Analysis of nonstationary time series using support vector machines

Ming-Wei Chang[1], Chih-Jen Lin[1], and Ruby C. Weng[2]

[1] Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan,
cjlin@csie.ntu.edu.tw,
WWW home page: http://www.csie.ntu.edu.tw/~cjlin

[2] Department of Statistics, National Chengchi University Taipei 116, Taiwan

Abstract. Time series from alternating dynamics have many important applications. In [5], the authors propose an approach to solve the drifting dynamics. Their method directly solves a non-convex optimization problem. In this paper, we propose a strategy which solves a sequence of convex optimization problems by using modified support vector regression. Experimental results showing its practical viability are presented and we also discuss the advantages and disadvantages of the proposed approach.

1 Introduction

Time series from dynamics systems arise from different real-world applications. It is therefore important to find algorithms which can solve this kind of problems. In this paper, we are interested in drifting dynamics model. That is, at some time steps, the series is a mixture of several operating modes. Our task is to find the weight of each mode.

This problem has been studied in, for example, [6, 7, 5]. Among them, [5] is a straightforward approach for the analysis of switching and drifting dynamics, where a non-convex global optimization problem is solved. Here, we consider the same formulation but tackle it by solving a sequence of convex problems. To be more precise, we hope to separate the mixed series to different functions (modes) and obtain their weights that reflect how series are mixed. Therefore, we propose an iterative procedure which alters in approximating the function of each mode and updating its weight.

To approximate the function of each mode, we consider a modified support vector regression (SVR) formulation. Recently support vector machine (SVM) [11] has been a promising method for data classification and regression. However, its use in time series modeling has not been exploited much. An earlier work using SVR for time series segmentation is [3] which considers a simpler problem of switching dynamics.

This paper is organized as follows. In Section 2, we define the problem and present our approach. Section 3 demonstrates experimental results on some data sets. Discussion about the advantages and disadvantages of the proposed method is in Section 4.

2 An Algorithm for Drifting Dynamics

Given a stationary time series $\{y_t\}$, usually we can model it by using $y_t = f(x_t)$, where $x_t = [y_{t-1}, y_{t-2}, \ldots, y_{t-d}]^T$, and d is the embedding dimension. However, this is not true if this time series is non-stationary. For example, if the time series is from switching dynamics, we can not determine one unique f to satisfy the whole sequence. Instead, several functions may have to be used. If m is the number of modes in this system, then we find out $f_1, \ldots, f_m$ for modeling the time series. That is,

$$y_t = f_{r_t}(x_t), \forall t = 1, \ldots, l, \tag{1}$$

where l is the length of the time series. The index $r_t \in \{1, \ldots, m\}$ means functions alter according to r_t. Many papers have tackled this issue by using neural networks. See [10, 9, 4] and references therein. Recently an approach using support vector machines is [3].

One more difficult problem is how to model time series from drifting dynamics. For this type of time series, y_t can not be determined by one single function but a mixture of several functions. We can assume that there are m functions such that

$$y_t = \sum_{i=1}^{m} p_i^t f_i(x_t), \forall t = 1, \ldots, l, \tag{2}$$

with

$$\sum_{i=1}^{m} p_i^t = 1, \forall t = 1, \ldots, l$$

and

$$0 \leq p_i^t \leq 1, \forall t = 1, \ldots, l, i = 1, \ldots, m.$$

This kind of problems is very complicated because of the high degrees of freedom in the formula (2). In other words, there are simply too many possible choices of p_i^t and $f_i(x_t)$. In [5], when proposing an approach for drifting dynamics, the authors had to make some assumptions such that the problem is easier. In this paper, we will follow these assumptions–for example, we assume that the number of functions is known a priori. Furthermore, we consider problems where for each i, $p_i^t, t = 1, \ldots, l$, is slowly changing.

