Available online at www.sciencedirect.com
SCIENCE DIRECT®

ELSEVIER

Chemometrics and Intelligent Laboratory Systems 69 (2003) 35–49

Chemometrics and intelligent laboratory systems

www.elsevier.com/locate/chemolab

Using support vector machines for time series prediction

U. Thissen[a], R. van Brakel[a], A.P. de Weijer[b], W.J. Melssen[a], L.M.C. Buydens[a,*]

[a] *Laboratory of Analytical Chemistry, University of Nijmegen, Toernooiveld 1 6525 ED Nijmegen, The Netherlands*
[b] *Teijin Twaron Research Institute, Postbus 9600, 6800 TC Arnhem, The Netherlands*

Received 20 December 2002; accepted 5 June 2003

Abstract

Time series prediction can be a very useful tool in the field of process chemometrics to forecast and to study the behaviour of key process parameters in time. This creates the possibility to give early warnings of possible process malfunctioning. In this paper, time series prediction is performed by support vector machines (SVMs), Elman recurrent neural networks, and autoregressive moving average (ARMA) models. A comparison of these three methods is made based on their predicting ability. In the field of chemometrics, SVMs are hardly used even though they have many theoretical advantages for both classification and regression tasks. These advantages stem from the specific formulation of a (convex) objective function with constraints which is solved using Lagrange Multipliers and has the characteristics that: (1) a global optimal solution exists which will be found, (2) the result is a general solution avoiding overtraining, (3) the solution is sparse and only a limited set of training points contribute to this solution, and (4) nonlinear solutions can be calculated efficiently due to the usage of inner products. The method comparison is performed on two simulated data sets and one real-world industrial data set. The simulated data sets are a data set generated according to the ARMA principles and the Mackey–Glass data set, often used for benchmarking. The first data set is relatively easy whereas the second data set is a more difficult nonlinear chaotic data set. The real-world data set stems from a filtration unit in a yarn spinning process and it contains differential pressure values. These values are a measure of the contamination level of a filter. For practical purposes, it is very important to predict these values accurately because they play a crucial role in maintaining the quality of the process considered. As it is expected, it appears that the ARMA model performs best for the ARMA data set while the SVM and the Elman networks perform similarly. For the more difficult benchmark data set the SVM outperforms the ARMA model and in most of the cases outperforms the best of several Elman neural networks. For the real-world set, the SVM was trained using a training set containing only one tenth of the points of the original training set which was used for the other methods. This was done to test its performance if only few data would be available. Using the same test set for all methods, it appeared that prediction results were equally well for both the SVM and the ARMA model whereas the Elman network could not be used to predict these series.

Keywords: Time series; Process chemometrics; SVM—support vector machines; ARMA—autoregressive moving average; RNN—recurrent neural networks

1. Introduction

Process chemometrics typically makes use of process parameters consecutively measured in time (a time series). Historical time series can be used to con-

* Corresponding author. Tel.: +31-24-36-53192; fax: +31-24-36-52653.
E-mail address: lbuydens@sci.kun.nl (L.M.C. Buydens).

0169-7439/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0169-7439(03)00111-4

struct control charts that monitor current process measurements and evaluate if they meet specific quality demands (i.e. facilitating early fault detection). Furthermore, the statistics used to set up control charts can also facilitate the very important aspect of fault diagnosis. However, none of these methods are able to prevent non-normal process operations. For the prevention of process malfunctioning, it is important to predict the value of key process parameters in the future and try to adapt them in advance.

Predicting the dynamic behaviour of a process in time as a function of process input parameters usually is called process identification and can be seen as regression with a time component. The step of process identification is a necessary predecessor before adapting process parameters to avoid unwanted process functioning (process control). Another useful approach is time series prediction in which future parameter values are predicted as a function of its values in the past (autoregression with time component). Using time series prediction it is possible to study the behaviour of key parameters in the future and to give early warnings of possible process malfunctioning.

Both control charts and predicting techniques can be used for process chemometrics but different requirements are set on the data. When using control charts, the time series have to consist of independent and identically distributed (i.i.d.) random values from a known distribution. This means that the series have to be uncorrelated with the same distribution for all measurements or have to be transformed as such because this distribution is used to perform fault detection [1]. In contrast, predicting parameters in time cannot be performed without a relation being present between past, current and future parameter values (explicit autocorrelation in time).

The goal of this paper is to use a support vector machine (SVM) for the task of time series prediction. SVM is a relatively new nonlinear technique in the field of chemometrics and it has been shown to perform well for classification tasks [2], regression [3] and time series prediction [4]. Useful references, data and software on SVMs are available on the internet (http://www.kernel-machines.org). For the task of classification, Belousov et al. [5] compare the performance of SVMs to both linear and quadratic discriminant analysis on their flexibility and generalization abilities. They show that SVMs can handle higher dimensional data better even with a relatively low amount of training samples and that they exhibit a very good generalization ability for complex models. In this paper, the performance of the SVM is compared with the performance of both autoregressive moving average (ARMA) models and recurrent neural networks (RNNs). ARMA models and RNNs are well known methods for time series prediction but they suffer from some typical drawbacks. The major disadvantage of an ARMA model is its linear behaviour but on the other hand it is easy and fast in use. RNNs can in principle model nonlinear relations but they are often difficult to train or even yield unstable models. Another drawback is the fact that neural networks do not lead to one global or unique solution due to differences in their initial weight set. On the other hand, SVMs have been shown to perform well for regression and time series prediction and they can model nonlinear relations in an efficient and stable way. Furthermore, the SVM is trained as a convex optimisation problem resulting in a global solution which in many cases yields unique solutions. However, their major drawback is that the training time can be large for data sets containing many objects (if no specific algorithms are used that perform the optimisation efficiently).

In this paper, two simulated data sets and a real-world data set are used for time series prediction. The simulated data sets are a time series that is constructed using the principles of ARMA models and the Mackey–Glass data set. The first data set is used to check the performance of the prediction methods in a case in which one of the methods should be able to find a parametric model resembling the underlying system of the series. The Mackey–Glass data set is included because it is often used as a benchmark data set. The real-world data set stems from a filtration unit used in a high performance yarn spinning process. It contains differential pressure values over the filter. The pressure difference relative to the initial pressure difference is a measure of the amount of contamination inside the filtration unit. If the filter is contaminated too much (i.e. it is replaced late), the yarn quality will decrease because of the extrusion of contaminants through the filter. However, replacing it too early should be avoided as well because of the high costs of the filter. Therefore, making early and accurate predictions of the differential pressure increases the efficiency in deciding when to replace a filter unit.

