \quad k = 1, ..., N. \end{cases} \tag{16}$$

Defining $Z = [\varphi(x_1)^T y_1; ...; \varphi(x_N)^T y_N]$, $Y = [y_1; ...; y_N]$, $\vec{1} = [1; ...; 1]$, $e = [e_1; ...; e_N]$, $\alpha = [\alpha_1; ...; \alpha_N]$ and eliminating w, e, one obtains

$$\left[\begin{array}{c|c} 0 & Y^T \\ \hline Y & \Omega + \gamma^{-1} I \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline \vec{1} \end{array}\right] \tag{17}$$

where $\Omega = ZZ^T$ and Mercer's condition is applied within the Ω matrix

$$\begin{aligned} \Omega_{kl} &= y_k y_l \, \varphi(x_k)^T \varphi(x_l) \\ &= y_k y_l \, K(x_k, x_l). \end{aligned} \tag{18}$$

LS-SVM classifiers with RBF kernel are able to realize strongly nonlinear decision boundaries such as occuring in two-spiral classification problems (Fig.5), where multilayer perceptrons typically have difficulties with learning and generalization [20, 36, 37]. For large data sets the system has to be solved in an iterative way. In order to apply iterative methods such as the conjugate gradient method the matrix involved in this set of linear equations should be positive definite. A transformation into a positive definite matrix has been discussed in [37]. When applying iterative methods, the matrices are not stored. The convergence of the conjugate gradient algorithm depends on the condition number of the matrix, hence it is influenced by the regularization parameter γ.

4 SVM for nonlinear function estimation

In this Section we discuss the standard SVM and LS-SVMs for nonlinear function estimation, together with a weighted LS-SVM version for robust estimation.

4.1 Vapnik's formulation

Consider a given training set $\{x_k, y_k\}_{k=1}^N$ with input data $x_k \in \mathbb{R}^n$ and output data $y_k \in \mathbb{R}$. The following model is taken

$$f(x) = w^T \varphi(x) + b \tag{19}$$

where the input data are projected to a higher dimensional feature space as in the classifier case. In empirical risk minimization one optimizes the cost function (Fig.6)

$$R_{emp} = \frac{1}{N} \sum_{k=1}^{N} |y_k - w^T \varphi(x_k) - b|_\epsilon, \tag{20}$$

containing Vapnik's ϵ-insensitive loss function which is defined as follows:

$$|y - f(x)|_\epsilon = \begin{cases} 0 & , \ \text{if} \ |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon & , \ \text{otherwise.} \end{cases} \tag{21}$$

This leads to the optimization problem

$$\min_{w,b,\xi,\xi^*} \ J(w, \xi, \xi^*) = \frac{1}{2} w^T w + c \sum_{k=1}^{N} (\xi_k + \xi_k^*) \tag{22}$$

subject to

$$\begin{cases} y_k - w^T \varphi(x_k) - b & \leq & \epsilon + \xi_k \\ w^T \varphi(x_k) + b - y_k & \leq & \epsilon + \xi_k^* \\ \xi_k, \xi_k^* & \geq & 0. \end{cases} \tag{23}$$

Here ϵ is the accuracy that one demands for the approximation, which can be violated by means of the slack variables ξ, ξ^*. The conditions for optimality yield the following dual problem in the Lagrange multipliers α, α_*:

$$\begin{aligned} \max_{\alpha,\alpha_*} \mathcal{Q}(\alpha, \alpha_*) = & -\frac{1}{2} \sum_{k,l=1}^{N} (\alpha_k - \alpha_k^*)(\alpha_l - \alpha_l^*) \, K(x_k, x_l) \\ & -\epsilon \sum_{k=1}^{N} (\alpha_k + \alpha_k^*) + \sum_{k=1}^{N} y_k (\alpha_k - \alpha_k^*) \end{aligned} \tag{24}$$

subject to

$$\begin{cases} \sum_{k=1}^{N}(\alpha_k - \alpha_k^*) & = & 0 \\ \alpha_k, \alpha_k^* & \in & [0, c]. \end{cases} \tag{25}$$

The kernel trick $K(x_k, x_l) = \varphi(x_k)^T \varphi(x_l)$ is again applied within the formulation of this quadratic programming problem. Finally, the SVM for nonlinear function estimation takes the form:

$$f(x) = \sum_{k=1}^{N}(\alpha_k - \alpha_k^*) K(x, x_k) + b. \tag{26}$$

4.2 LS-SVM for nonlinear function estimation

The LS-SVM model for function estimation has the following representation in feature space

$$y(x) = w^T \varphi(x) + b \tag{27}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}$. The use of the nonlinear mapping $\varphi(\cdot)$ is similar to the classifier case.

Given a training set $\{x_k, y_k\}_{k=1}^N$ one defines now the optimization problem

$$\min_{w,b,e} \mathcal{J}(w, e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=1}^{N} e_k^2 \tag{28}$$

subject to equality constraints

$$y_k = w^T \varphi(x_k) + b + e_k, \quad k = 1, ..., N. \tag{29}$$

The cost function with 2-norm and regularization corresponds to a form of ridge regression [11]. A similar problem has been studied in [23] but without considering a bias term.

One constructs the Lagrangian

$$\mathcal{L}(w, b, e; \alpha) = \mathcal{J}(w, e) - \sum_{k=1}^{N} \alpha_k \{w^T \varphi(x_k) + b + e_k - y_k\} \tag{30}$$

where α_k are Lagrange multipliers. The conditions for optimality are given by

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 & \rightarrow & w = \sum_{k=1}^{N} \alpha_k \varphi(x_k) & \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \rightarrow & \sum_{k=1}^{N} \alpha_k = 0 & \\ \frac{\partial \mathcal{L}}{\partial e_k} = 0 & \rightarrow & \alpha_k = \gamma e_k, & k = 1, ..., N \\ \frac{\partial \mathcal{L}}{\partial \alpha_k} = 0 & \rightarrow & w^T \varphi(x_k) + b + e_k - y_k = 0, & k = 1, ..., N \end{cases} \tag{31}$$

with solution

$$\left[\begin{array}{c|c} 0 & \vec{1}^T \\ \hline \vec{1} & \Omega + \gamma^{-1} I \end{array}\right] \left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right] \tag{32}$$

with $y = [y_1; ...; y_N]$, $\vec{1} = [1; ...; 1]$, $\alpha = [\alpha_1; ...; \alpha_N]$. From application of the Mercer condition one obtains

$$\begin{aligned} \Omega_{kl} &= \varphi(x_k)^T \varphi(x_l), \qquad k, l = 1, ..., N \\ &= K(x_k, x_l). \end{aligned} \tag{33}$$

The resulting LS-SVM model for function estimation becomes

$$y(x) = \sum_{k=1}^{N} \alpha_k K(x, x_k) + b \tag{34}$$

where α_k, b are the solution to the linear system. The large scale algorithm as outlined for LS-SVM classifiers can be applied in a similar way to the function estimation case. Note that in the case of RBF kernels, one has only two additional tuning parameters γ, σ, which is less than for standard SVMs.

4.3 Robust estimation and weighted LS-SVM

In order to obtain a robust estimate based upon the previous LS-SVM solution, one can weight the error variables in a subsequent step by using extra weighting factors [41]. This leads to the optimization problem:

$$\min_{w^\star, b^\star, e^\star} J(w^\star, e^\star) = \frac{1}{2} w^{\star T} w^\star + \frac{1}{2} \gamma \sum_{k=1}^{N} v_k \, e_k^{\star 2} \tag{35}$$

such that

$$y_k = w^{\star T} \varphi(x_k) + b^\star + e_k^\star, \qquad k = 1, ..., N.$$

A similar derivation as in the previous subsection can be made. In order to produce a robust estimate [14, 21], the weighting factors v_k can be chosen as a function of the error distribution e_k of the unweighted LS-SVM. In the second step one applies then the weighted LS-SVM. This can be repeated within an iterative procedure. In standard SVM approaches one usually chooses a certain cost function (any convex cost function can be taken according to [30]). However, in such a top-down approach one assumes in fact that the noise distribution is known because one knowns which cost function is optimal for a certain given noise distribution. The weighted LS-SVM procedure tries to work bottom-up and aims at tuning an optimal underlying cost function for the given noisy data. The methods can be conceptually linked by the fact that when one applies an interior point method for solving a standard SVM problem, at every iteration step a KKT system is solved which is similar to solving one single LS-SVM. Hence, solving a standard SVM with arbitrary convex cost function implicitly corresponds to solving a sequence of LS-SVMs. Some simulation results of weighted LS-SVM on a noisy sinc function with outliers and heavy tailed non-Gaussian distributions are shown in Fig.7-8.