If we take this problem as an optimization issue, then it has the following form:

$$\min_{p_i^t, f_i} \sum_{t=1}^{l} (y_t - \sum_{i=1}^{m} p_i^t f_i(x_t))^2. \tag{3}$$

Unfortunately, (3) is a difficult global optimization problem. In [5], the authors use Nadaraya-Watson kernel estimators to model each f_i:

$$f_i(x_t) = \frac{\sum_{k=1}^{l} y_k K(x_k, x_t)(p_i^k)^2}{\sum_{k=1}^{l} K(x_k, x_t)(p_i^k)^2}, \tag{4}$$

where p_i^k become the only variables. $K(x_k, x_t) = e^{-\gamma\|x_k - x_t\|^2}$ is usually called the RBF kernel. Then they solve a complex non-convex problem:

$$\begin{aligned} \min_{p_i^t} \quad & \sum_{t=1}^{l}(y_t - \sum_{i=1}^{m} p_i^t f_i(x_t))^2 + \lambda_1 \sum_{t=2}^{l}\sum_{i=1}^{m}(p_i^t - p_i^{t-1})^2 \\ \text{subject to} \quad & \sum_{i=1}^{m} p_i^t = 1, \\ & 0 \le p_i^t \le 1, t = 1, \ldots, l, i = 1, \ldots, m, \end{aligned} \tag{5}$$

with f_i defined in (4). Here, λ_1 serves as a penalty parameter so that p_i^t and p_i^{t-1} are close. This reflects that $p_i^t, t = 1, \ldots, l$ are slowly changing.

Here, we propose to separate (3) into two easier parts: modeling f_i and updating the weights p_i^t.

At first, we initialize each p_i^t with a random value in $[0, 1]$ while keeping $\sum_{i=1}^{m} p_i^t = 1$. Then, we fix p_i^t and model f_i of the system. After obtaining an approximate f_i, we then fix each $f_i(x_t)$ to find the best p_i^t. We repeat this process until some stopping criteria are satisfied.

2.1 Modeling the Dynamical System

Unlike [5] which used Nadaraya-Watson kernel estimators, in this paper, we use a modified support vector regression (SVR) formulation to model the dynamical system. In particular, we approximate each $f_i(x)$ with a linear function $w_i^T \phi(x) + b_i$, where x is mapped to a higher dimensional space by ϕ.

Note that the original formulation of support vector regression is:

$$\begin{aligned} \min_{w,b,\xi,\xi^*} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \text{subject to} \quad & y_i - (w^T \phi(x_i) + b) \le \epsilon + \xi_i, \\ & (w^T \phi(x_i) + b) - y_i \le \epsilon + \xi_i^*, \\ & \xi_i, \xi_i^* \ge 0, i = 1, \ldots, l, \end{aligned} \tag{6}$$

where ξ_i is the upper training error (ξ_i^* is the lower) subject to the ϵ-insensitive tube $|y - (w^T \phi(x) + b)| \le \epsilon$. The constraints imply that we would like to put most data x_i in the tube $|y - (w^T \phi(x) + b)| \le \epsilon$. If x_i is not in the tube, there is an error ξ_i or ξ_i^* which we would like to minimize in the objective function. For traditional regression, ϵ, is always zero and data are not mapped into higher dimensional spaces. Hence, SVR is a more general and flexible treatment on regression problems. Note that SVR uses a linear loss function $\sum_{i=1}^{l}(\xi_i + \xi_i^*)$ instead of the quadratic one: $\sum_{i=1}^{l}(\xi_i^2 + (\xi_i^*)^2)$. The parameters which control the regression quality are the cost of error C, the width of the tube ϵ, and the mapping function ϕ.

Next, we discuss how to modify SVR for our purpose. In (5), if p_i^t are fixed scalars, it reduces to an unconstrained problem:

$$\min_{f_i} \sum_{t=1}^{l} (y_t - \sum_{i=1}^{m} p_i^t f_i(x_t))^2. \tag{7}$$

With $f_i(x) = w_i^T \phi(x) + b_i$, $\frac{1}{2} w_i^T w_i, i = 1, \ldots, m$, as regularization terms, and the linear ϵ-insensitive loss function, (7) is extended to

$$\begin{aligned} \min_{w_i, b_i, \xi, \xi^*} \quad & \frac{1}{2} \sum_{i=1}^{m} w_i^T w_i + C \sum_{t=1}^{l} (\xi_t + \xi_t^*) \\ \text{subject to} \quad & -\epsilon - \xi_t^* \leq \sum_{i=1}^{m} p_i^t (w_i^T \phi(x_t) + b_i) - y_t \leq \epsilon + \xi_t, \\ & \xi_t \geq 0, \xi_t^* \geq 0, t = 1, \ldots, l. \end{aligned} \tag{8}$$

From (7) to (8), an important difference is the add of regularization terms. They avoid data overfitting and smooth the functions.