2. Theory

2.1. Support vector machines

SVMs have originally been used for classification purposes but their principles can be extended easily to the task of regression and time series prediction. Because it is out of the scope of this paper to explain the theory on SVM completely, this section focuses on some highlights representing crucial elements in using this method. The literature sources used for this section are excellent general introductions to SVMs [6–9], and tutorials on support vector classification (SVC) and support vector regression (SVR) by Refs. [2,3], respectively.

SVMs are linear learning machines which means that a linear function ($f(x) = wx + b$) is always used to solve the regression problem. The best line is defined to be that line which minimises the following cost function (Q):

$$Q = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} L^{\varepsilon}(x_i, y_i, f)$$

$$\text{subject to} \begin{cases} y_i - wx_i - b \le \varepsilon + \xi_i \\ wx_i + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} \tag{1}$$

The first part of this cost function is a weight decay which is used to regularize weight sizes and penalizes large weights. Due to this regularization, the weights converge to smaller values. Large weights deteriorate the generalization ability of the SVM because, usually, they can cause excessive variance. Similar approaches can be observed in neural networks and ridge regression [9,10]. The second part is a penalty function which penalizes errors larger than $\pm\varepsilon$ using a so-called ε-insensitive loss function L^{ε} for each of the N training points. The positive constant C determines the amount up to which deviations from ε are tolerated. Errors larger than $\pm\varepsilon$ are denoted with the so-called slack variables ξ (above ε) and ξ^* (below ε), respectively. The third part of the equation are constraints that are set to the errors between regression predictions ($wx_i + b$) and true values (y_i). The values of both ε and C have to be chosen by the user and the optimal values are usually data and problem dependent. Fig. 1 shows the use of the slack variables and the linear ε-insensitive loss function that are used throughout this paper.

The minimisation of Eq. (1) is a standard problem in optimisation theory: minimisation with constraints. This can be solved by applying Lagrangian theory and from this theory it can be derived that the weight vector, w, equals the linear combination of the training data:

$$w = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) x_i \tag{2}$$

In this formula, α_i and α_i^* are Lagrange multipliers that are associated with a specific training point. The asterisks again denote difference above and below the regression line. Using this formula into the equation of

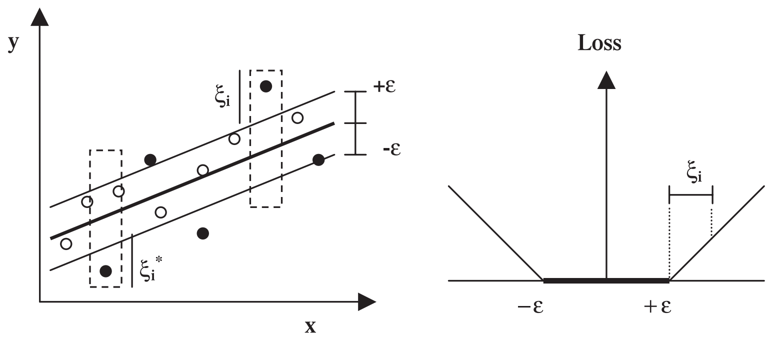

Fig. 1. Left, the tube of ε accuracy and points that do not meet this accuracy. The black dots located on or outside the tube are support vectors. Their origin is explained later. Right, the (linear) ε-insensitive loss function is shown in which the slope is determined by C.

a linear function, the following solution is obtained for an unknown data point x:

$$f(x) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) \cdot \langle x_i, x\rangle + b \qquad (3)$$

Because of the specific formulation of the cost function and the use of the Lagrangian theory, the solution has several interesting properties. It can be proven that the solution found always is global because the problem formulation is convex [2]. Furthermore, if the cost function is strictly convex, the solution found is also unique. In addition, not all training points contribute to the solution found because of the fact that some Lagrange multipliers are zero. If these training points would have been removed in advance, the same solution would have been obtained (sparseness). Training points with nonzero Lagrange multipliers are called *support vectors* and give shape to the solution. The smaller the fraction of support vectors, the more general the obtained solution is and less computations are required to evaluate the solution for a new and unknown object. However, many support vectors do not necessarily result in an overtrained solution. Furthermore, note that the dimension of the input becomes irrelevant in the solution (due to the use of the inner product).

In the approach above it is described how linear regression can be performed. However, in cases where nonlinear functions should be optimised, this approach has to be extended. This is done by replacing x_i by a mapping into feature space, $\varphi(x_i)$, which linearises the relation between x_i and y_i (Fig. 2). In the feature space, the original approach can be adopted in finding the regression solution.

When using a mapping function, the solution of Eq. (3) can be changed into:

$$f(x) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b \qquad (4)$$

with

$$K(x_i, x) = \langle \varphi(x_i), \phi(x)\rangle$$

In Eq. (4), K is the so-called kernel function which is proven to simplify the use of a mapping. Finding this mapping can be troublesome because for each dimension of x the specific mapping has to be known. Belousov et al. [5] illustrate this for classification purposes. Using the kernel function the data can be mapped implicitly into a feature space (i.e. without full knowledge of φ) and hence very efficiently. Representing the mapping by simply using a kernel is called the *kernel trick* and the problem is reduced to finding kernels that identify families of regression formulas. The most used kernel functions are the Gaussian RBF with a width of σ: $K(x_i, x) = \exp(-0.5\|x - x_i\|^2/\sigma^2)$ and the polynomial kernel with an order of d and constants a_1 and a_2: $K(x_i, x) = (a_1 x_i^T x + a_2)d$. If the value of σ is set to a very large value, the RBF kernel approximates the use of a linear kernel (polynomial with an order of 1).

In contrast to the Lagrange multipliers, the choice of a kernel and its specific parameters and ε and C do not follow from the optimisation problem and have to be tuned by the user. Except for the choice of the kernel function, the other parameters can be optimised by the use of Vapnik–Chervonenkis bounds, cross-validation, an independent optimisation set, or Bayesian learning [6,8,9]. Data pretreat-

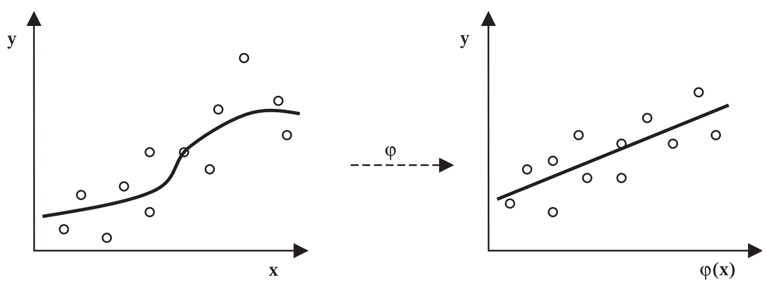

Fig. 2. Left, a nonlinear function in the original space that is mapped into the feature space (right) were the function becomes linear.

ment of both x and y can be options to improve the regression results just as in other regression methods but this has to be investigated for each problem separately.

