Support Vector Machines for System Identification

P. Drezet and R. F. Harrison

Department of Automatic Control & Systems Engineering
The University of Sheffield, Sheffield, S1 3JD

Research Report No. 704

Abstract

Support Vector Machines (SVMs) are used for system identification of both linear and non-linear dynamic systems. Discrete time linear models are used to illustrate parameter estimation and non-linear models demonstrate model structure identification. The VC dimension of a trained SVM indicates the model accuracy without using separate validation data. We conclude that SVMs have potential in the field of dynamic system identification, but that there are a number of significant issues to be addressed.

1 Introduction

Support vector machines (SVMs) were developed in a pattern classification context as an implementation of structural risk minimisation [6]. Regression estimation can be performed by an extended form of SVM and has been shown to perform well in areas such as identifying chaotic time series models [4].

This paper demonstrates SVMs in linear and non-linear system identification applications. Simple regression problems illustrate practical implications of SVMs including linear dynamic model parameter estimation. Nonlinear systems show how SVMs include the minimum complexity in their constitution to solve a problem with a particular set of parameters.

A feature of this type of learning algorithm is that an estimate of generalisation ability may be obtained from a trained network without using test data, assuming the training data is representative. Vapnick Chervonenkis (VC) theory [5] is the fundamental idea behind SVMs which defines a measure of a learning machine's complexity (VC dimension). This metric provides a bound on the probability of test set errors and can be calculated theoretically for a type of learning machine, or estimated from a particular trained learning machine. These general attributes are useful for system identification, especially for non-linear systems where required model complexity is difficult to estimate.

2 Support Vector Regression

Support Vector Machines for regression purposes yield an approximation function of the form

$$f(\vec{x}) = \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) K(\vec{x}_i, \vec{x}) + b \qquad (1)$$

The function, K, can be a linear dot product or a nonlinear function obeying Mercer's condition [2]. In the linear case, the function, K, can be separated and the weights found by $\sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\vec{x}_i$. Non-linear kernel functions imply a non-linearly expanded feature space (Hilbert space) in which inner products are performed. A sum of *convolved inner products*, K, [6] replace an expansion, ϕ, of input vectors, $\vec{x}$; i.e. $\sum_{i=1}^{l} K(\vec{x}_i, \vec{x}) \equiv \vec{w}.\phi(\vec{x})$. The use of kernel functions replaces a possibly very high dimensional Hilbert space and therefore does not explicitly increase the feature space dimension.

The algorithm calculates values α_i and α_i^* associated with a subset of training vectors, $\vec{x}_i$. The subset of training vectors are called support vectors. The value, b, in equation (1) is a bias term.

To calculate the parameters, the following cost function is minimised.

$$C(\vec{w}, b) = \frac{1}{2}\vec{w}^2 + C\sum_{i=1}^{l}(\zeta(\xi_i) + \zeta(\xi_i^*)) \qquad (2)$$

subject to

$$y_i - (\vec{w}.\vec{x}_i + b) \leq \xi_i^* + \epsilon \qquad (3)$$

$$\vec{w}.\vec{x}_i + b - y_i \leq \xi_i + \epsilon \qquad (4)$$

$$\xi_i, \xi_i^* \geq 0 \quad i = 1, \cdots, l \qquad (5)$$

This functional minimises the magnitude of the weights, $\vec{w}$, within the limits of the interpolation function (1) causing errors on the training data. The parameter, C, controls the cost incurred by training errors and is used for regularisation. The slack variables, ξ, are introduced to accommodate errors on the training set.

The constraints also include a term, ϵ, which allows a margin of error without incurring any cost. A non-negative value of ϵ means that the support vectors can be a sparse subset of the training data. The ϵ-insensitive zone defines a manifold in feature space where the approximating function can exist without being considered a training error. Minimisation of the weight vector norm fixes the approximation function within this manifold, if possible.