Note that $\sum_{i=1}^{m} p_i^t b_i$ can be considered as a single variable b. Then with the condition $\sum_{i=1}^{m} p_i^t = 1$, we can simply have $b_i = b, i = 1, \ldots, m$.

Usually we solve SVR through its dual. Here, we also derive the dual of (8) through its Lagrangian:

$$\begin{aligned} L = & \frac{1}{2} \sum_{i=1}^{m} w_i^T w_i + C \sum_{t=1}^{l} (\xi_t + \xi_t^*) \qquad (9) \\ & - \sum_{t=1}^{l} \alpha_t (\epsilon + \xi_t + y_t - \sum_{i=1}^{m} p_i^t w_i^T \phi(x_t) - b) \\ & - \sum_{t=1}^{l} \alpha_t^* (\epsilon + \xi_t^* - y_t + \sum_{i=1}^{m} p_i^t w_i^T \phi(x_t) + b) \\ & - \sum_{t=1}^{l} \mu_t \xi_t - \sum_{t=1}^{l} \mu_t^* \xi_t^*, \end{aligned}$$

where α, α^*, μ, and μ^* are nonnegative Lagrangian multipliers.

By calculating the gradient of L on w_i, we have

$$
\begin{aligned}
w_i &= -\sum_{t=1}^{l} \alpha_t p_i^t \phi(x_t) + \sum_{t=1}^{l} \alpha_t^* p_i^t \phi(x_t) \\
&= \sum_{t=1}^{l} (-\alpha_t + \alpha_t^*) p_i^t \phi(x_t). \qquad (10)
\end{aligned}
$$

In addition,

$$
\frac{\partial L}{\partial \xi_t} = 0 = C - \alpha_t - \mu_t, \qquad (11)
$$

$$
\frac{\partial L}{\partial \xi_t^*} = 0 = C - \alpha_t^* - \mu_t^*, \qquad (12)
$$

$$
\frac{\partial L}{\partial b} = \sum_{t=1}^{l} \alpha_t - \sum_{t=1}^{l} \alpha_t^* = 0. \qquad (13)
$$

Using (10),

$$
\begin{aligned}
& w_i^T w_i \\
&= (\alpha - \alpha^*)^T \begin{bmatrix} p_i^1 & & \\ & \ddots & \\ & & p_i^l \end{bmatrix} K \begin{bmatrix} p_i^1 & & \\ & \ddots & \\ & & p_i^l \end{bmatrix} (\alpha - \alpha^*),
\end{aligned}
$$

where K is an l by l matrix and $K_{ij} = \phi(x_i)^T \phi(x_j)$ is the kernel function. By combining (10)-(13) back to (9), the dual problem is

$$
\begin{aligned}
\min_{\alpha, \alpha^*} \quad & \frac{1}{2}(\alpha - \alpha^*)^T Q (\alpha - \alpha^*) + \epsilon e^T (\alpha + \alpha^*) + y^T (\alpha - \alpha^*) \\
\text{subject to} \quad & \sum_{t=1}^{l} \alpha_t - \sum_{t=1}^{l} \alpha_t^* = 0, \qquad (14) \\
& 0 \leq \alpha_t, \alpha_t^* \leq C, t = 1, \ldots, l,
\end{aligned}
$$

where e is the vector of all ones, $y \equiv [y_1, \ldots, y_l]^T$, and

$$
Q = \sum_{i=1}^{m} \operatorname{diag}(p_i) K \operatorname{diag}(p_i).
$$

Here, we denote $p_i \equiv [p_i^1, \ldots, p_i^l]^T$ and $\operatorname{diag}(p_i)$ is a diagonal matrix with elements of p_i on the diagonal. Immediately, we also know that (14) is still a convex optimization problem. We can further prove that if K is positive definite, then (14) is a strictly convex problem:

Theorem 1. *If K is positive definite, then so is Q.*

Proof. Since Q is positive semi-definite, if the result is wrong, then there exists a vector $v \neq 0$ such that

$$v^T Q v = 0. \tag{15}$$

Using the definition of Q, we assume $v = [v^1, \ldots, v^l]^T$, so

$$v^T Q v = \sum_{i=1}^{m} v_i^T K v_i,$$

where

$$v_i = [v_i^1, \ldots, v_i^l]^T \text{ and } v_i^t \equiv v^t p_i^t, \forall t = 1, \ldots, l.$$

Since K is positive definite, (15) implies

$$v_i = 0, i = 1, \ldots, m.$$

Therefore,

$$v^t = v^t \sum_{i=1}^{m} p_i^t = \sum_{i=1}^{m} v_i^t = 0, \forall t = 1, \ldots, l.$$

Then $v = 0$ contradicts our assumption. □

After α and α^* are obtained, our approximate functions are

$$f_i(x) = \sum_{t=1}^{l} p_i^t(-\alpha_t + \alpha_t^*)K(x_t, x) + b.$$

We modify LIBSVM [2], a library for support vector machines, to solve the new formulation (14).

2.2 Updating p_i^t

Based on the current models (i.e., $f_i, i = 1, \ldots, m$), we then find the best p_i^t. This is via fixing $f_i(x_t)$ in (5) and solve an optimization problem with variables p_i^t. To be more precise, it has the same form as (5) except that in (5), $f_i(x_t)$ is a function of p_i^t but here it is a constant. Then the optimization problem becomes convex. To handle the linear constraint $\sum_{i=1}^{m} p_i^t = 1$, we add a penalty term in the objective function so an optimization problem with only bounded constraints is obtained:

$$\begin{aligned} \min_{p_i^t} \quad & \sum_{t=1}^{l}(y_t - \sum_{i=1}^{m} p_i^t f_i(x_t))^2 + \lambda_1 \sum_{t=2}^{l}\sum_{i=1}^{m}(p_i^t - p_i^{t-1})^2 + \\ & \lambda_2 \sum_{t=1}^{l}(1 - \sum_{i=1}^{m} p_i^t)^2 \qquad (16) \\ \text{subject to} \quad & 0 \leq p_i^t \leq 1, \forall t = 1, \ldots, l, i = 1, \ldots, m, \end{aligned}$$

where λ_2 is a coefficient for the penalty term.

We choose the software TRON by Lin and Moré [8] to find the optimal solution. TRON implements a Newton's method which requires the first order (gradient) and the second order (Hessian) information. Therefore, we work on the AMPL interface [1] of TRON which can automatically generate and use the gradient and Hessian provided the objective function is available.

2.3 Stopping Criterion

A stopping criterion is needed for the proposed algorithm. We define the error function as

$$\sum_{t=1}^{l}(y_t - \sum_{i=1}^{m} p_i^t f_i(x_t))^2.$$

Then we consider the following stopping condition:

$$\frac{|e\hat{r}r - err|}{|err|} \leq 0.02,$$

where $e\hat{r}r$ and err are errors of two consecutive iterations.

3 Experiments

The same examples [5] are used here to illustrate our approach. We generate time series from drifting dynamics using the Mackey-Glass delay differential equation:

$$\frac{dx(t)}{dt} = \gamma_{t_d} = -0.1x(t) + \frac{0.2x(t-t_d)}{1+x(t-t_d)^{10}}.$$

The first example consists of five segments with three stationary modes and two non-stationary (mixture) modes. The three stationary modes are established by using time delay t_d to be 17, 23, and 30, respectively. The mixture is generated as follow:

$$\frac{dx(t)}{dt} = a\gamma_{17} + b\gamma_{23} + c\gamma_{30}. \tag{17}$$

First we obtain 100 points by using the stationary mode with $t_d = 17$, and then another 100 points as mixtures of three modes by using (17) with $a = 0.6, b = 0.3$, and $c = 0.1$. Next, we generate 100 points by using the stationary mode with $t_d = 23$ and then 100 points by mixing three modes with $a = 0.2, b = 0.3$, and $c = 0.5$. Finally, for $t = 401, \ldots, 500$, it is the stationary mode with $t_d = 30$.