5 Hyperparameter selection methods

In the case when one takes e.g. an RBF kernel for LS-SVM, one still has to determine an optimal choice of the kernel width σ and the regularization parameter γ. There are several possibilities:

- *Cross-validation methods:*

  A simple method is to define a training, validation and test set. One tries several (γ, σ) combinations by determining models with support values α from the training data and selecting the (γ, σ) combination which results into a minimal error on the validation set. Test sets are independent fresh data which are completely left untouched during identification of the model. A better but computationally heavier procedure is to apply cross-validation methods or bootstrapping. For larger data sets one typically prefers 10-fold cross-validation in order to select (γ, σ). In [42] the latter method was successfully applied to many binary and multiclass UCI benchmark data sets in comparison with many methods reported in the literature.

- *VC bounds, statistical learning theory:*

  SVMs have been originally introduced within statistical learning theory [45, 46, 49]. In this theory one derives bounds on the generalization error of the model, which doesn't require a specific form of the underlying distribution of the data (i.i.d. assumption). The bounds are expressed in terms of the VC (Vapnik-Chervonenkis) dimension, which is considered to be a combinatorial measure for model complexity that characterizes the capacity of a chosen class of functions. For SVM models, an upper bound on this VC dimension can be computed by solving an additional quadratic programming problem. The hyperparameters can be chosen then in such a way that the upper bound on the VC dimension is minimized. Despite the fact that this upper bound can be conservative, it is nevertheless often indicative. Several efforts have been made in order to sharpen these bounds.

- *Bayesian learning methods:*

  Bayesian learning methods have been successful for the training and understanding of classical neural networks [1, 16]. In [43] a Bayesian framework has been developed for LS-SVMs with three levels of inference. At the first level of inference one considers a probability distribution on w (in a potentially infinite dimensional space) where the prior corresponds to the regularization term $w^T w$ and the least squares cost function to the likelihood. This interpretation allows for probabilistic interpretations of the LS-SVM. At the second level of inference one infers the hyperparameters (in this case γ). Automatic hyperparameter selection can be done based upon formulas which depend on the effective number of parameters and the eigenvalues of the Gram matrix. The consideration of the bias term b complicates the problem in comparison with e.g. Gaussian processes. At the third level of inference one obtains model comparison criteria by computation of the Occam factor. The kernel width σ is selected at this level. This framework also allows to perform automatic input selection [44].

6 Sparse approximation by LS-SVM

A drawback of LS-SVMs, at first sight, is that sparseness is lost due to the choice of the 2-norm, which is also clear from the fact that $\alpha_k = \gamma e_k$. In the case of Vapnik's formulation one typically has many α_k values which are exactly equal to zero. In [38] we have shown how sparseness can be imposed to LS-SVMs by a pruning procedure which is based upon the sorted support value spectrum. This is important in view of an equivalence between SVMs and sparse approximation, shown in [10]. An important difference with pruning methods in classical neural networks [1, 12, 15], e.g. optimal brain damage and optimal brain surgeon, is that the LS-SVM pruning procedure requires no computation of a Hessian matrix. One immediately gets all necessary pruning information from the solution to the linear system itself. Hence, by plotting the spectrum of the sorted $|\alpha_k|$ values one can evaluate which data are most significant for contribution to the LS-SVM (Fig.9). Sparseness is imposed then by gradually omitting the least important data from the training set and re-estimating the LS-SVM.

The algorithm for sparse approximation using LS-SVMs looks as follows:

1. Train LS-SVM based on N points.

2. Remove a small amount of points (e.g. 5% of the set) with smallest values in the sorted $|\alpha_k|$ spectrum.

3. Re-train the LS-SVM based on the reduced training set.

4. Go to 2, unless the user-defined performance index degrades. If the performance becomes worse, one checks whether an additional modification of (γ, σ) might improve the performance.

This procedure is applicable both for classification and function estimation (Fig.10). Note that omitting data points implicitly corresponds to creating an ϵ-insensitive zone in the underlying cost function which leads to sparseness. Links between standard SVM and LS-SVM can be established for any convex cost function [30] though interior point algorithms, being aware of the fact that in every iteration step one solves a KKT system of the same form as an LS-SVM. The sparse approximation method is also related to [3]. An advantage of this approach is that (γ, σ) can be updated during the pruning process, while for standard SVM approaches one needs to solve the total system for given (γ, σ). Hence the pruning procedure keeps the (γ, σ) selection localized.

7 Recurrent LS-SVM

While the use of classical neural networks (e.g. multilayer perceptrons) in recurrent networks is straightforward, the situation is much more complicated in the context of SVMs. At this point it is important to be aware of the difference between feedforward models (also called NARX or series-parallel models)

$$\hat{y}_k = f(y_{k-1}, y_{k-2}, ..., y_{k-p}, u_{k-1}, u_{k-2}, ..., u_{k-p}) \tag{36}$$

and recurrent models (also called NOE or parallel models)

$$\hat{y}_k = f(\hat{y}_{k-1}, \hat{y}_{k-2}, ..., \hat{y}_{k-p}, u_{k-1}, u_{k-2}, ..., u_{k-p}) \tag{37}$$

with input u_k, output y_k and estimated output $\hat{y}_k$. When using SVMs within the context of the former NARX models, one can apply the standard SVM methods for function estimation or LS-SVMs for function estimation as explained in the previous Sections. When making parametrizations by multilayer perceptrons, the latter NOE models are trained e.g. by Narendra's dynamic backpropagation or Werbos' backpropagation through time [17, 27, 33, 50]. Due to the recursive nature of this model description, the application of SVM methodology within this context is more complicated.

In [39] a recurrent version of LS-SVMs has been introduced. Let us consider the following autonomous recurrent model

$$\hat{y}_k = f(\hat{y}_{k-1}, \hat{y}_{k-2}, ..., \hat{y}_{k-p}). \tag{38}$$

We take the following parametrization

$$\hat{y}_k = w^T \varphi(\begin{bmatrix} \hat{y}_{k-1} \\ \hat{y}_{k-2} \\ \vdots \\ \hat{y}_{k-p} \end{bmatrix}) + b \tag{39}$$

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

$$y_k - e_k = w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) + b \tag{40}$$

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. One can formulate a recurrent LS-SVM as follows

$$\min_{w,b,e} \mathcal{J}(w, b, e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=p+1}^{N+p} e_k^2 \tag{41}$$

subject to the equality constraints

$$y_k - e_k = w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) + b, \quad k = p+1, ..., N+p. \tag{42}$$

It is important to note that the constraints are no longer linear in the unknowns e. At this point convexity is lost. Nevertheless one is still able to retain some of the support vector ideas. From the Lagrangian

$$\mathcal{L}(w, b, e; \alpha) = \mathcal{J}(w, b, e) + \sum_{k=p+1}^{N+p} \alpha_{k-p} \left[y_k - e_k - w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) - b \right] \tag{43}$$

one has the following conditions for optimality

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} & = & w - \sum_{k=p+1}^{N+p} \alpha_{k-p}\, \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) = 0 \\ \frac{\partial \mathcal{L}}{\partial b} & = & \sum_{k=p+1}^{N+p} \alpha_{k-p} = 0 \\ \frac{\partial \mathcal{L}}{\partial e_k} & = & \gamma e_k - \alpha_{k-p} - \sum_{i=1}^{p} \alpha_{k-p+i} \frac{\partial}{\partial e_{k-i}}[w^T \varphi(x_{k-1|k-p} - \xi_{k-1|k-p})] = 0,\ \ k = p+1, ..., N \\ \frac{\partial \mathcal{L}}{\partial \alpha_{k-p}} & = & y_k - e_k - w^T\, \varphi(x_{k-1|k-p} - \xi_{k-1|k-p}) - b = 0,\ \ k = p+1, ..., N+p. \end{cases} \tag{44}$$