The function, ζ, can be used to adjust the loss function to training noise, e.g. quadratic or Huber robust loss function [3]. The choice of the function ζ

is limited by the optimisation process chosen. The preferred method of optimisation is quadratic programming, which limits the function, ζ, to a linear or quadratic function of ξ. For example choosing $\zeta(u) = u^2$ the loss function can be written as the following Lagrangian

$$\begin{array}{c} L(\vec{w}, b, \vec{\xi}, \vec{\xi}^*, \vec{\alpha}, \vec{\alpha}^*, \vec{\eta}\vec{\eta}^*) = \\ \frac{1}{2}|\vec{w}|^2 + \frac{C}{2}\sum_{i=1}^{l}\left(\xi_i^2 + \xi_i^{*2}\right) \\ -\sum_{i=1}^{l}\alpha_i\left[\xi_i + \epsilon_i + y_i - (\vec{w}.\vec{x}_i + b)\right] \\ -\sum_{i=1}^{l}\alpha_i^*\left[\xi_i^* + \epsilon_i^* - y_i - (\vec{w}.\vec{x}_i + b)\right] \\ -\sum_{i=1}^{l}\eta_i\xi_i + \eta_i^*\xi_i^* \end{array} \quad (6)$$

Lagrange multipliers $\alpha, \alpha^*, \eta, \eta^*$ are introduced for constraints (3)-(5).

Partially differentiating (6) w.r.t $\vec{w}$, ξ_i, ξ_i^* and b and back substitution yields the functional in the parameters $\vec{\alpha}, \vec{\alpha}^*, \vec{\eta}, \vec{\eta}^*$. The choice of ζ in this example makes constraint (5) redundant and so terms including η can be dropped giving,

$$\begin{array}{c} L(\vec{\alpha}, \vec{\alpha}^*) = \\ -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)(\vec{x}_i.\vec{x}_j) \\ +\sum_{i=1}^{l}(\alpha_i^* - \alpha_i)y_i - \epsilon\sum_{i=1}^{l}\alpha_i + \alpha_i^* \\ -\sum_{i=1}^{l}\frac{\alpha_i^2 - \alpha_i^{*2}}{C} \end{array} \quad (7)$$

This dual quadratic functional can be maximised using any standard optimisation technique. The quadratic form of the cost function is convex and hence has a unique solution. The training process finds values of Lagrange multipliers, α_i, α_i^*. Each pair of these variables complements a training vector, $\vec{x}_i$. Training vectors not involved in defining the constraints (3) and (4) must satisfy Kühn-Tucker conditions and so both multipliers are set to zero. The pairs, α_i, α_i^*, with a positive value define the set of support vectors, $\vec{x}_i$.

This complementary set and a bias value, b, give the necessary information for the interpolation function (1). The bias value can be calculated from any support vector $\vec{x}_j$ using equation (1) and setting $f(\vec{x}_j) = y_j + \mathrm{sgn}(\alpha_j - \alpha_j^*)\epsilon + \alpha_j - \alpha_j^*$.

3 Experiments

Examples of the SVM paradigm are described for simple cases of linear and non-linear regression, and applied to parameter estimation of dynamic systems.

3.1 Function Approximation

Figure 1(a) shows a single dimensional linear approximation of the function $y = x$. It demonstrates how a non-zero ϵ-insensitive zone allows the constraints (3) & (4) to be in effect for a sparse set of training vectors, $\vec{x}_i$. In the noise free case this results in an approximation error that is a function of ϵ and the range of y and $\vec{x}$. The insensitive zone will give unbiased results when uniform additive noise is present and the value ϵ is equal to the maximum noise amplitude.

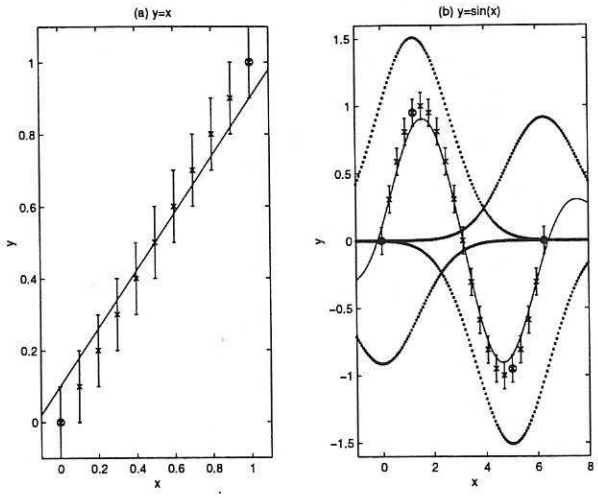

Figure 1: (a) The linear case shows the training data (crosses) with 0.1 ϵ insensitive zones marked on each as error bars. The estimated function (solid line) has a minimum gradient constrained by the ϵ insensitive zone. The two support vectors are marked by circles. (b) The sin function training data is estimated with Gaussian kernel functions $exp[(\vec{x}_i - \vec{x})^2/4]$. Four support vectors are required. Their corresponding contributing Gaussians are shown (dotted).