We then apply our iterative method on this data set. For the modified SVR, we set $C = 0.5$ and $\epsilon = 0$. The RBF kernel $K(x_i, x_j) = e^{-\gamma\|x_i - x_j\|^2}$ is used with

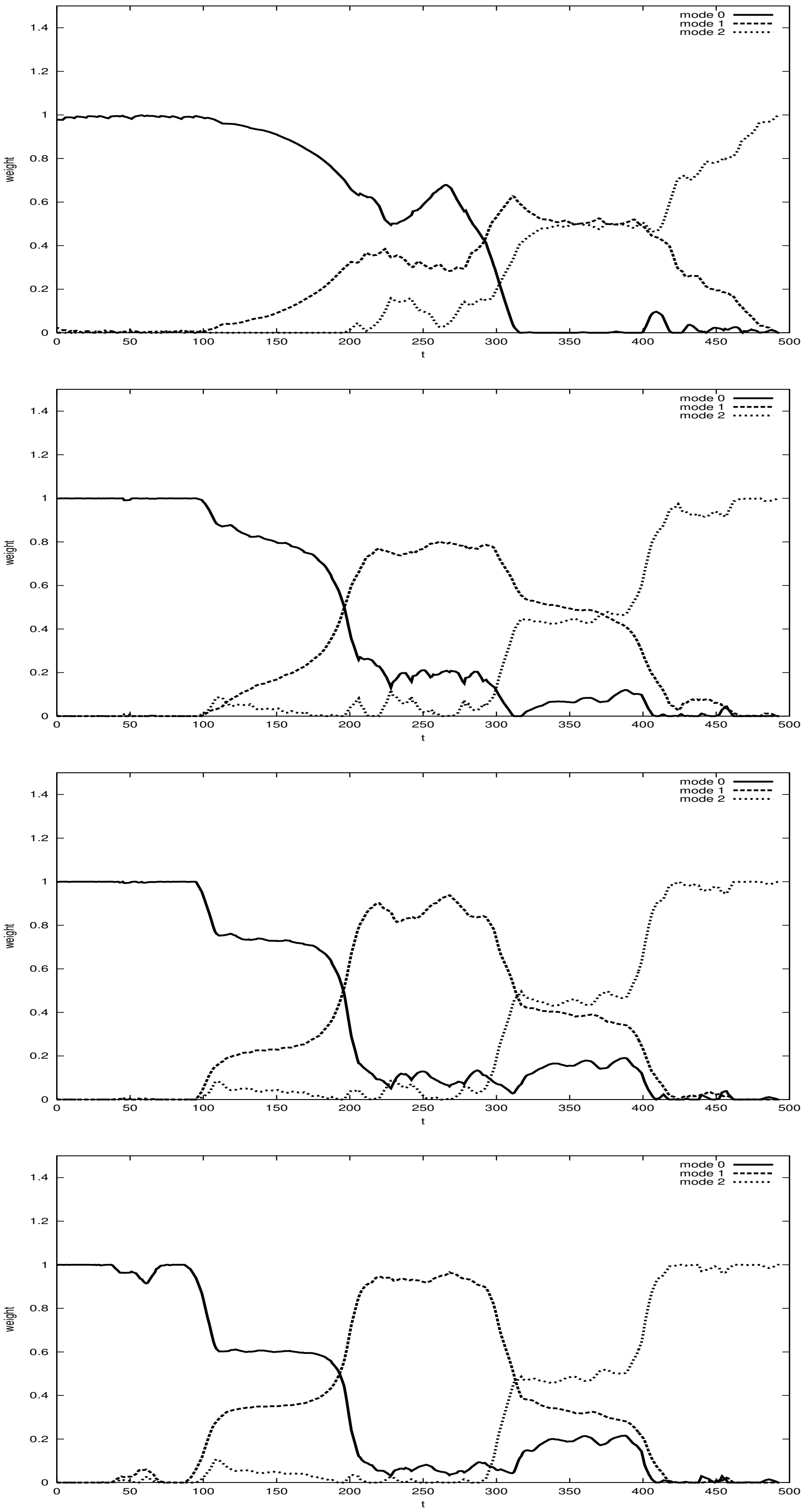

Fig. 1. First, second, third, and final iterations: three mixture modes in five segments