By replacing w into the last two conditions and applying Mercer's condition $K(z_{k-1|k-p}\ ,$ $z_{l-1|l-p}) = \varphi(z_{k-1|k-p})^T \varphi(z_{l-1|l-p})$ with $z_{k-1|k-p} = x_{k-1|k-p} - \xi_{k-1|k-p}$, one obtains

$$\begin{cases} \sum_{k=p+1}^{N+p} \alpha_{k-p} = 0 \\ \gamma e_k - \alpha_{k-p} - \sum_{i=1}^{p} \alpha_{k-p+i} \frac{\partial}{\partial e_{k-i}}[\sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{k-1|k-p}, z_{l-1|l-p})] = 0, \quad k = p+1, ..., N \\ y_k - e_k - \sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{k-1|k-p}, z_{l-1|l-p}) - b = 0, \quad k = p+1, ..., N+p. \end{cases} \tag{45}$$

Finding a solution to these equations is computationally heavy. Therefore, in [39] the case $\gamma \to \infty$ has been studied which corresponds to

$$\min_{\alpha,e,b} \ \frac{1}{2} \sum_{k=p+1}^{N+p} e_k^2 \tag{46}$$

subject to

$$\begin{cases} y_k - e_k = \sum_{l=p+1}^{N+p} \ \alpha_{l-p}\, K(z_{l-1|l-p}, z_{k-1|k-p}) + b, \quad k = p+1, ..., N+p \\ \sum_{k=p+1}^{N+p} \ \alpha_{k-p} = 0. \end{cases} \tag{47}$$

Notice that the vectors z depend here on the error variables e. Hence the cost function is subject to a set of nonlinear constraints in e. The resulting recurrent simulation model is given by

$$\hat{y}_k = \sum_{l=p+1}^{N+p} \alpha_{l-p} K(z_{l-1|l-p}, \begin{bmatrix} \hat{y}_{k-1} \\ \hat{y}_{k-2} \\ \vdots \\ \hat{y}_{k-p} \end{bmatrix}) + b \tag{48}$$

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

$$K(z_{k-1|k-p}, z_{l-1|l-p}) = \exp(-\nu \, \|z_{k-1|k-p} - z_{l-1|l-p}\|_2^2) \tag{49}$$

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 [4, 19, 24]. For the recurrent LS-SVM case the parameter estimation problem becomes non-convex. The constrained nonlinear optimization problem can be solved e.g. by sequential quadratic programming (SQP). For large data sets special methods from large scale nonlinear optimization have to be applied. For the limit case $\gamma \to \infty$ which partially neglects the regularization term $\frac{1}{2} w^T w$ (the form of the solution, however, was derived by taking into account a regularization term) one should carefully avoid overfitting. Instead of minimizing the cost function to its local minimum one can apply early stopping in this case, which is equivalent to a form of regularization [1, 27]. Fig.11 illustrates this method on a time-series prediction problem for a double-scroll chaotic attractor. In SVM theory for static problems the parameter ν can be determined e.g. by minimizing upper bounds on the generalization error. In the recurrent case on the other hand the input data of the model are no longer independent from each other.

8 LS-SVMs and optimal control

A first method for using SVMs as controllers has been introduced in [40]. We discuss here how LS-SVM can be used within the N-stage optimal control problem [2] and how robust local stability can be imposed to the scheme.

8.1 N-stage optimal control problem

One can formulate the following N-stage optimal control problem:

$$\min \mathcal{J}(x_k, u_k, w, e_k) = \mathcal{J}_N(x_k, u_k) + \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=1}^{N} e_k^2 \tag{50}$$

subject to the system dynamics

$$x_{k+1} = f(x_k, u_k), \quad k = 1, ..., N \ (x_1 \text{ given})$$

and the control law

$$u_k = w^T \varphi(x_k) + e_k, \quad k = 1, ..., N \tag{51}$$

where $\mathcal{J}_N = \rho(x_{N+1}) + \sum_{k=1}^N h(x_k, u_k)$ (e.g. $h(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$, $\rho(x_{N+1}) = x_{N+1}^T Q x_{N+1}$ with $Q = Q^T > 0$, $R = R^T > 0$), $w \in \mathbb{R}^{n_h}$, $\varphi(\cdot) : \mathbb{R}^n \to \mathbb{R}^{n_h}$ with n_h the number of hidden units of $\varphi(\cdot)$, which can be again infinite dimensional. The actual control signal applied to the plant is assumed to be $w^T \varphi(x_k)$. In the linear control case one has $\varphi(x_k) = x_k$ with $n_h = n$. In the sequel we will employ radial basis function (RBF) kernels. The support vector method exploits Mercer's condition such that one doesn't have to employ $\varphi(\cdot)$ but a kernel function.

One takes the Lagrangian

$$\begin{array}{c} \mathcal{L}(x_k, u_k, w, e_k; \lambda_k, \alpha_k) = \mathcal{J}_N(x_k, u_k) + \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=1}^N e_k^2 + \\ \sum_{k=1}^N \lambda_k^T [x_{k+1} - f(x_k, u_k)] + \sum_{k=1}^N \alpha_k [u_k - w^T \varphi(x_k) - e_k]. \end{array} \tag{52}$$

The conditions for optimality are given by

$$\left\{ \begin{array}{rcll} \frac{\partial \mathcal{L}}{\partial x_k} & = & \frac{\partial h}{\partial x_k} + \lambda_{k-1} - (\frac{\partial f}{\partial x_k})^T \lambda_k - & \\ & & \alpha_k \frac{\partial}{\partial x_k} [w^T \varphi(x_k)] = 0, \qquad k = 2, ..., N & \text{(adjoint equation)} \\ \frac{\partial \mathcal{L}}{\partial x_{N+1}} & = & \frac{\partial \rho}{\partial x_{N+1}} + \lambda_N = 0 & \text{(adjoint final condition)} \\ \frac{\partial \mathcal{L}}{\partial u_k} & = & \frac{\partial h}{\partial u_k} - \lambda_k^T \frac{\partial f}{\partial u_k} + \alpha_k = 0, \quad k = 1, ..., N & \text{(variational condition)} \\ \frac{\partial \mathcal{L}}{\partial w} & = & w - \sum_{k=1}^N \alpha_k \varphi(x_k) = 0 & \text{(support vectors)} \\ \frac{\partial \mathcal{L}}{\partial e_k} & = & \gamma e_k - \alpha_k = 0 \qquad k = 1, ..., N & \text{(support values)} \\ \frac{\partial \mathcal{L}}{\partial \lambda_k} & = & x_{k+1} - f(x_k, u_k) = 0, \quad k = 1, ..., N & \text{(system dynamics)} \\ \frac{\partial \mathcal{L}}{\partial \alpha_k} & = & u_k - w^T \varphi(x_k) - e_k = 0, \quad k = 1, ..., N & \text{(SVM control).} \end{array} \right. \tag{53}$$

From the variational condition one sees how the Lagrange multipliers which are related to the system dynamics interact with the support values of the controller. The obtained set of nonlinear equations is of the form

$$F_1(x_k, x_{N+1}, u_k, w, e_k, \lambda_k, \alpha_k) = 0 \tag{54}$$

for $k = 1, ..., N$ with x_1 given and can be numerically solved in the unknown variables. In comparison with function estimation or classification problems, one looses the advantage of solving a quadratic program or a linear least squares problem. Nevertheless, one is still able to exploit some of the interesting SVM features as in the recurrent LS-SVM case.

After elimination of w, a set of nonlinear equations of the form

$$F_2(x_k, x_{N+1}, u_k, \lambda_k, \alpha_k) = 0 \tag{55}$$

for $k = 1, ..., N$ with x_1 given is obtained. More specifically, after exploiting Mercer's condition, one has

$$\begin{cases} \frac{\partial h}{\partial x_k} + \lambda_{k-1} - (\frac{\partial f}{\partial x_k})^T \lambda_k - \alpha_k \sum_{l=1}^N \alpha_l \frac{\partial K(x_k, x_l)}{\partial x_k} = 0, & k = 2, ..., N \\ \frac{\partial \rho}{\partial x_{N+1}} + \lambda_N = 0 & \\ \frac{\partial h}{\partial u_k} - \lambda_k^T \frac{\partial f}{\partial u_k} + \alpha_k = 0, & k = 1, ..., N \\ x_{k+1} - f(x_k, u_k) = 0, & k = 1, ..., N \\ u_k - \sum_{l=1}^N \alpha_l K(x_l, x_k) - \alpha_k/\gamma = 0, & k = 1, ..., N. \end{cases} \tag{56}$$

The actual control signal applied to the plant becomes

$$u_k = \sum_{l=1}^N \alpha_l K(x_l, x_k) \tag{57}$$

where $\{x_l\}_{l=1}^N, \{\alpha_l\}_{l=1}^N$ are obtained from solving the set of nonlinear equations and x_k is the actual state vector at time k. The data $\{x_l\}_{l=1}^N$ are used as support vector data for the control signal.