The nonlinear example in figure 1(b) shows how the superposition of the Gaussian kernel functions, centred at the support vectors, constructs the sin function. In many practical nonlinear regression problems noise is also present and the art of learning the underlying function lies in not over-fitting the noise while capturing the non-linearities. The value of C in the cost function (2) trades training set precision for regularisation. The ϵ-insensitive zone trades precision for sparsity in the support vectors. The choice of both these parameters requires knowledge of noise distribution in the training data.

order	a_1	a_2	b_1	b_2	#SV
1 (s)	0.5	-	0.5	-	-
1(m)	0.500	-	0.500	-	3
2(m)	0.500	0.167	0.167	0.167	4
2(s)	0.5	0	0.5	0.2	-
2(m)	0.500	0.000	0.500	0.200	5

Table 1: Linear system parameters of system (s) and estimated model (m). Noise free training examples were used. ϵ was set to 10^{-3} and C set to 10^6.

3.2 Dynamic System Parameter Identification

For the purposes of time series prediction and system identification, the objective is to find a set of parameters for a proposed model. These parameters are chosen to fit measured values of output, and usually, input data.

3.2.1 Linear Systems

A difference equation of the form (8) is a suitable type of model for discrete measurements from deterministic linear systems. The model parameters are then the order of the model, N_a, N_b, and the values for the coefficients a_n, b_n.

$$y(k) = \sum_{n=1}^{N_a} a_n u(k-n) - \sum_{n=1}^{N_b} b_n y(k-n) \qquad (8)$$

The results of estimating parameters of system orders 1 & 2 are shown in table 1. In cases where the estimation model order is the same as the generating model the coefficients are found accurately. In the case where a second order estimation model and first order generating model are used, the parameters do not appear to concur. Pole-zero cancellation is responsible for the observed discrepancy. A consequence of this is that there is a degree of freedom in the solution.

When the model order is correct the number of support vectors is one larger than the dimension of the input space ($N_a + N_b + 1$). This is the maximum VC dimension of a linear learning machine. Geometrically the support vectors are the minimum number of points required to define a hyper-plane in the feature space.

Where the model order is greater than that of the system, the number of support vectors is less than the maximum VC dimension of the learning machine. Each pole-zero cancellation gives rise to an extra degree of freedom and hence one less support vector per cancellation. The solution of the hyper-plane is that which minimises the sum of squared weight values, w_i, but does not necessarily set individual weights to zero.

For linear systems, the coefficients $(\vec{a}, \vec{b})$ are equal to the weights $(\vec{w})$.

	a_1	b_1	b_2	b_3	#SV
System	0.5	0.3	0.2	0.015	-
Pol(2)	0.500	0.299	0.200	0.0151	15
Pol(3)	0.500	0.297	0.201	0.0153	35

Table 2: Non-linear system coefficient values after expansion of kernel functions $(\vec{x}_i.\vec{x}+1)^2$, $(\vec{x}_i.\vec{x}+1)^3$. Ineffective coefficients are below 0.001.

3.2.2 Non-linear Systems

Nonlinear systems are problematic for system identification because model structure permutations can be huge. The dimension explosion when NARMAX [1] type models are expanded for linear parameter estimation can be prohibitive. The effective number of non-linear terms in the real system is generally much smaller than that of a general NARMAX model $F(\vec{y}_{-1..N_y}, \vec{u}_{0..N_u}, \xi_{0..N_\xi})$. Where ξ_i are the noise terms.

SVMs allow a high dimensional feature space to be represented implicitly by scalar kernel functions. Suitable kernel functions include Gaussian, hyperbolic tangent, and polynomial, which can be directly related to network configurations such as radial basis, multi-layer perceptrons, or polynomial learning machines. During training, only the necessary number of non-linear terms are found to describe the training data.