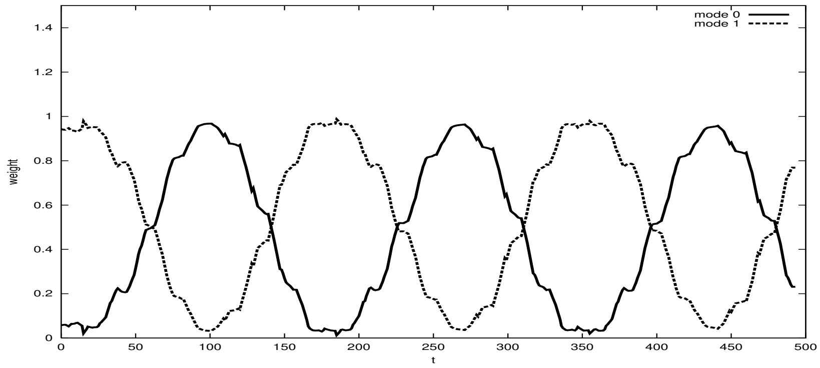

Fig. 2. Final iteration of the second experiment: continuous drifting systems

$\gamma = 1.0$. In (16), we set $\lambda_1 = 0.5$ and $\lambda_2 = 100$. The embedding dimension is six. That is, y_t is the one-step ahead value of $x_t = [y_{t-1}, \ldots, y_{t-6}]^T$.

The procedure of our approach is in Figure 1. Each sub-figure presents the weights of three modes. Typically the number of iterations is ten and we show only the first, second, third, and the final iterations due to space limitation. We can see that our algorithm works very well for stationary parts and is reasonably well for $t = 101$ to 200. We fail to precisely solve the period $t = 301$ to 400 and this situation is the same as [5]. Nevertheless, the order of the weights of three modes is still correct.

For the second experiment we consider continuous drifting dynamics where there are no stationary parts anymore. Using (17), we generate 500 points by setting $a = 0.5 + 0.5\sin(\pi t/100)$, $b = 1 - a$, and $c = 0$. That is, two modes are mixed together. To solve this problem, we must modify one parameter of (16) since the changing rate of weights is higher. We set a smaller $\lambda_1 = 0.03$ and keep all other parameters the same. The final result is in Figure 2. It can be seen the our method easily separates the mixed series. Interestingly this continuous drifting case is easier to be solved. For the previous example, as weight are not continuous at $t = 100, 200, 300, and 400$, the penalty term $(p_i^t - p^(t-1)_i)^2$ of (16) may not be very appropriate at these points.

4 Discussion

4.1 Avoid Local Minima

The major advantage of our algorithm is that we iteratively solve easier convex optimization problems. Unfortunately, as the goal is still to solve a non-convex problem, sometimes our algorithm can fall into local minima. A possible solution is to consider in the beginning that series are not mixed but only switching dynamics. Then by the competition of experts using approaches in [3, 10, 9, 4], we obtain a better initial set of p_i^t. Then the algorithm is less likely to be trapped into local minima.

4.2 Scaling Models (Functions)

Another possible problem of our approach is that $f_i(x)$ may have values in a larger range than those of y_t provided. To make $f_i(x)$ in the same range of $y_t, t = 1, \ldots, l$, we can linearly scale $[f_i(x_1), \ldots, f_i(x_l)]$ back to the interval $[y_{\min}, y_{\max}]$ where

$$y_{\min} \equiv \min_{t=1,\ldots,l} y_t \text{ and } y_{\max} \equiv \max_{t=1,\ldots,l} y_t.$$

Then the scaled values are used for updating p_i^t.

For the above two examples, our approach already works without such mechanisms. It is interesting to see [5] inherently conducts this scaling. In (4), $f_i(x_k)$ is actually a weighted average of $y_1, \ldots, y_l$. As $K(x_k, x_t)(p_i^k)^2, k = 1, \ldots, l$ are all positive, $f_i(x_t)$ must be in $[y_{\min}, y_{\max}]$.

4.3 Comparisons with Approaches for Time Series from Switching Dynamics

In an earlier work [3] we consider time series which alter in time. That is, each y_t is related to only one model (see (1)). We also use p_i^t to indicate the weight that the tth point is associated with the ith function. For each time point t, only one of the true $p_i^t, i = 1, \ldots, m$, is one and the others are zero. Hence, the optimization problem considered is

$$\min_{p_i^t, f_i} \sum_{i=1}^{m} \sum_{t=1}^{l} p_i^t (y_t - f_i(x_t))^2.$$

When p_i^t are fixed, we employ SVR to approximate f_i:

$$\begin{aligned} \min_{w_i, b_i} \quad & \frac{1}{2} w_i^T w_i + C \sum_{t=1}^{l} p_i^t (\xi_i^t + \xi_i^{t,*}) \\ \text{subject to} & -\epsilon - \xi_i^{t,*} \leq y_t - (w_i^T \phi(x_t) + b_i) \leq \epsilon + \xi_i^t, \\ & \xi_i^t \geq 0, \xi_i^{t,*} \geq 0, t = 1, \ldots, l, \end{aligned} \tag{18}$$

Therefore, m SVRs have to be solved. If the changing rate is slow, given Δ, $x_{t-\Delta}, \ldots, x_t, \ldots, x_{t+\Delta}$ should be with the same series. Based on this property, there is a method to update p_i^t.

Note that (7) can also be considered as the formulation for such time series as finally we require one $p_i^t = 1$ and others are zero. Then following the approach in this paper, there is an advantage that only one SVR is required in each iteration. Note that dual problems of (18) and (7) both have $2l$ variables. If m SVRs (18) are replaced by (7) and the same rule for updating p_i^t in [3] is used, the solution quality is nearly the same. However, if this particular rule for updating p_i^t is not used and instead, (7) and (16), i.e., the whole approach proposed in this paper, are used, results are less stable. This clearly shows that time series from switching dynamics is much simpler than the drifting case discussed in this paper. For the non-drifting case, the property that eventually only one of $p_1^t, \ldots, p_m^t$ is one provides useful information and makes the problem easier.

4.4 Future Directions

We hope to develop a simple way for updating p_i^t. Then, the optimization problem in Section 2.2 is not needed.

Acknowledgments

This work was supported in part by the National Science Council of Taiwan via the grant NSC 90-2213-E-002-111. The authors thank J. Kohlmorgen who kindly provided us the code for generating non-stationary series using the Mackey-Glass equation.

References

[1] NEOS AMPL interfaces to TRON. Software available at `http://neos.mcs.anl.gov/neos/solvers/BCO:TRON-AMPL/`.

[2] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[3] M.-W. Chang, C.-J. Lin, and R. C. Weng. Analysis of switching dynamics with competing support vector machines. In *Proceedings of IJCNN*, 2002.

[4] A. Kehagias and V. Petridis. Time-series segmentation using predictive modular neural networks. *Neural Computation*, 9:1691–1709, 1997.

[5] J. Kohlmorgen, S. Lemm, G. Rätsch, and K.-R. Müller. Analysis of nonstationary time series by mixtures of self-organizing predictors. In *Proceedings of IEEE Neural Networks for Signal Processing Workshop*, pages 85–94, 2000.

[6] J. Kohlmorgen, K.-R. Müller, and K. Pawelzik. Segmentation and identification of drifting dynamical systems. pages 326–335, 1997.

[7] J. Kohlmorgen, K.-R. Müller, J. Rittweger, and K. Pawelzik. Identification of nonstationary dynamics in physiological recordings. *Biological Cybernetics*, 83:73–84, 2000.

[8] C.-J. Lin and J. J. Moré. Newton's method for large-scale bound constrained problems. *SIAM J. Optim.*, 9:1100–1127, 1999.

[9] K.-R. Müller, J. Kohlmorgen, and K. Pawelzik. Analysis of switching dynamics with competing neural networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E78–A(10):1306–1315, 1995.

[10] K. Pawelzik, J. Kohlmorgen, and K.-R. Müller. Annealed competition of experts for a segmentation and classification of switching dynamics. *Neural Computation*, 8(2):340–356, 1996.

[11] V. Vapnik. *Statistical Learning Theory.* Wiley, New York, NY, 1998.