8.2 Alternative formulation and stability issues

An alternative formulation can be done by considering the optimization problem

$$\min_{x_k, \alpha_k} J_N(x_k, x_k^r, u_k) + \lambda \sum_{k=1}^N \alpha_k^2 \tag{58}$$

subject to

$$\begin{cases} x_{k+1} & = & f(x_k, u_k)\,, \qquad x_1 \text{ given} \\ u_k & = & \sum_{l=1}^{N} \alpha_l\, K([x_l; x_l^r], [x_k; x_k^r]). \end{cases}$$

Instead of solving a set of nonlinear equations one solves here a constrained nonlinear optimization problem in less unknowns. In this formulation a reference state vector x_k^r is considered. A regularization term $\sum_k \alpha_k^2$ is included with λ a positive real constant.

An important difference between neural network controllers [18, 31, 33] and LS-SVM control is that in the former one solves a parametric optimization problem in the unknown interconnection weights while in the latter also the state vector sequence is part of the unknown parameter vector. The points of the unknown trajectory serve in fact as support vectors. Another difference is that standard neural network approaches always work in the primal weight space, while in SVM methodology the computations are done in the dual space such that the number of unknowns equals the number of training data points (and not the number of weights in the primal space, which can be infinite dimensional). An advantage of LS-SVM control in comparison with classical RBF [22] and fuzzy control methods is that the method is better able to cope with the curse of dimensionality.

In order to impose local stability at a fixed equilibrium point x^{eq}, one locally linearizes the autonomous closed-loop simulation model by evaluating $\frac{\partial f}{\partial \hat{x}_k}$ at x^{eq}. LS-SVM control with a local stability constraint can then be formulated as:

$$\min_{x_k, \alpha_k, Q} \; J_N(x_k, x^{eq}, u_k) + \lambda \sum_{k=1}^{N} \alpha_k^2 \tag{59}$$

subject to

$$\begin{cases} x_{k+1} = f(x_k, \sum_{l=1}^{N} \alpha_l K([x_l; x^{eq}], [x_k; x^{eq}])), \;\; x_1 \text{ given} \\ A^T P A - P < 0 \end{cases}$$

where $P = Q^T Q$, $A = \frac{\partial f}{\partial \hat{x}_k} \mid_{\hat{x}_k = x^{eq}}$. Imposing global asymptotic stability can be done in the spirit of NLq neural control theory [32, 33] or one can impose robust local stability [26, 48] and enlarge the basin of attraction around the equilibrium point. The LS-SVM control method has been successfully applied to the problem of swinging up and stabilizing an inverted pendulum system, where a locally stabilizing LQR controller has been embedded within the LS-SVM control law [31, 40] (Fig.12-13).

When applying an optimization method the constraints will only hold with a certain tolerance. These small numerical errors may lead to differences between the state vectors as solution to optimization problem and the simulation of the closed-loop system with LS-SVM control, especially in the control of unstable systems. For control of stable systems this problem will be less critical. A simulation of the closed-loop system with LS-SVM control

$$\hat{x}_{k+1} = f(\hat{x}_k, \sum_{l=1}^{N} \alpha_l \, K([x_l; x_l^r], [\hat{x}_k; x_k^r])) \quad \hat{x}_1 = x_1 \text{ given} \tag{60}$$

is always needed in order to validate the results, where $\{x_l\}_{l=1}^N$ and $\{\alpha_l\}_{l=1}^N$ are obtained as a solution to the optimization problem.

9 Conclusions

SVM which originated from statistical learning theory is currently an important new direction in the area of neural networks and machine learning. SVM models are obtained by convex optimization problems and are able to learn and generalize in high dimensional input spaces. LS-SVM solutions are characterized by linear KKT systems in dual space where one can have an infinite number of weights in the primal weight space. This new class of kernel based techniques has been successfully applied to a large number of data sets. The LS-SVM method is computationally attractive and easier to extend than standard SVM, e.g. for use in recurrent models and optimal control. One can overcome potential drawbacks as the underlying Gaussian assumptions related to a least squares cost function and the lack of sparseness by applying weighted LS-SVM for robust nonlinear estimation and pruning procedures. In general, SVM methodology might offer several new opportunities in the area of control theory, of which many remain to be further explored for the future.

References

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

[2] Bryson A.E. & Ho Y.C., *Applied optimal control*, Waltham, MA: Blaisdel, 1969.

[3] Chen S.C., Donoho D.L., Saunders M.A., "Atomic decomposition by basis pursuit," *SIAM J. Scientific Computing*, Vol.20, No.1, pp.33-61, 1998.

[4] 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.

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

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

[7] Cristianini N., Shawe-Taylor J., *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.

[8] Evgeniou T., Pontil M., Poggio T., "Regularization networks and support vector machines," *Advances in Computational Mathematics,* Vol.13, No.1, pp.1-50, 2000.

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

[10] Girosi F., "An equivalence between sparse approximation and support vector machines," *Neural Computation*, 10(6), 1455-1480, 1998.

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

[12] Hassibi B., Stork D.G., "Second order derivatives for network pruning: optimal brain surgeon," In Hanson, Cowan, Giles (Eds.) *Advances in Neural Information Processing Systems*, Vol.5, pp.164-171, San Mateo, CA: Morgan Kaufmann, 1993.

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

[14] Huber P.J., *Robust statistics*, New York: Wiley, 1981.

[15] Le Cun Y., Denker J.S., Solla S.A., "Optimal brain damage," In Touretzky (Ed.) *Advances in Neural Information Processing Systems*, Vol.2, pp.598-605, San Mateo, CA: Morgan Kaufmann, 1990.

[16] MacKay D.J.C., "Bayesian Interpolation," *Neural Computation,* Vol.4, No.3, pp.415-447, 1992.

[17] 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.

[18] Parisini T. & Zoppoli R., "Neural networks for feedback feedforward nonlinear control systems," *IEEE Transactions on Neural Networks*, Vol.5, No.3, pp.436-449, 1994.

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

[20] Ridella S., Rovetta S., Zunino R., "Circular backpropagation networks for classification," *IEEE Transactions on Neural Networks*, Vol.8, No.1, pp.84-97, 1997.

[21] Rousseeuw P.J., Leroy A., *Robust regression and outlier detection*, New York: Wiley, 1987.

[22] Sanner R.M. & Slotine J.-J. E., "Gaussian networks for direct adaptive control," *IEEE Transactions on Neural Networks*, Vol.3, No.6, pp.837-863, 1992.

[23] Saunders C., Gammerman A., Vovk V., "Ridge Regression Learning Algorithm in Dual Variables," *Proc. of the 15th Int. Conf. on Machine Learning ICML-98*, Madison-Wisconsin, 1998.

[24] 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.

[25] Schölkopf B., Burges C., Smola A. (Eds.), *Advances in Kernel Methods - Support Vector Learning*, MIT Press, 1998.

[26] Sezer M.E. & Siljak D.D., "Robust stability of discrete systems," *International Journal of Control*, Vol.48, No.5, pp.2055-2063, 1988.

[27] Sjöberg J., Zhang Q., Ljung L., Benveniste A., Delyon B., Glorennec P.-Y., Hjalmarsson H., Juditsky A., "Nonlinear black-box modeling in system identification: a unified overview," *Automatica*, Vol.31, No.12, pp.1691-1724, Dec. 1995.

[28] Smola A., Schölkopf B., Müller K.-R., "The connection between regularization operators and support vector kernels," *Neural Networks*, 11, 637-649, 1998.

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

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

[31] Suykens J.A.K., De Moor B. & Vandewalle J., "Static and dynamic stabilizing neural controllers, applicable to transition between equilibrium points," *Neural Networks*, Vol.7, No.5, pp.819-831, 1994.

[32] Suykens J.A.K., De Moor B. & Vandewalle J., "NL_q theory: a neural control framework with global asymptotic stability criteria," *Neural Networks*, Vol.10, No.4, pp.615-637, 1997.

[33] 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.