The following noise free non-linear system is used to demonstrate the use of various kernel functions.

$$\begin{aligned} y(k) = {} & a_1 u(k-1) - b_1 y(k-1) \\ & -b_2 y(k-2) - b_3 y^2(k-1) \end{aligned} \qquad (9)$$

The model used for identification is

$$y(k) = f\left(u(k-1), u(k-2), y(k-1), y(k-2)\right) \quad (10)$$

Table 2 shows results with two types of polynomial kernels. The coefficients have been estimated accurately within the bounds of ϵ-insensitivity. The ineffective coefficients were all found have absolute values less than 10^{-3}. The number of support vectors increases as the complexity of kernel functions increases. Inaccuracies in coefficient estimation also increase with kernel function complexity.

The inaccuracy of model coefficients is related to the SVM's decreasing generalisation ability because data can be over-fitted when complexity is available in the learning machine. A bound to generalisation ability is given by the structural risk [5]. This can be calculated with probability $1-\eta$:

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h\left(log\frac{2l}{h}+1\right) - log(\eta/4)}{l}} \quad (11)$$

for $l > h$, and where $R(\alpha)$ is the risk, given parameters, α, and $R_{emp}(\alpha)$ is the empirical risk. Empirical risk is a function of test set errors which in these examples is zero (i.e. solution lies inside the insensitive zone). The additional term is the *confidence term* which includes, h, the VC dimension of the SVM, and l, the number of training vectors. Polynomial functions have finite VC dimensions which are less than or equal to the expanded dimension. SVMs using kernels such as Gaussians, have possibly infinite VC dimension. The actual VC dimension of a trained SVM is estimated from $h_{est} = R^2|\vec{w}_o|^2$ [6] where R is the smallest radius of a sphere which contains training vectors in feature space. $\vec{w}_o$ is the optimal weight vector.

A practical implication of the above generalisation bound is that for any kernel function, an indication of generalisation ability can be gained from the ratio of support vectors to training vectors. In the extreme case, if the number of support vectors equals the number of training examples it is improbable that future estimates will be accurate. These metrics can be used to compare prototype learning machines with different kernel functions without the need for test data.

3.2.3 Noisy Data

In real system identification problems, data is usually corrupted by noise. Uncertainty can arise from measurement instruments, system noise, or unmodelled dynamics. The character of the noise may be additive, multiplicative, and be variously distributed. Comparisons are made with SVMs and least square estimates for stochastic systems.

Uncorrelated white noise in linear systems can be modelled as additive noise at the output. The following model's uncertainty averages to unbiased estimates of parameters by using simple least square methods.

The system model used is

$$y(k) = au(k-1) + by(k-1) + \xi(k) \quad (12)$$

First-order model			ϵ			
	(Sys.)	(LS)	0.001	0.1	1.0	2
a	0.5	0.488	0.488	0.487	0.483	0.432
b	0.4	0.418	0.418	0.416	0.392	0.3353
#SV			50	42	6	3
Second-order model						
a_1	0.5	0.490	0.490	0.489	0.494	0.4171
a_2	0	-0.281	-0.280	-0.283	-0.324	-0.147
b_1	0.4	0.146	0.145	0.149	0.204	0.102
b_2	0	0.232	0.232	0.233	0.283	0.0568
#SV			50	46	5	4
MPO		0.404	0.405	0.410	0.466	0.888

Table 3: Results from Gaussian noise added to data from 1st order system. Parameters are shown for first-order and second order models with various values of ϵ. The second-order model is tested by the mean magnitude of errors from model predicted output (MPO)

where $\xi(k)$ is Gaussian white noise with standard deviation 0.5, and input $u(k-1)$ is Gaussian white noise with standard deviation of 10.

The results in table 3 show that for small ϵ-insensitivity zones the parameter estimates are unbiased and similar to least square estimates. Indeed without insensitivity the loss function $\zeta(\xi_i) = \xi_i^2$ implements a least square estimation. The number of support vectors for diminishing ϵ values increases until each training vector is included in the support vector set. The reason for this is that noise in training data forces errors in the training set for small values of ϵ. Training vectors which cause errors meet the constraints (3)-(4) and hence become support vectors.

The second-order models are performance tested using the mean absolute error from model predicted outputs, produced from separate test data. The MPO error can be seen to increase dramatically when ϵ is set to 2. This insensitivity is greater than output noise amplitudes and so the number of support vectors concurs with the noise free case. The ϵ-insensitivite zone is not an appropriate loss function for Gaussian noise because its deviation from its mean is not limited.

Table 4 shows the effects of uniform additive noise on linear and non-linear models. The linear model SVMs perform comparably with least square estimates. By comparing the SVMs trained with uniform noise to those trained with Gaussian noise it can be seen that performance is improved where support vectors are sparse. The non-linear system given by

First-order model			ϵ			
	(Sys.)	(LS)	0.9	1.0	1.1	1.2
a	0.5	0.507	0.505	0.509	0.510	0.507
b	0.4	0.389	0.386	0.384	0.375	0.346
#SV			8	6	4	3
Second-order model			0.9	1.1	1.4	-
a_1	0.5	0.509	0.501	0.506	0.487	-
b_1	0.015	0.382	0.396	0.398	0.313	-
b_2	0.4	0.0176	0.0108	0.0096	0.0124	-
#SV			12	9	6	-

Table 4: Results with Uniform noise added to data from a linear 1st order system and non-linear first order system. Parameters are shown for varying ϵ.

$y(k) = a_1 u(k-1) - b_1 y(k-1) - b_2 y^2(k-1) + \xi(k)$ also shows the SVM's performance to compare well with a least squares estimate.

The analytical properties of SVMs can be seen to be compromised in stochastic problems because noise generates additional support vectors. Sometimes insensitivity may be increased with little effect on model accuracy with the benefit of yielding a sparse support vector set. In circumstances where signal to noise ratio is good, and noise amplitude is restricted, the insensitivity zone can allow the SVM to solve a problem as if a deterministic case.

4 Conclusions

The properties of support vector regression are attractive for non-linear system identification because high dimensional expansions are not explicitly required to solve for linear parameters. By comparison with other kernel-based learning methods such as radial basis functions, SVMs have the advantage that the number of kernels and centre positions are found automatically and optimally. Other parameters associated with kernels can be optimised relatively easily because prototypes do not have to be tested to judge performance.

In linear models the occurrence of pole-zero cancellation caused by an excess of lags is detected by missing support vectors. In general, however, for non-linear or stochastic systems, the expected number of support vectors is not easily calculated, and superfluous model lags may be difficult to spot. For analytical treatment of the model, standard model coefficients can be calculated from the weighted sum of the relevant terms in the kernel functions.

Ordinary support vector regression does not always give a parsimonious model description as can the orthogonal parameter estimation algorithm for non-linear stochastic systems [1]. Arbitrarily complex systems can however be learnt with controllable and measurable 'accuracy' within the physical limits of the machine. The SVM's inherent properties fulfill the majority of requirements for general purpose linear and non-linear system identification.

Problems that must be overcome to improve SVMs practical viabilty are mainly concerned with the treatment of noisey data. For stochastic systems, a dillema exists between model accuracy and simplicity, for which the magnitude of ϵ-insensitivity is instrumental. The pre-requisite value for ϵ has many practical disadvantages because of the arbitary nature of the noise it is attempting to eliminate. Further weakness lies in model order identification. Detection of superfluous lags is detectable for deterministic linear systems only, and explicit identification of model order is not automatic. Methods of improving these points are currently under investigation.

References

[1] M. Korenberg S. Billings and Y. P. Liu. an orthogonal parameter estimation algorithm for nonlinear systems. *Research report 307, Dpt. Automatic Control & Systems Engineering, University of Sheffield*, 1987.

[2] R. Courant and D. Hilbert. *Methods of Mathematical Physics.* J. Whiley, New York, 1953.

[3] P. Huber. Robust estimation of location parameter. *Annals of Mathematical Statistics*, 1(20), 1964.

[4] S. Mukherjee E. Osuna and F. Girosi. Nonlinear prediction of chaotic time series using support vector machines. *proceedings of IEEE NNSP'97*, pages 24–26, 1997.

[5] V. Vapnik and Z. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Doklady Akademii Nauk USS*, 4(181), 1968.

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