Time series prediction can be seen as autoregression in time. For this reason, a regression method can be used for this task. When performing time series prediction by SVMs, one input object (x_i) to the SVM is a finite set of consecutive measurements of the series: $\{x(t_i), x(t_i-s),\ldots, x(t_i-\tau s)\}$. In this series, t_i is the most recent time instance of the input object i and s is the (sampling) time step. The integer factor of τ determines the time window and, thus, the number of elements of the input vector. The output of the regression, y_i, is equal to $x(t_i+h)$ where h is the prediction horizon. Usually, for industrial problems, the value of h is determined in advance for the application as a border condition. When performing time series prediction, the input window becomes an additional tunable parameter. This is also the case for the autoregressive moving average models and the Elman recurrent neural networks which are explained next.

2.2. Autoregressive moving average models

ARMA models have been proposed by Box and Jenkins [11] as a mix between autoregressive and moving average models for the description of time series. In an autoregressive model of order p ($\mathrm{AR_p}$), each individual value x_t is expressed as a finite sum of p previous values and white noise, z_t:

$$x_t = \alpha_1 x_{t-1} + \ldots + \alpha_p x_{t-p} + z_t \tag{5}$$

The parameters α_i can be estimated from the Yule–Walker equations which are a set of linear equations in terms of their autocorrelation coefficient [1,11,12]. In a moving average model of order q ($\mathrm{MA_q}$), the current value x_t is expressed as a finite sum of q previous z_t:

$$x_t = \beta_0 z_t + \beta_1 z_{t-1} + \ldots + \beta_q z_{t-q} \tag{6}$$

In this equation, z_i is the white noise residual of the measured and predicted value of x at time instance i. The model parameters β_i usually are determined by a set of nonlinear equations in terms of the autocorrelations. The z's are usually scaled so that $\beta_0=1$. In the past, moving average models have particularly been used in the field of econometrics where economic indicators can be affected by a variety of 'random' events such as strikes or government decisions [12]. An ARMA model with order (p,q) is a mixed $\mathrm{AR_p}$ and $\mathrm{MA_q}$ model and is given by:

$$x_t = \alpha_1 x_{t-1} + \ldots + \alpha_p x_{t-p} + z_t + \beta_1 z_{t-1} + \ldots + \beta_q z_{t-q} \tag{7}$$

Using the backward shift operator B, the previous equation can be written as:

$$\phi(B)x_t = \theta(B)z_t \tag{8}$$

where $\phi(B)$ and $\theta(B)$ are polynomials of order p, q, respectively, such that:

$$\phi(B) = 1 - \alpha_1 B - \ldots - \alpha_p B^p \text{ and } \theta(B) = 1 + \beta_1 B + \ldots + \beta_q B^q. \tag{9}$$

2.3. Elman recurrent neural networks

An Elman recurrent neural network is a network which in principle is set up as a regular feedforward network [13]. This means that all neurons in one layer are connected with all neurons in the next layer. An exception is the so-called *context layer* which is a special case of a hidden layer. Fig. 3 shows the architecture of an Elman recurrent neural network. The neurons in this layer (context neurons) hold a copy of the output of the hidden neurons. The output of each hidden neuron is copied into a specific neuron in the context layer. The value of the context neuron is used as an extra input signal for all the neurons in the hidden layer one time step later (delayed cross-talk). For this reason, the Elman network has an explicit memory of one time lag.

Similar to a regular feedforward neural network, the strength of all connections between neurons are indicated with a weight. Initially, all weight values are chosen randomly and are optimised during the stage of training. In an Elman network, the weights from the hidden layer to the context layer are set to *one* and are fixed because the values of the context neurons have to be copied exactly. Furthermore, the initial output weights of the context neurons are equal to half the output range of the other neurons in the network. The Elman network can be trained with gradient descent

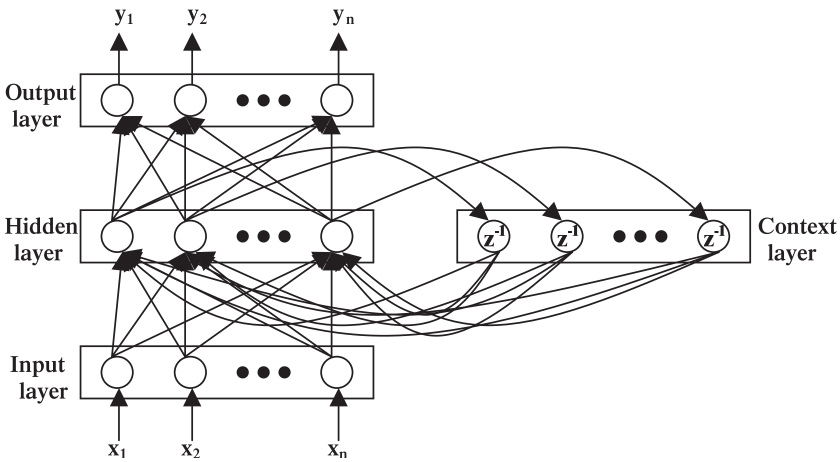

Fig. 3. A schematic representation of an Elman recurrent neural network.

backpropagation, similar to regular feedforward neural networks.

3. Experimental

3.1. Data sets used

3.1.1. Simulated data

The two simulated data sets used in this study are data set that was generated according to the principles of ARMA and the Mackey–Glass data set.

First, the ARMA data set is used to compare the prediction methods. This data set has been chosen because it is based on the ARMA principles and its underlying model should be found almost exactly by the ARMA model. The ARMA time series used (Fig. 4) stems from an ARMA(4,1) model according to Eq. (10) [12]:

$$\phi(B) = 1 - 0.55B - 0.53B^2 + 0.74B^3 - 0.25B^4 \text{ and } \theta(B) = 1 - 0.2B^1. \tag{10}$$

From the figure, it follows that the ARMA time series is not periodic and from a certain time lag autocorrelation hardly is present.

The Mackey–Glass data set originally has been proposed as a model of blood cell regulation [14]. It is often used in practice as a benchmark set because of its nonlinear chaotic characteristics. Chaotic time series do not converge or diverge in time and their trajectories are highly sensitive to initial conditions. The differential equation leading to the time series is:

$$\frac{dx_t}{dt} = ax_t + \frac{bx_{t-d}}{1 + x_{t-d}^c} \tag{11}$$

The typical values used for benchmarking are: $a = -0.1$, $b = 0.2$, $c = 10$, $d = 17$. Fig. 5 shows a plot of the Mackey–Glass data together with the auto-

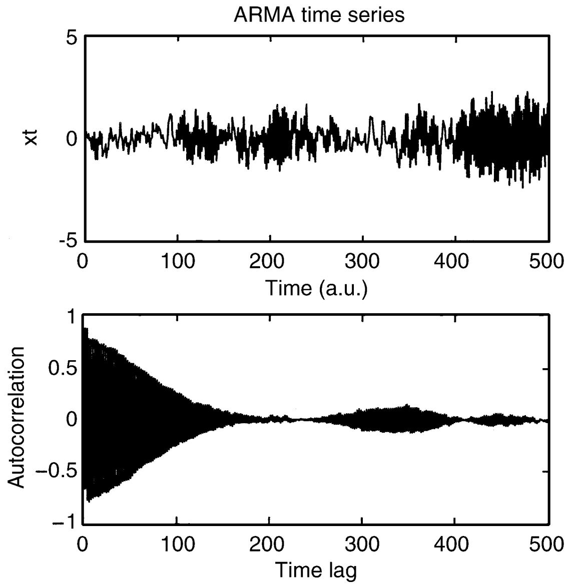

Fig. 4. The first 500 points in the ARMA(4,1) time series together with its autocorrelation plot which is based on 1500 data points. One data point can be considered as one time step.

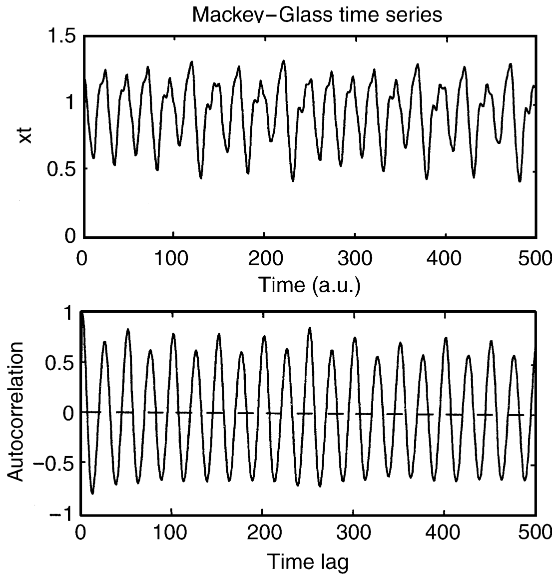

Fig. 5. The first 500 points in the Mackey-Glass time series together with its autocorrelation plot based on 1500 data points.

correlation plot. From the autocorrelation plot, it can be seen that in contrast to the previous data set, the autocorrelation values are not vanishing to zero and from its behaviour it can be concluded that the data is periodic which can make prediction easier.

3.1.2. Industrial data

Filtration of solid and gel-like particles in a viscous liquid phase is an important unit operation in the fiber production industry. Contamination in spinning solution or polymer melt has a direct effect on the mechanical properties and quality of the yarn. The service life of a filter element depends on the contamination level, the filtrate volume and filter surface area. In the process, the particles being removed are collected by the filter, which becomes progressively blocked. As the filter begins to retain more contaminant and the flow rate remains constant, the differential pressure (ΔP_t) increases. During normal operation, the ΔP_t relative to the initial pressure difference (ΔP_0) is measured continuously and a graph is generated which shows differential pressure versus time. Because this quantity indicates the amount of contamination inside the filtration unit, the graph is used to determine the maximum operating time of the filter element. In this specific industrial application, the limit of $\Delta P_t/\Delta P_0$ used to assure a good quality of the yarns is the factor of 16/10.

Most of the methods that determine the cleanness of spinning solutions assume a completely specific filtration mechanism. For laminar, non-pulsatile fluid flow through a filter, the flow rate (volume per unit time) is given by Darcy's Equation:

$$\phi = \frac{k \cdot A \cdot \Delta P}{\eta \cdot x} \tag{12}$$

with ϕ: filtrate flux (m^3/s), k: permeability factor (m^2), A: effective filter area (m^2), ΔP: pressure drop over filter (N/m^2), η: viscosity of the fluid (N s/m^2), x: filter thickness (m).

At constant flow and viscosity of the filtrate the increase in pressure drop over a filtration system can simply be regarded as a reduction of the permeability factor due to the clogging of the filter medium. In practice, these laws are often not applicable. Filtration proceeds by intermediate laws complicated by deformation of the filter medium, by retained gel-like particles, non-Newtonian flow behaviour or by local temperature differences. Therefore, exact modeling is difficult and thus time series prediction of the differential pressure is a useful alternative to be able to take preventive actions before yarn quality is affected.

Fig. 6 shows some examples of time series of the differential pressure relative to the pressure difference measured directly after a new filter has been installed. It can be seen that the series can be very diverse. Some series are short (series A) while others are very long (series B, D, and F). Furthermore, some series suffer from large vertical drops (series C) that can stem from various changes in the process but are not relevant for this problem. Finally, for not all series the critical limit is exceeded or is it exceeded only in the end (series B and D). Filter series A and B are used as a training set because of their typical contamination behaviour.

3.2. Software used

For the SVM calculations, a Matlab toolbox was used, developed by Gunn [15]. Standard Matlab

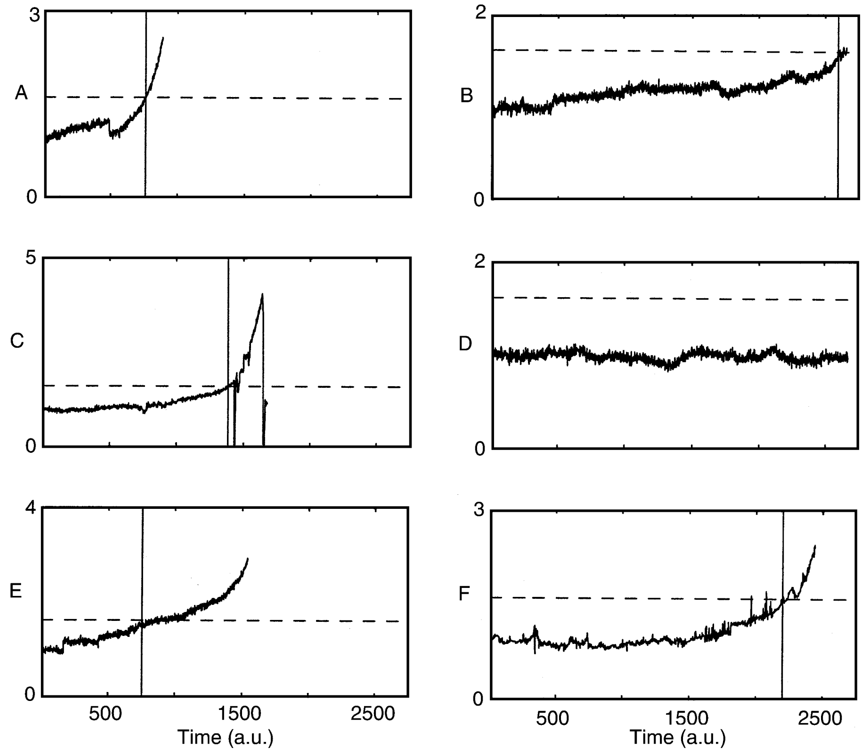

Fig. 6. The differential pressure relative to the initial differential pressure for several typical examples. The horizontal line indicates the critical point after which a filter should be replaced (critical point: 16/10). The vertical line indicates the time instance at which this critical point is exceeded.

toolboxes have been used for the calculations of ARMA (system identification toolbox) and the Elman network (neural network toolbox).

4. Results

4.1. ARMA time series

The ARMA time series is used because it is a relatively easy data set constructed according to the principles of ARMA. It is expected that the ARMA model performs (slightly) better than the other methods because it must be possible to find the underlying model almost exactly. It can be expected that the SVM and the Elman network, on the other hand, cannot exactly reproduce the model underlying the data.

The ARMA data set is divided into four consecutive data sets each with a length of 300 data points. The result is a training set, an optimisation set, a test set, and a separate validation set. Increasing the size of the training set does not lead to a decrease of the prediction error. The training and the optimisation sets are used to determine the best model settings while the test set is used to determine the final prediction errors for each prediction horizon. The validation set is used for the Elman network to prevent overtraining. Optimisation of all model settings has been done by performing a systematic grid search over the parameters with a prediction horizon of 5 using the optimisation set. It is expected that all methods can make reasonable predictions with this horizon. The optimal model settings found are shown in Table 1. The input window is a time-delayed matrix of the time series as described in the Theory. All prediction errors are calculated by taking the RMSE between the predicted signal and its reference signal, divided by the standard deviation of the reference signal.

Table 1
The optimal settings found for the prediction of the ARMA(4,1) time series

	ARMA(4,1) time series		
	SVM	Elman	ARMA
Input window	4	3	4
Hidden neurons	–	1	–
Error terms	–	–	1
C	1	–	–
ε	0.1	–	–
Kernel enter (width)	RBF (3)	–	–
TF hidden layer	–	Sigmoid	–
TF other layers	–	Linear	–

The factor of C is the weight of the loss function in SVM, where ε is the maximal acceptable error. TF denotes the use of a transfer function in the Elman network.

In Fig. 7, the relative prediction error as a function of the prediction horizon is plotted for the three methods used. Showing the results of all five Elman networks would not change the principle of the figure but would only make it less clear. The best Elman network is the one which has lowest overall error. The number of support vectors for the SVM range from 75% to 86%. As expected, it follows that the ARMA model performs better than the other methods. The SVM and the best Elman network perform more or less similarly. However, the prediction error is relatively high which is caused by the fact that the data have very little autocorrelation. This again causes the training set to differ slightly from the test set. However, the model found by ARMA resembles the target model of Eq. (10) very closely:

$$\phi(B) = 1 - 0.6081B - 0.4934B^2 + 0.7414B^3 - 0.2927B^4 \text{ and } \theta(B) = 1 - 0.2644B^1.$$

The difference for each coefficient is:

$$\Delta\phi(B) = 0.0581B - 0.0366B^2 - 0.0014B^3 + 0.0427B^4 \text{ and } \Delta\theta(B) = 0.0644B^1.$$

The ARMA model also finds the correct order of the data. This is also true for the input window of the SVM whereas the input window of the Elman network slightly deviates. Furthermore, the figure shows that the best prediction horizon of the three methods is 1 which is closely related to the high prediction error. Furthermore, the relative high number of support vectors are also caused by the fact that time series prediction is complicated due to the low autocorrelation present.

4.2. Mackey–Glass data set

In contrast to the ARMA data set, the underlying model of the chaotic Mackey–Glass data cannot be modelled parameterically by any of the three methods used. In this case, the predicting performance of the three methods are compared under different experimental conditions. These are the noise level on the data and the prediction horizon. The noise levels used are 0%, 5%, 15%, and 25% and the prediction horizon ranged from 1 to 25 time steps. For a simulated data set, these conditions can be set but these usually are fixed border conditions for real-world data. In the ideal case, for all experimental conditions, the prediction methods should be optimised separately. However, in this approach, two situations are considered: (1) noise-free data for which the prediction methods are optimised with a prediction horizon of 5 time steps and (2) noisy data for which the methods are optimised with one noise level (15%) and a prediction horizon also of 5 time steps. These specific settings for optimisation are selected because these are intermediate values in this comparison and it is expected that the predicting ability of all methods for these settings is reasonable. This has also been verified for a few other settings.

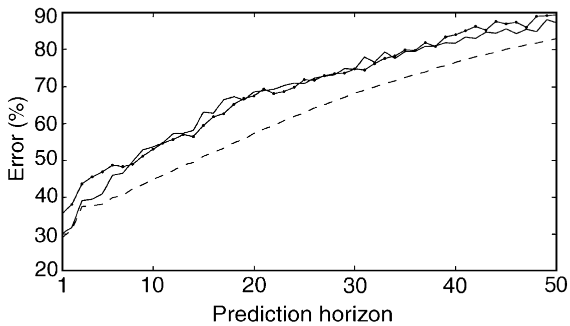

Fig. 7. The prediction results of the ARMA time series as a function of the prediction horizon. The results plotted are the SVM model (solid dotted line), the Elman network (solid line), and the ARMA model (dashed line).

Table 2
The optimal settings found for the prediction of the Mackey-Glass data

	Noise-free series			Noisy series		
	SVM	Elman	ARMA	SVM	Elman	ARMA
Input window	15	8	18	10	13	19
Hidden neurons	–	6	–	–	11	–
Error terms	–	–	5	–	–	7
C	15	–	–	1	–	–
ε	0.001	–	–	0.1	–	–
Kernel (width)	RBF (1)	–	–	RBF (1)	–	–
TF hidden layer	–	Sigmoid	–	–	Sigmoid	–
TF other layers	–	Linear	–	–	Linear	–

A separation between noise-free data and noisy data is made because noise-free data are a special case requiring different settings of the prediction methods. All methods have been optimised similarly to the previous data in addition with prior knowledge derived from Ref. [4].

Table 2 shows the optimal settings found with the same data division as for the ARMA data set. The only difference is the optimal training set size which is 200 for this case. The number of support vectors range from about 10% for the easier cases until 95% for the more difficult prediction cases (with a large prediction horizon). The high value of the support vectors for these cases can be caused because the SVM is optimised for a prediction horizon of 5. This is supported by Fig. 8 which shows that the prediction errors remain more or less stable up to a prediction horizon of 10. For (much) larger horizons, the problem becomes more difficult and more support vectors will be required to describe the regression function. However, this does not indicate overtraining which is avoided using the independent optimisation set.

For the noise-free situation, it can be seen that the SVM clearly outperforms both the ARMA model and the Elman network. For the Elman network, the results of five trained models are shown. The differences between these results represent the variation in the models due to the usage of different initial weight sets. For the noisy situations, the SVM also outperforms the ARMA model and the Elman networks perform in an intermediate way. The SVM model performs better than or equal to the best Elman network at each situation. However, the best Elman network for each prediction horizon does not always stem from the same initial weight set. It can be seen that all methods perform worse if more noise is added to the series. Additionally, the different Elman networks also show

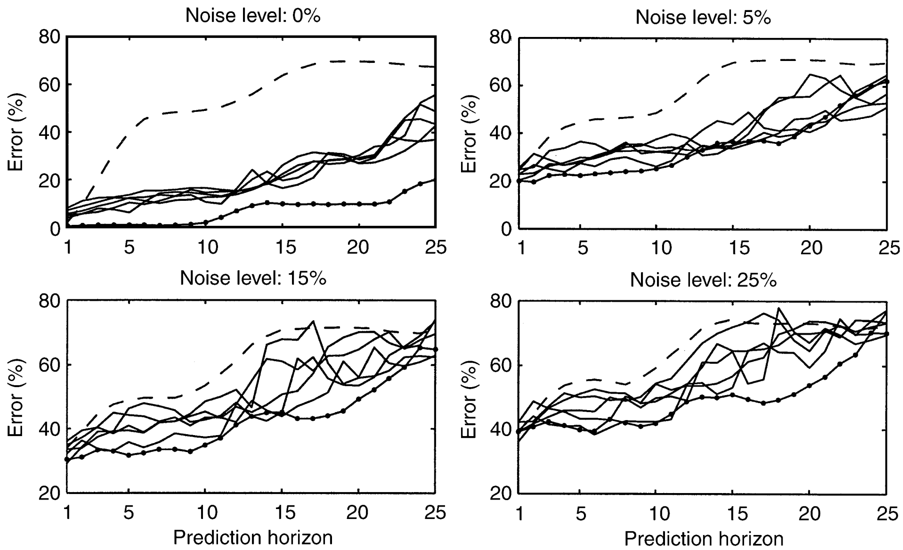

Fig. 8. The prediction results of the Mackey-Glass time series as a function of the noise level and the prediction horizon. The results plotted stem from the SVM model (solid dotted line), the Elman networks (solid lines), and the ARMA model (dashed line).

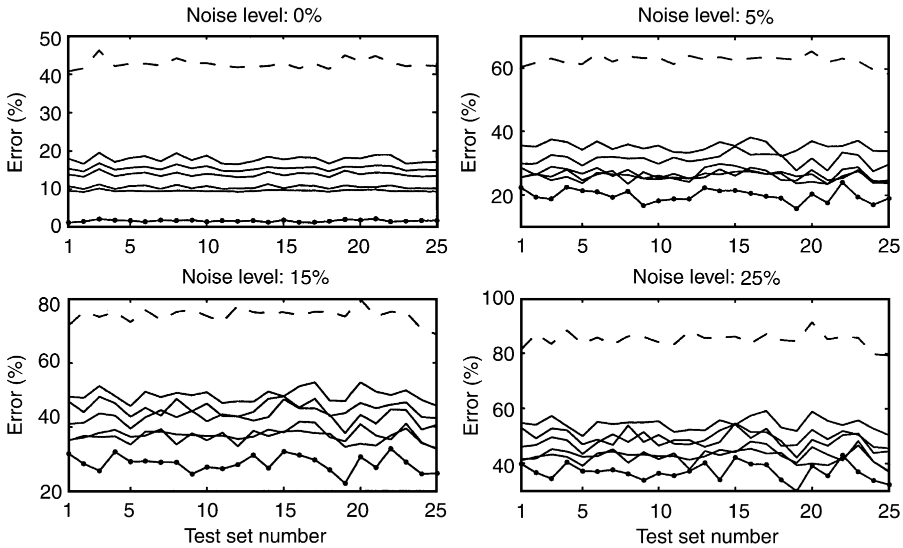

Fig. 9. The prediction error of the Mackey-Glass time series as a function of different consecutive test sets. The results plotted stem from the SVM model (solid dotted line), the Elman networks (solid lines), and the ARMA model (dashed line).

more variation. The reason why the ARMA model performs worst can be explained from its linearity where the time series dependency is nonlinear. Furthermore, the optimal sizes of the layers of the Elman network are relatively high which make them more difficult to train leading to less accurate results. The SVM performs best due to the use of a nonlinear kernel combined with the use of an ε-insensitive band which (partly) reduces the effect of the noise [16].

In order to investigate the dependency of the test set used on the prediction error, in total 25 consecutive test sets of size 200 are used to calculate the error for a prediction horizon of 5 time steps (Fig. 9). From this figure, it can be seen that there does exist a small

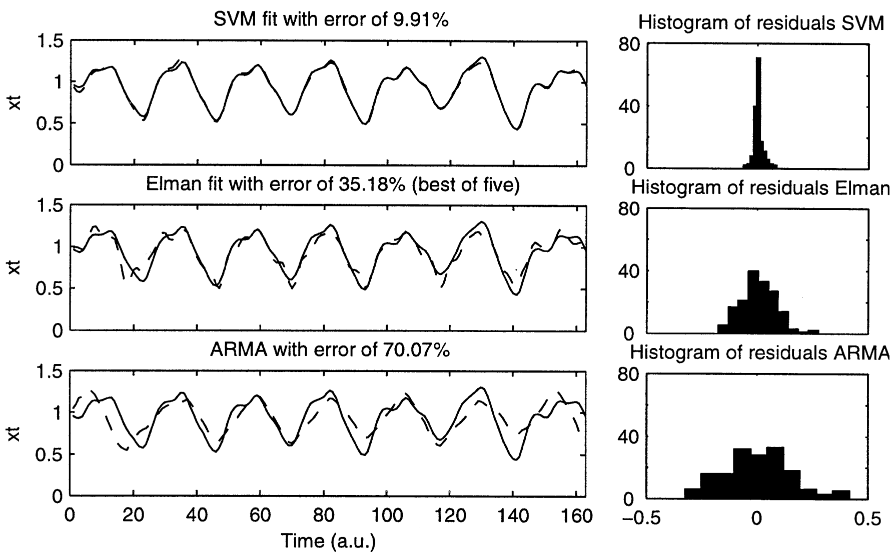

Fig. 10. Each row shows a part of the noise-free Mackey-Glass time series (solid line) with the predicted values (dashed line) together with the corresponding histogram of the prediction residuals. The best Elman network is the one with lowest overall error.

Table 3
The optimal settings found for the prediction of the Filter series

	Filter series		
	SVM	Elman	ARMA
Input window	15	7	14
Hidden neurons	–	1	–
Error terms	–	–	1
C	15	–	–
ε	0.0001	–	–
Kernel (width)	RBF (1.5)	–	–
TF hidden layer	–	Sigmoid	–
TF other layers	–	Linear	–

variation in prediction error for different test sets but this variation is not very large and the trend observed in Fig. 8 still remains. The variation in error is probably caused by the noise on the test sets because for the noise-free situation hardly any variation exists.

In order to show the quality of the fit of the time series at certain error percentages, Fig. 10 depicts a part of the Mackey–Glass time series together with the predicted values and the residual histograms. The results are obtained for the noise-free series with a prediction horizon of 20 time steps. It can be seen that a fit with an error of 35.18% still follows the shape of the series reasonably. Even with the ARMA prediction, the shape of the original series still is present but the deviation from the true values is high.

If a time series is identified well, the residuals should follow a normal distribution with a mean of *zero*. It can be seen that the residuals of the SVM more or less follow a narrow normal distribution around *zero*. The residuals of the Elman network and the ARMA model follow a much broader distribution which means that the errors are much higher.

4.3. Real-world data set

From the previous section, it appears that SVMs outperform both the ARMA model and the Elman networks. However, if no specific computational efficient training approaches are used, a disadvantage of the SVM method is that many calculations can be required to find the optimal solution. This can lead to unpractical long calculation times. This is also observed when using this real-world data set. For this reason a different approach was tested in which for the ARMA model and the Elman networks the complete training set was used (i.e. 3545 data points) while for the SVM only every tenth data point was used. This also allows us to test the accuracy of the SVM if only few data are present. The optimised and trained models have been used to predict the test filter series set in their original resolution. The optimal settings found are shown in Table 3. These have been found using a systematic grid search. For the SVM, the number of support vectors found is about 75%. Again, overtraining was excluded, due to the use of an independent optimisation set. Therefore, the relative high number of support vectors can be attributed to the use of a low resolution training set for predicting a test set in its original resolution.

In Fig. 11, the results are shown for predicting independent test filter series with a prediction horizon of 10 time steps. The prediction horizon for this case

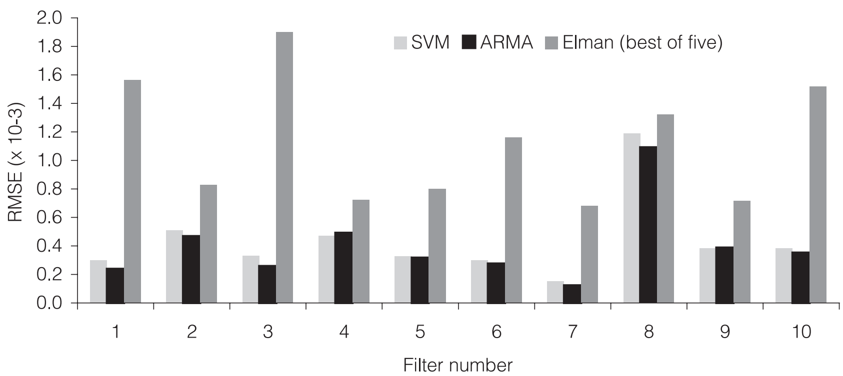

Fig. 11. The RMSE of the SVM (light grey), the ARMA model (black) and the best of five Elman networks (dark grey) for the prediction of 10 independent filter series.

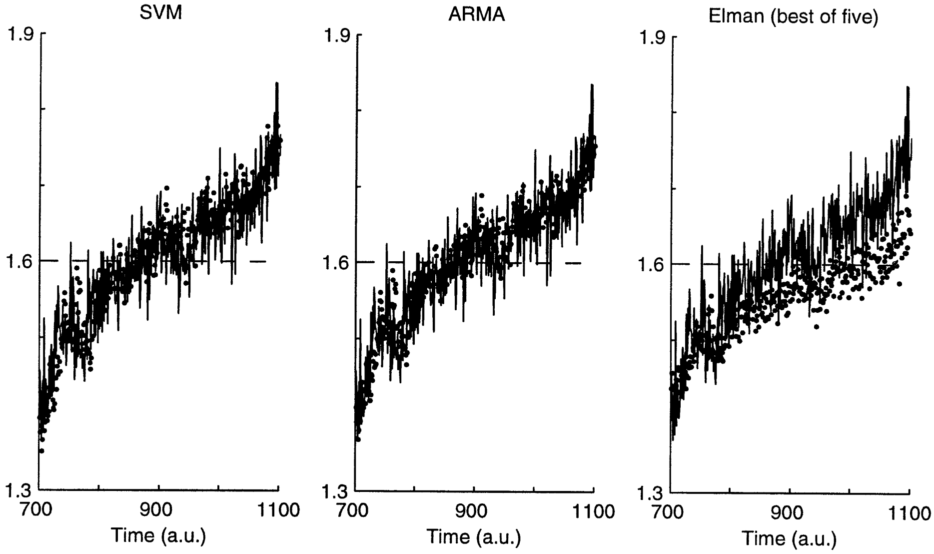

Fig. 12. A part of filter series 7 (solid line) together with the predicted values (represented by dots). The prediction horizon is 10 time steps. The fit is focussed on the time interval in which the critical limit (dashed line) is exceeded.

has been set in advance because it is considered to be appropriate for the specific problem. In fact, the used value is based on industrial expert knowledge. Increasing the horizon leads to a more or less similar decrease of the performances for all methods. It follows that the SVM performs comparable to the ARMA model even though a much smaller training set was used. However, the prediction accuracy of both the SVM and the ARMA model are satisfying. The Elman network performs much less than the other methods. This is probably caused by the typical shape of the filter series because it can be observed that the Elman predictions underestimate the true values in case of a large increase of the series. Furthermore, the Elman network also has problems in fitting the small curves in series that do not show a large increase in their values (such as series *D* in Fig. 6). In addition, the Elman networks' results show large variance in their predictions. A possible explanation is that Elman networks try to implicitly embed time correlation in the hidden layer. So, if the model is trained with two series with different time constants, the result can be an intermediate estimation which leads to inaccurate predictions for all series. In Fig. 12, filter series 7 (example *E* in Fig. 6) are shown for all prediction methods. It can be seen that the SVM and the ARMA model are able to predict the series in a similar and accurate way while the best Elman network cannot follow the true measurements.

If all methods were trained using only one tenth of the training set while predicting the test series in their original resolution, SVM outperformed the other methods (an RMSE which is up to a factor of 15 lower). This shows that SVMs can make models in a much more efficient way using less data than the other methods. Comparable observations have been made by Ref. [5]. For situations in which only few data are available, this can be very advantageous.

5. Discussion and conclusions

It has been shown in the literature that SVMs can perform very well on several tasks that are highly interesting for the field of chemometrics. In this paper, it has also been demonstrated that SVMs are a very good candidate for the prediction of time series. Three data sets have been considered which are ARMA time series, the chaotic and nonlinear Mackey–Glass data, and a real-world practice data set containing relative differential pressure changes of a filter.

As was expected, for the ARMA data set, the SVM and the Elman networks performed more or less equally. The system underlying this data set was based on the ARMA principles and this method performed best for this data set. This is expected because the ARMA model is able to build a parametric model similar to the underlying system. For the more difficult Mackey–Glass data set, the ARMA model was clearly outperformed by the SVM whereas the Elman network was able to predict the series in an intermediate way. In some cases, the SVM performed slightly worse than the best Elman network. When using the real-world data set, the problem was encountered that the training phase of the SVM was not feasible with this relatively large data set. This was caused because the training algorithm was implemented in a straightforward way without using more efficient training algorithms exploiting specific mathematical properties of the technique. However, when using a training set with a much lower resolution (i.e. 10%) the SVM was able to predict the filter series well and with similar results as the ARMA model which was trained with the full training set. The Elman networks were not able to perform well on this data set. For this data set, it can be concluded that the SVM can build qualitatively good models even with much less training objects.

The largest benefits of the SVMs are the facts that a global solution exists and is found in contrast to neural networks which have to be trained with randomly chosen initial weight settings. Furthermore, due to the specific optimisation procedure it is assured that overtraining is avoided and the SVM solution is general. No extra validation set has to be used for this task as is the case with the Elman network. Additionally, the trained SVM decision function can be evaluated relatively easy because of the reduced number of training data that contribute to the solution (the support vectors). A drawback of the SVMs is the fact that the training time can be much longer than for the Elman network and the ARMA model (up to a factor of 100 in this paper). More recently, efficient training algorithms have been developed such as Sequential Minimal Optimisation (SMO) [17], Nyström method [18], Cholesky factorisation [19], or methods that decompose the optimisation problem [20]. Another alternative to standard SVMs is the use of least-squares SVMs that reformulate the optimisation problem leading to solving a set of linear equations that can be solved easier [9,21].

In general, for each data set and each situation, it is advised to optimise the prediction method used and to perform a small comparison with other prediction methods if possible. If several methods perform in a comparable way, the most simple, fast or most diagnosing method is preferred above the other prediction methods. For this reason, the examples shown in this paper should be considered as illustrative rather than exhaustive.

Acknowledgements

The authors would like to thank Paul Lucas and Micha Streppel for their literature studies on support vector machines and recurrent neural networks, respectively. Furthermore, Erik Swierenga (Acordis Industrial Fibers), Sijmen de Jong (Unilever) and Stan Gielen (University of Nijmegen) are thanked for giving useful comments. This project is financed by STW (STW-Project NCH4501).

References

[1] A. Rius, I. Ruisánchez, M.P. Callao, F.X. Rius, Reliability of analytical systems: use of control charts, time series models and recurrent neural networks (RNN), Chemometrics and Intelligent Laboratory Systems 40 (1998) 1–18.

[2] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (1998) 121–167.

[3] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London, UK, 1998. Available on http://www.kernel-machines.org/.

[4] K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, V. Vapnik, Predicting time series with support vector machines, in: W. Gerstner, A. Germond, M. Hasler, J.-D. Nicoud (Eds.), Proceedings of ICANN '97, Springer LNCS 1327, Berlin, 1997, pp. 999–1004.

[5] A.I. Belousov, S.A. Verzakov, J. Von Frese, A flexible classification approach with optimal generalisation performance: support vector machines, Chemometrics and Intelligent Laboratory Systems 64 (2002) 15–25.

[6] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge Univ. Press, Cambridge, UK, 2000.

[7] C. Campbell, Kernel methods: a survey of current techniques, Neurocomputing 48 (2002) 63–84.

[8] B. Schölkopf, A.J. Smola, Learning with Kernels, MIT Press, Cambridge, UK, 2002.

[9] J.A.K. Suykens, T. van Gestel, J. de Brabanter, B. de Moor,

J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002.

[10] A. Krogh, J.A. Hertz, A simple weight decay can improve generalization, in: J.E. Moody, S.J. Hanson, R.P. Lippman (Eds.), Advances in Neural Information Processing Systems, vol. 4, Morgan Kauffmann Publishers, San Mateo, 1995, pp. 950–957.

[11] G.E.P. Box, G.M. Jenkins, Time Series Analysis, Forecasting and Control, Holden-Day, Oakland, CA, USA, 1976, Revised version.

[12] C. Chatfield, The Analysis of Time Series: An Introduction, 2nd ed., Chapman & Hall, London, UK, 1980.

[13] D.P. Mandic, J.A. Chambers, Recurrent Neural Networks for Prediction, Wiley, Chichester, UK, 2001.

[14] M.C. Mackey, L. Glass, Oscillation and chaos in physiological control systems, Science 197 (1977) 287–289.

[15] S.R. Gunn, Support vector machines for classification and regression, Technical report, Image speech and intelligent systems research group, University of Southampton, UK, 1997. Available on http://www.isis.ecs.soton.ac.uk/isystems/kernel/.

[16] D. Mattera, S. Haykin, Support vector machines for dynamic reconstruction of a chaotic system, in: B. Schölkopf, C.J.C. Smola, A.J. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, 1999, pp. 243–254.

[17] J. Platt, Flat training of support vector machines using sequential minimal optimisation, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, 1999, pp. 185–208.

[18] C.K.I. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, vol. 13, MIT Press, Cambridge, 2001, pp. 682–688.

[19] S. Fine, K. Scheinberg, Efficient SVM training using low-rank kernel representation, Journal of Machine Learning Research 2 (2001) 243–264.

[20] T. Joachims, Making large-scale support vector machines learning practical, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, 1999, pp. 169–184.

[21] J.A.K. Suykens, J. Vandewalle, Least squares support vector machines, Neural Processing Letters 9 (1999) 293–300.