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

[35] Suykens J.A.K., Vandewalle J., "Training multilayer perceptron classifiers based on a modified support vector method," *IEEE Transactions on Neural Networks*, Vol.10, No.4, pp.907-912, July 1999.

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

[37] Suykens J.A.K., Lukas L., Van Dooren P., De Moor B., Vandewalle J., "Least squares support vector machine classifiers: a large scale algorithm," *European Conference on Circuit Theory and Design, (ECCTD'99)*, pp.839-842, Stresa Italy, August 1999.

[38] Suykens J.A.K., Lukas L., Vandewalle J., "Sparse approximation using least squares support vector machines," *IEEE International Symposium on Circuits and Systems (ISCAS 2000)*, pp.II757-II760, Geneva, Switzerland, May 2000.

[39] Suykens J.A.K., Vandewalle J., "Recurrent least squares support vector machines," *IEEE Transactions on Circuits and Systems-I*, Vol.47, No.7, pp.1109-1114, Jul. 2000.

[40] Suykens J.A.K., Vandewalle J., De Moor B., "Optimal control by least squares support vector machines," *Neural Networks*, Vol.14, No.1, pp.23-35, Jan. 2001.

[41] Suykens J.A.K., De Brabanter J., Lukas L., Vandewalle J., "Weighted least squares support vector machines : robustness and sparse approximation," to appear in *Neurocomputing special issue.*

[42] Van Gestel T., Suykens J.A.K., Baesens B., Viaene S., Vanthienen J., Dedene G., De Moor B., Vandewalle J., "Benchmarking Least Squares Support Vector Machine Classifiers," *Internal Report 00-37, ESAT-SISTA, K.U.Leuven*, submitted for publication.

[43] Van Gestel T., Suykens J.A.K., Baestaens D., Lambrechts A., Lanckriet G., Vandaele B., De Moor B., Vandewalle J., "Financial time series prediction using least squares support vector machines within the evidence framework," *IEEE Transactions on Neural Networks* (special issue on Neural Networks in Financial Engineering), in press.

[44] Van Gestel T., Suykens J.A.K., De Moor B. Vandewalle J., "Automatic relevance determination for least squares support vector machine regression", *9th European Symposium on Artificial Neural Networks* (ESANN 2001), pp.13-18, Bruges Belgium, April 2001.

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

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

[47] 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.

[48] Vidyasagar M., *Nonlinear Systems Analysis*, Prentice-Hall, 1993.

[49] Vidyasagar M., *A Theory of Learning and Generalization*, Springer-Verlag, 1997.

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

[51] Williams C.K.I., Rasmussen C.E., "Gaussian processes for regression," In D.S. Touretzky, M.C. Mozer, and M.E. Hasselmo (Eds), *Advances in Neural Information Processing Systems* 8, pp 514-520. MIT Press, 1996.

Captions of Figures

Figure 1: Typical problems with classical neural networks: local minima and choice of model complexity. In SVMs for classification and static nonlinear function estimation one solves convex optimization problems. The number of hidden units follows as a solution to this problem.

Figure 2: LS-SVMs are related with regularization networks and Gaussian processes and lead to solving linear systems. However, LS-SVMs have closer links with Vapnik type SVMs and optimization theory bcause of explicit primal-dual formulations. This enables to extend the LS-SVM methodology from static to recurrent models and optimal control.

Figure 3: (Top-left) linearly separable classification problem: several separating hyperplanes exist; (Top-right) in SVMs a unique hyperplane is defined which maximizes the margin to the nearest points; (Bottom) non-separable data case which leads to the use of slack variables in the SVM formulation.

Figure 4: In nonlinear SVMs the input data are projected to a higher dimensional (possibly infinite dimensional) feature space by the mapping $\varphi(\cdot)$. A linear separation is made in this higher dimensional feature space, corresponding to a nonlinear separation in the original input space of interest.

Figure 5: LS-SVM with RBF kernel for solving a two-spiral classification problem with good generalization, which is known to be hard for multilayer perceptrons. The points indicated by 'o' and '$*$' are the given training data for the two classes; black and white visualize the binary classification regions.

Figure 6: Vapnik's ϵ-insensitive loss function in the case of SVMs for nonlinear function estimation.

Figure 7: Application of weighted LS-SVM to robust estimation of a sinc function corrupted with zero mean white Gaussian noise and outliers.

Figure 8: Application of weighted LS-SVM to robust estimation of a sinc function corrupted by a heavy tailed non-Gaussian distribution.

Figure 9: Pruning of the LS-SVM spectrum in order to obtain a sparse approximation by LS-SVM.

Figure 10: LS-SVM sparse approximation procedure with RBF kernel applied to a sinc function. The figure shows the result before and after pruning, together with the support vectors.

Figure 11: Use of a recurrent LS-SVM in a time-series prediction for a double-scroll chaotic system. The training data are shown before the vertical line. The recurrent LS-SVM output is shown in dashed line. As given initial state $[y_{12}; y_{11}; ...; y_1]$ was used. Given this initial state the trajectory is well predicted over a time horizon of about 500 points, being aware of the fact that chaotic systems are very sensitive with respect to small perturbations.

Figure 12: LS-SVM control for swinging up and stabilizing an inverted pendulum system. An LQR control law is embedded in the LS-SVM control in order to ensure local stability at the endpoint.

Figure 13: (Continued) state variables and control signal with respect to time. The vertical line shows the time horizon over which an optimal control problem was formulated for the LS-SVM controller. Good generalization and stability is obtain and visible beyond this horizon.

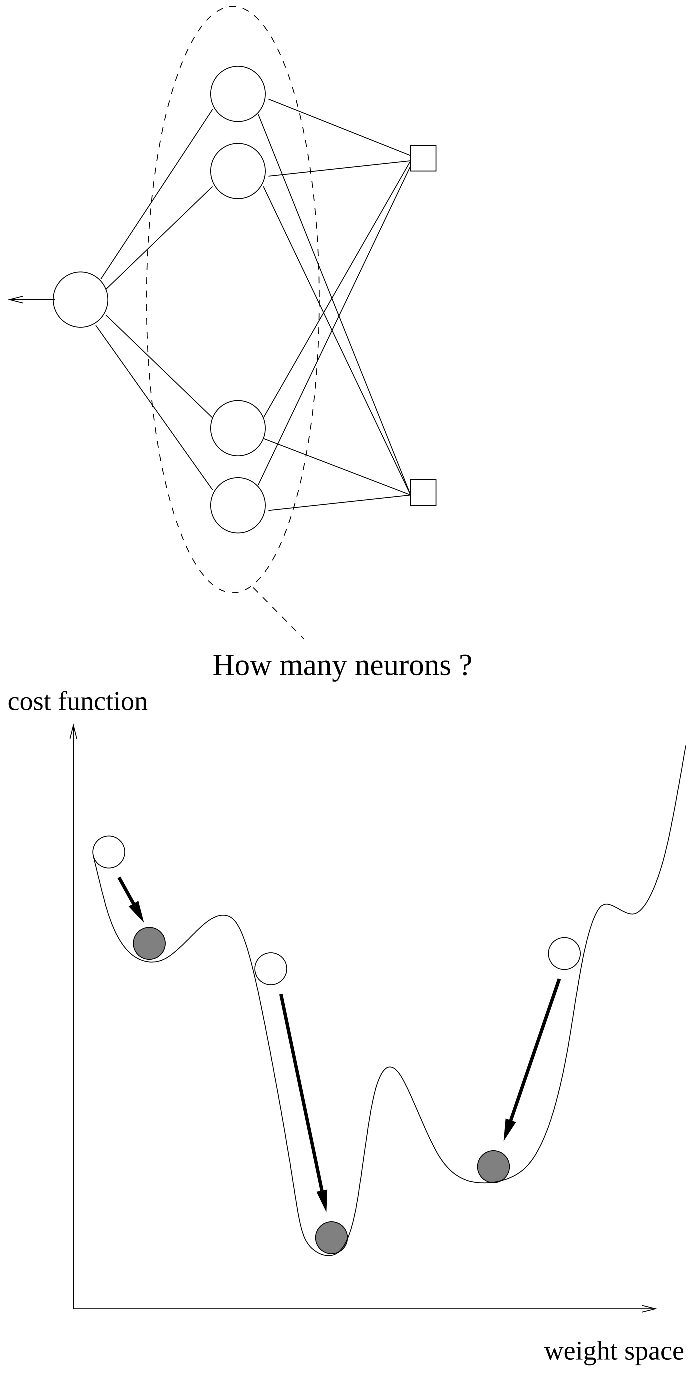

Figure 1:

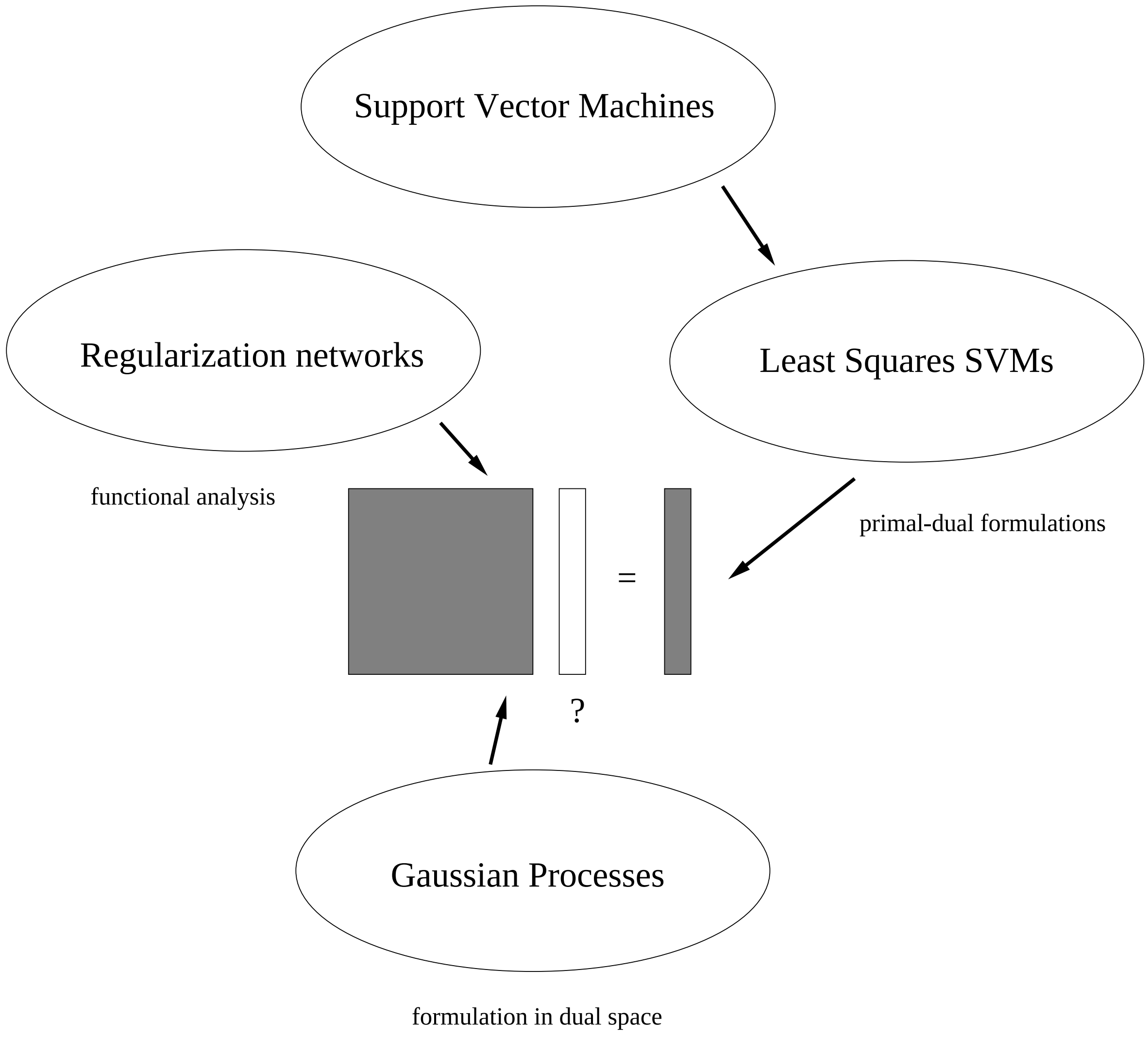
Support Vector Machines
Regularization networks
Least Squares SVMs
functional analysis
primal-dual formulations
=
?
Gaussian Processes
formulation in dual space

Figure 2:

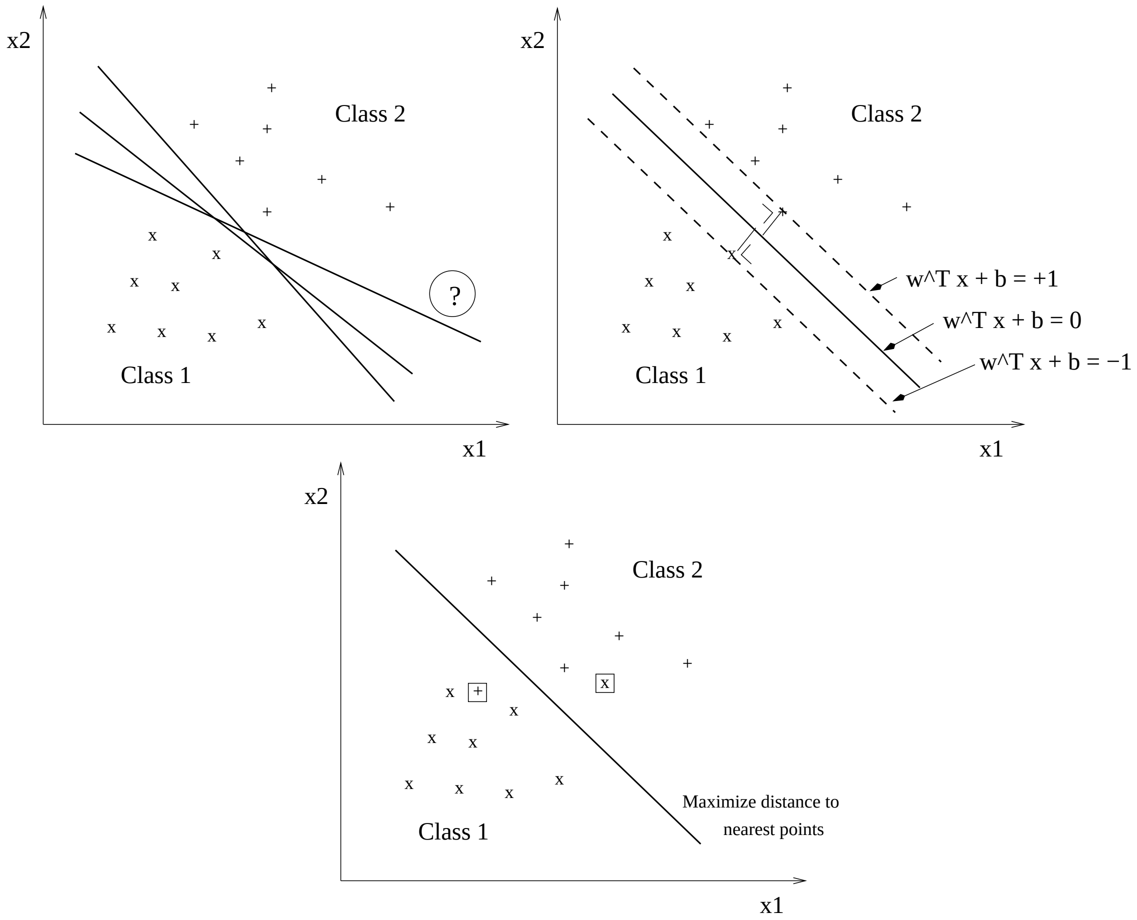
x2
Class 2
?
Class 1
x1
x2
Class 2
w^T x + b = +1
w^T x + b = 0
w^T x + b = −1
Class 1
x1
x2
Class 2
Class 1
Maximize distance to
nearest points
x1

Figure 3:

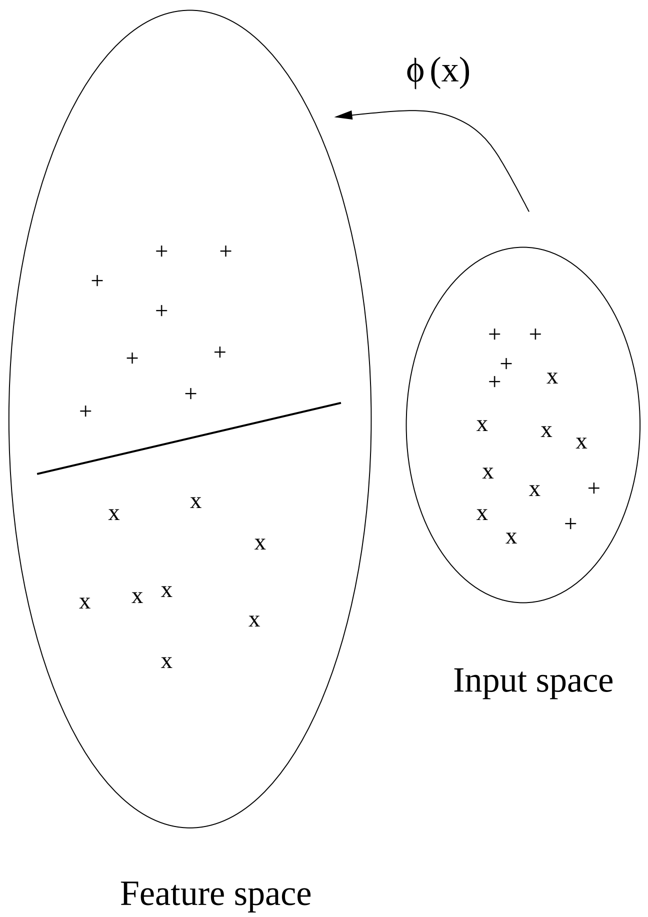

Figure 4:

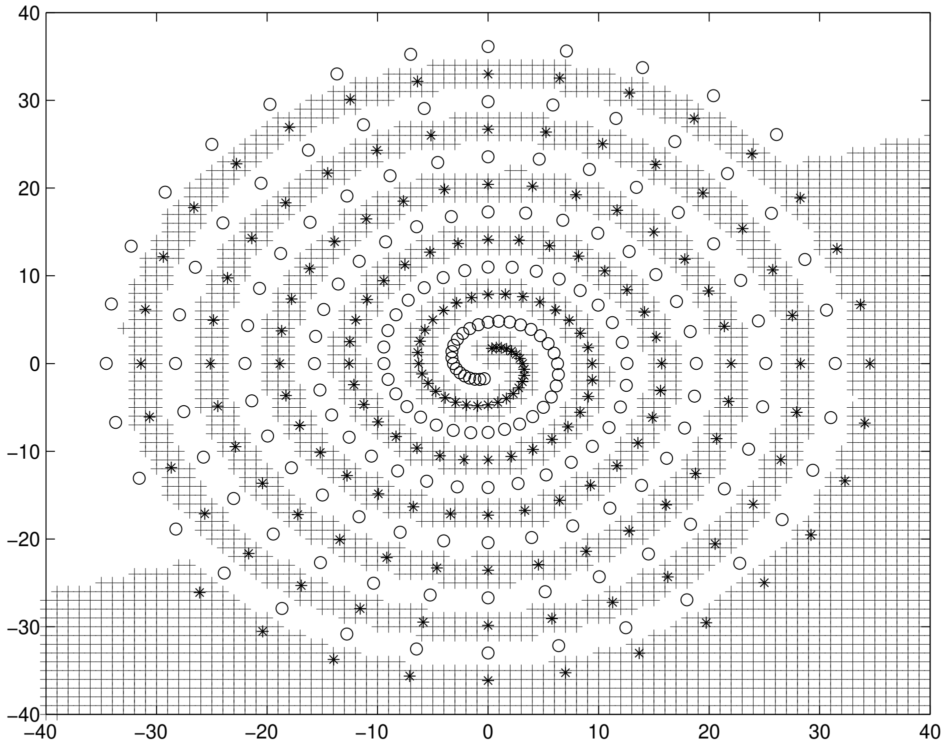

Figure 5:

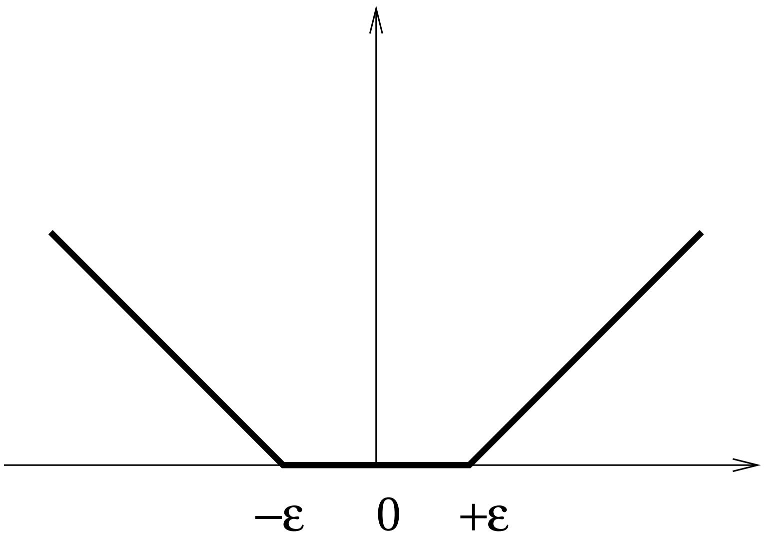

Figure 6:


TRAINING DATA: f(x)=sinc(x)
f(x)
x
TEST DATA: f(x)=sinc(x)
f(x)
x
alpha/gamma values
alpha/gamma
index
histogram (alpha/gamma) values,
freq
(alpha/gamma)

Figure 7:

Figure 8:

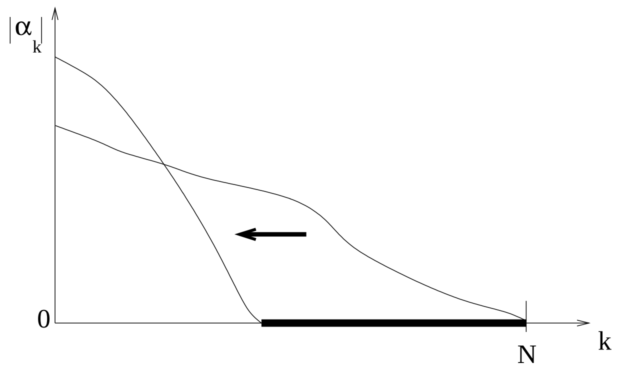

Figure 9:

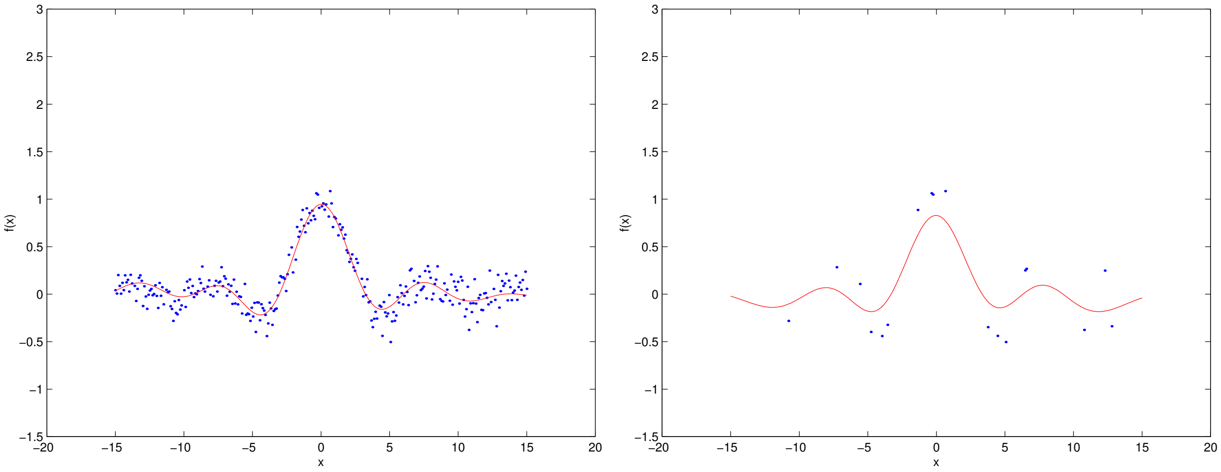

Figure 10:

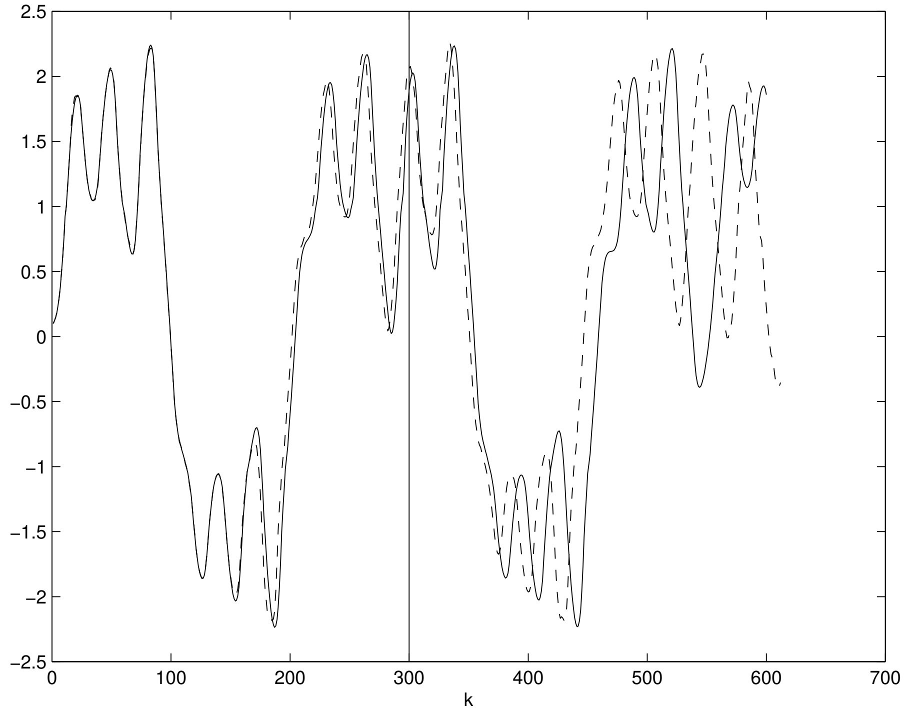

Figure 11:

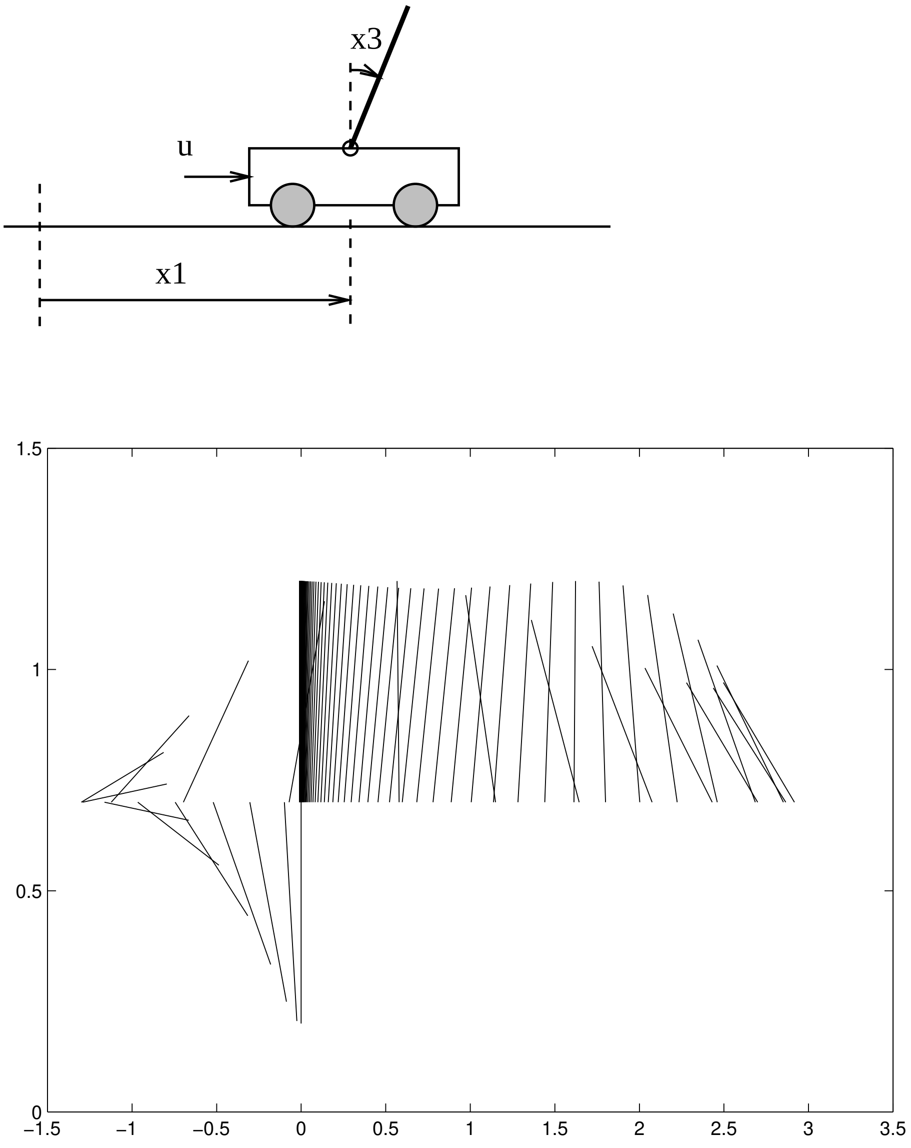

Figure 12:

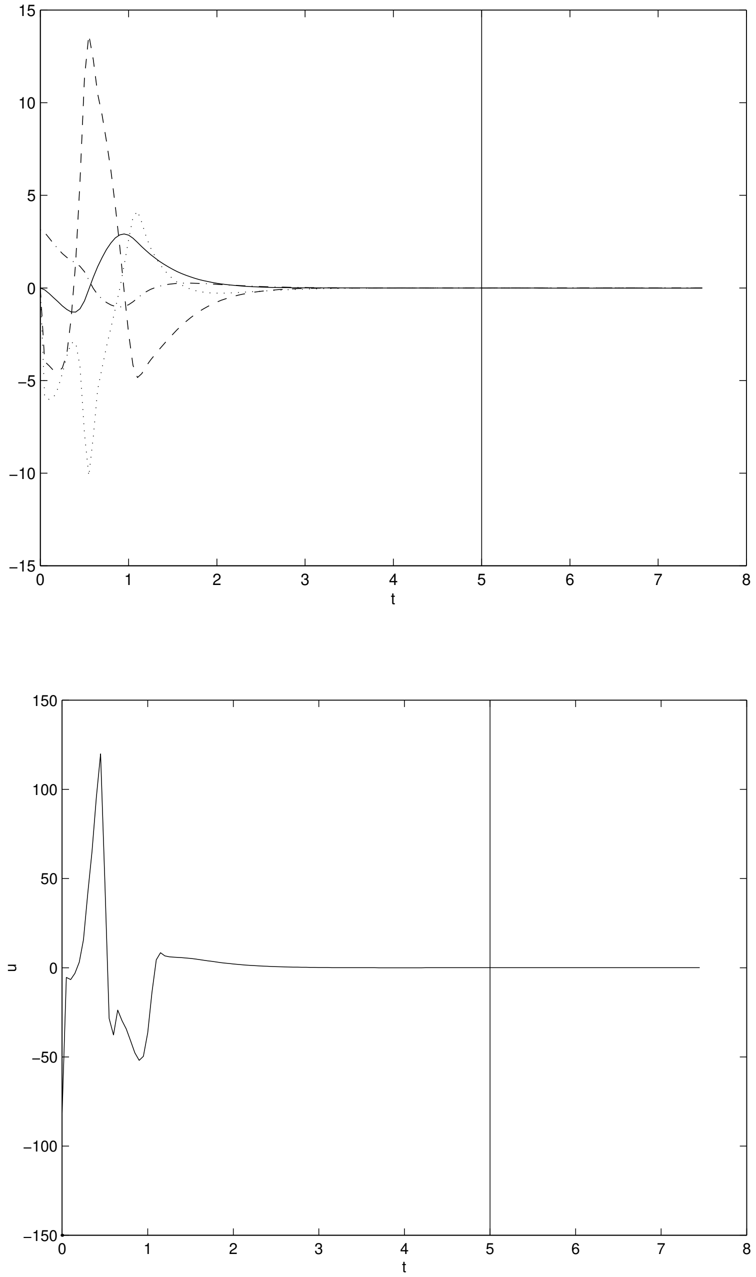

Figure 13: