International Journal of Applied Science and Engineering
2005. 3, 1: 51-59

Using Least Squares Support Vector Machines for Adaptive Communication Channel Equalization

Cheng-Jian Lin[a*], Shang-Jin Hong[a], and Chi-Yung Lee[b]

[a]*Department of Computer Science and Information Engineering,*
Chaoyang University of Technology,
Wufong, Taichung County 41349, Taiwan, R.O.C.
[b]*Department of Computer Science and Information Engineering,*
Nankai Institute of Technology,
Caotun, Nantou County 542, Taiwan, R.O.C.

Abstract: Adaptive equalizers are used in digital communication system receivers to mitigate the effects of non-ideal channel characteristics and to obtain reliable data transmission. In this paper, we adopt least squares support vector machines (LS-SVM) for adaptive communication channel equalization. The LS-SVM involves equality instead of inequality constraints and works with a least squares cost function. Since the complexity and computational time of a LS-SVM equalizer are less than an optimal equalizer, the LS-SVM equalizer is suitable for adaptive digital communication and signal processing applications. Computer simulation results show that the bit error rate of the LS-SVM equalizer is very close to that of the optimal equalizer and better than multilayer perceptron (MLP) and wavelet neural network (WNN) equalizers.

Keywords: digital communication; adaptive equalizer; support vector machines; time-varying channel; kernel function.

1. Introduction

In digital communication systems, nonlinear distortion is a major factor that limits the performance of a communication system. Channel inter-symbol interference (ISI), additive white Gaussian noise (AWGN) [1-2] and effects of time-varying channels [3] can severely corrupt transmitted symbols. Nevertheless, adaptive equalizers can be used in digital communication system receivers to mitigate the effects of non-ideal channel characteristics and to obtain reliable data transmission. The process of equalization is well-known for being able to reconstruct transmitted symbols based on observations of the corrupted channel. Equalization is treated as a natural inverse filter, and the equalizer forms an approximation of the inverse of the distorting channel. Due to noise enhancement, accurate approximation and better equalization performance can not be achieved.

Among all the different techniques, the methods that are based on multilayer perceptron (MLP) and wavelet neural networks (WNNs) have become popular research topics in recent years [4-5]. Gibson et al. [6] proposed an adaptive equalizer based on MLPs. The MLP structure is less sensitive to learning gain variation and capable of converging to a lower mean square error. Despite providing considerable performance improvements,

* Corresponding author: e-mail: cjlin@cyut.edu.tw
Accepted for Publication: April 7, 2005

MLP equalizers are still problematic because of their convergence performance and complex structure.

Wavelet neural networks (WNNs) originates from wavelet decomposition in signal processing [7-8]. WNNs combine the capability of artificial neural networks to learn from processes with the capability of wavelet decomposition. Wavelets are a class of basic elements with oscillations of effectively finite duration that makes them look like "little waves." The self-similar, multiple resolution nature of wavelets offers a natural framework. The main characteristics of WNNs are their capability of incorporating the time-frequency localization properties of wavelets and with the learning abilities of general neural networks; hence, a WNN is usually applied to the identification and control of dynamic systems and complex nonlinear system modeling.

Support vector machines (SVM) [9] and kernel methods [10-12] have become more and more spectacular. The SVM has good performance and is used widely in various applications, such as image recognition [15], DNA data analysis [16], classification [17], and control problems [18]. In the SVM, the inputs are non-linearly mapped onto a very high dimension feature space. In the feature space, a linear decision surface is constructed, and a special property of the decision surface ensures high generalization ability of the learning machine. The SVM typically follows from the solution to a quadratic programming (QP) problem.

In this paper, we adopt least squares support vector machines (LS-SVM) for adaptive communication channel equalization. The LS-SVM involves equality instead of inequality constraints and works with a least squares cost function [11]. It is more suitable for adaptive communication and signal processing applications because a solution to the algorithm can be obtained by solving a set of linear equations instead of solving a QP problem. For this reason, the LS-SVM can be implemented by daptive on–line algorithms.

This paper is organized as follows. Sections 2 and 3 describe the structure and functionality of SVM and LS-SVM. The simulation results are presented in Section 4. We will apply the LS-SVM equalizer to time-invariant and time-varying channel problems. Comparisons of the LS-SVM, Bayesian, MLP, and WNN equalizers are shown in this section. Finally, the conclusion is presented in Section 5.

2. Support vector machines

In this section, we briefly introduce the standard SVM for binary classification problems. The structure of the standard SVM is shown in Figure 1.

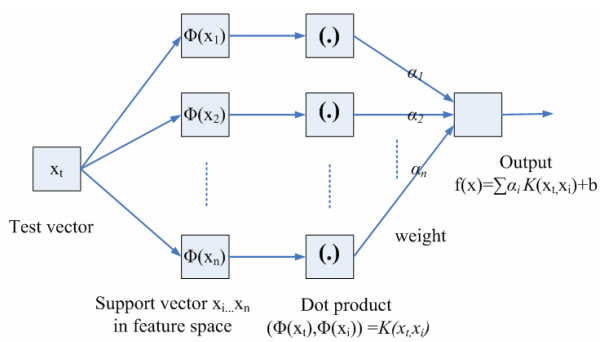

Figure 1. The architecture of the standard SVM model

Support vector machines (SVM) have been successfully applied in classification and function estimation problems [12] after their introduction by Vapnik within the context of statistical learning theory and structural risk minimization [14]. Vapnik constructed the standard SVM to separate training data into two classes. The goal of the SVM is to find the hyper-plane that maximizes the minimum distance between any data point.

The purpose for training a SVM is to find the classifier $f(x) = \omega \cdot x + b$ so that we can use the classifier to classify data. The training set $\{x_i, y_i\}_{i=1}^{l}$, where $x_i \in \Re^n$ is the input and $y_i \in \{-1,+1\}$ is the output, indicates the class.

SVM formulations start from the assumption that the linear separable case is

$$\begin{cases} \omega^T x_i + b \geq +1, & \text{if } y_i = +1 \\ \omega^T x_i + b \leq -1, & \text{if } y_i = -1 \end{cases} \tag{1}$$

For the non-separable case,

$$\begin{cases} \omega^T \phi(x_i) + b \geq +1, & \text{if } y_i = +1 \\ \omega^T \phi(x_i) + b \leq -1, & \text{if } y_i = -1 \end{cases} \tag{2}$$

where $\phi(\cdot)$ denotes a mapping of the input onto a so-called higher dimensional feature space.

In this space, a linear decision surface is constructed with special properties that ensure high generalization ability of the network. Its diagram is shown in Fig. 2. By use of a nonlinear kernel function, it is possible to compute a separating hyper-plane with a maximum margin in a feature space.

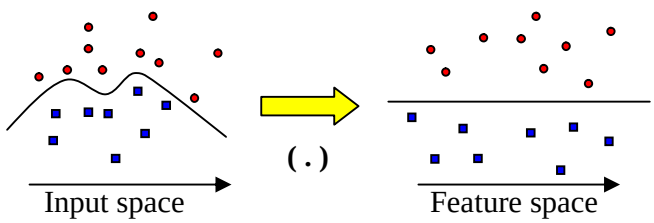

Figure 2. Space translation using a nonlinear kernel function

We need to find, among all hyper-planes separating the data, an existing maximum margin $\frac{2}{\|\omega\|}$ between the classes. The problem is transformed into a quadratic programming problem

$$\min \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l}\xi_i \tag{3}$$
$$\text{s.t } y_i(\omega^T\phi(x_i) + b) = 1 - \xi_i \ , \xi_i \geq 0, i = 1,...l$$

where C is the trade-off parameter between the error and margin.

The quadratic programming problem can be solved by using Lagrangian multipliters $\alpha_i \in \Re$. The solution satisfies the Karush-Kuhn-Tucker (KKT) conditions. The ω can be recovered by using $\omega = \sum_{i=1}^{l}\alpha_i y_i \phi(x_i)$, where x_i are non-zero values and α_i are support vectors (SV).

The decision boundary is determined only by the SV. Let t_j , $(j = 1, ..., s)$ be the indices of the s support vectors. Then we can rewrite

$$\omega = \sum_{j=1}^{s}\alpha_{t_j} y_{t_j} \phi(x_{t_j}) \tag{4}$$

The quadratic programming problem is solved by considering the dual problem

$$\max_\alpha Q(\alpha) = -\frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j K(x_i, x_j)$$
$$s.t. \begin{cases} 0 \leq \alpha_i \leq C, & \forall \text{i} \\ \sum_{i=1}^{l}\alpha_i y_i = 0 \end{cases} \tag{5}$$

with the kernel trick (Mercer Theorem):

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \tag{6}$$

Several types of kernels, such as linear, polynomial, splines, RBF, and MLP, can be used within the SVM. This finally results in the following:

$$y(x) = sign(\sum_{i=1}^{l}\alpha_i y_i K(x, x_i) + b) \tag{7}$$

3. Least squares support vector machines

The standard SVM is solved using quadratic programming methods. However, these methods are often time consuming and are difficult to implement adaptively. Least squares support vector machines (LS-SVM) is capable of solving both classification and regression problems and is receiving more and more attention because it has some properties that are related to the implementation and the computational method. For example, training requires solving a set of linear equations instead of solving the quadratic programming problem involved in the original SVM. The original SVM formulation of Vapnik [9] is

modified by considering equality constraints within a form of ridge regression rather than by considering inequality constraints.

The solution follows from solving a set of linear equations instead of a quadratic programming problem. In LS-SVMs, an equality constraint-based formulation is made within the context of ridge regression as follows:

$$\min \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l} e_i^2 \qquad (8)$$
$$\text{s.t } y_i(\omega^T\phi(x_i)+b) = 1-e_i \ , \ i=1,...l$$

One defines the Lagrangian

$$L(\omega,b,e;\alpha) = \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l} e_i^2 - \sum_{i=1}^{l}\alpha_i\{y_i(\omega^T\phi(x_i)+b)-1+e_i\} \qquad (9)$$

with Lagrangian multipliers $\alpha_i \in \Re$. The conditions for optimality are given by

$$\begin{cases} \dfrac{\partial L}{\partial \omega} = 0 \rightarrow \omega = \sum_{i=1}^{l}\alpha_i y_i \phi(x_i) \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{l}\alpha_i y_i = 0 \\ \dfrac{\partial L}{\partial e_i} = 0 \rightarrow \alpha_i = Ce_i \quad i=1,..l \\ \dfrac{\partial L}{\partial \alpha_i} = 0 \rightarrow y_i(\omega^T\phi(x_i)+b)-1+e_i = 0 \quad i=1,..l \end{cases} \qquad (10)$$

By eliminating e, ω, one obtains the KKT system

$$\begin{bmatrix} 0 & Y^T \\ Y & \Omega + C^{-1}I \end{bmatrix}\begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{1} \end{bmatrix} \qquad (11)$$

where C is a positive constant, b is the bias,

$$y = (y_1...y_l)^T, \vec{1} = (1...1), \alpha = (\alpha_1...\alpha_l) \text{ and}$$
$$\Omega_{ij} = y_i y_j \phi(x_i)^T\phi(x_j), \quad 1 \le i, j \le l = y_i y_j K(x_i, x_j) \qquad (12)$$

After application of the Mercer condition, the final result is the same as Equation (7).

Because LS-SVM does not incorporate the support vector selection method, the resulting network size is usually much larger than the original SVM. To solve this problem, a pruning method can be used to achieve sparseness in LS-SVM [19]. The pruning technique reduces the complexity of the network by eliminating as much hidden neurons as possible.

4. Experimental results

In this section, we will demonstrate the validity of the LS-SVM in a communication system. In the experiment, we used a Pentium4 1.5GHz CPU, 512MB of main memory, and the Matlab 6.1 simulation software.

High speed communication channels are often impaired by channel Inter-symbol Interference (ISI), Additive White Gaussian Noise (AWGN) and effects of time-varying channels. A discrete time model of a digital communication system is shown in Fig.3. A random sequence x_i was passed through a dispersive channel of finite impulse response (FIR) to produce a sequence of outputs $y_i^{'}$. Two kinds of channels, time-invariant and time-varying, were used in our experiment. Additive noise in the system, represented by e_i, was then added to each $y_i^{'}$ to produce an observation sequence y_i.

The problem to be considered was how to utilize the information represented by the observed channel outputs $y_i, y_{i-1}, \ldots, y_{i-m+1}$ to produce an estimate of the input symbol x_{i-d}. A device which performs this function is known as an equalizer. The symbols *m* and *d* are known as the order and the delay of the equalizer, respectively. In our experiment, the equalizer order was chosen to be m=2 and delay of the equalizer was chosen to be d=0. Throughout the experiment, the input samples were chosen from {-1,1} with equal probability and were assumed to be independent of one another.

The equalizer performance is described by the probability of misclassification with respect to the signal-to-noise ratio (SNR). When an independent identically distributed (i.i.d.) sequence is assumed, the SNR can be defined as

$$SNR = 10\log_{10}\frac{\sigma_s^2}{\sigma_e^2} \tag{13}$$

where σ_s^2 represents the signal power and σ_e^2 is the variance of the Gaussian noise.

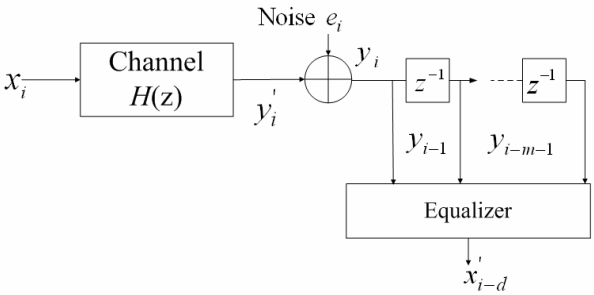

Figure 3. Discrete-time model of a data transmission system

4.1. Bayesian equalizer

The Bayesian decision theory provides the optimal solution to the general decision problem. Therefore, the optimal symbol-by-symbol equalizer can be formed from the Bayesian probability theory and is termed a Bayesian or a maximum posteriori probability (MAP) equalizer.

This minimum error probability decision can be rewritten as

$$f_B(y(n)) = \sum_{i=1}^{n_s^+} p_i(2\pi\sigma_e^2)^{\frac{-m}{2}} \exp\left(-\left\|y(n)-y_i^+\right\|^2 \Big/ 2\sigma_e^2\right) - \sum_{i=1}^{n_s^-} p_j(2\pi\sigma_e^2)^{\frac{-m}{2}} \exp\left(-\left\|y(n)-y_i^-\right\|^2 \Big/ 2\sigma_e^2\right) \tag{14}$$

where σ_e denotes the standard deviation (std) of the Gaussian additive noise *e(k)*. For equiprobable symbols, the coefficients $p_j(2\pi\sigma_e^2)^{\frac{-m}{2}}$ in $f_B(y(n))$ become redundant and can be removed. This gives rise to the following simpler form of the optimal decision function:

$$f_B(y(n)) = \sum_{i=1}^{n_s^+} \exp\left(-\left\|y(n)-y_i^+\right\|^2 \Big/ 2\sigma_e^2\right) - \sum_{i=1}^{n_s^-} \exp\left(-\left\|y(n)-y_i^-\right\|^2 \Big/ 2\sigma_e^2\right) \tag{15}$$

$$x'(n) = \mathrm{sgn}(f_B(y(n))) = \begin{cases} 1, & f_B(y(n) \ge 0 \\ -1, & f_B(y(n) < 0 \end{cases} \tag{16}$$

where y_i^+ and y_i^- refer to the channel states which are the +1 and –1 signal states, which have estimates of the noise-free received signal vector.

Therefore, based on the above function, we can decide on the optimal decision boundary. From this point of view, the equalizer can be viewed as a classifier, and the problem can be considered as a classification problem.

4.2. The time-invariant channel problem

Let the channel transfer function be

$$\begin{aligned} y'(n) &= \alpha(n) + 0.1\times\alpha(n)^3 \\ \alpha(n) &= a1\times x(n) + a2\times x(n-1) \end{aligned} \tag{17}$$

where a1=0.5 and a2=1. Then the output channel signal is

$$\begin{aligned} y'(n) &= (0.5x(n) + x(n-1)) \\ &\quad + 0.1\times(0.5x(n) + x(n-1))^3 \end{aligned} \tag{18}$$

All the combinations of *x(n)* and the desired channel states are listed in Table 1.

To see the actual bit error rate (BER), a realization of 10^6 points of sequence *y(n)* were used to test the BER of the LS-SVM equalizer. The resulting BER curve of the LS-SVM equalizer under different SNRs is shown in Fig. 4. When the level of the noise was 10db, 1000 samples of *y(n)* were used for training, as shown in Figure 5.

We tested the performance of the Bayesian, LS-SVM, multilayer perceptron (MLP), and

wavelet neural network (WNN) equalizers. The Bayesian equalizer was the most optimal method for communication channel equaliza tion. Computer simulation results showed that the bit error rate of the LS-SVM equalizer was close to the optimal equalizer and better than the MLP and WNN equalizers.

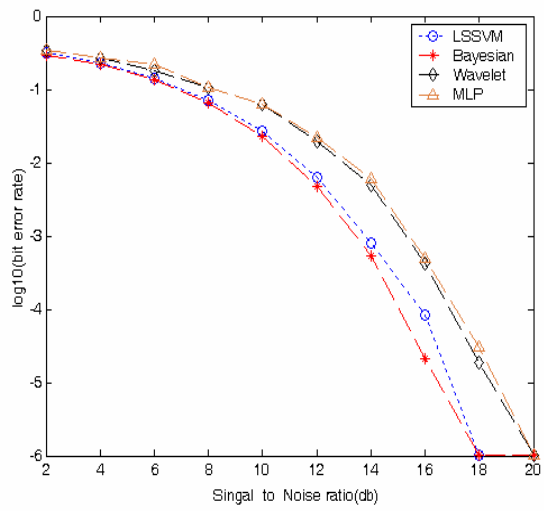

Figure 4. Comparison of bit-error-rate curves for the Bayesian, LS-SVM, MLP, and WNN equalizers in a time-invariant channel

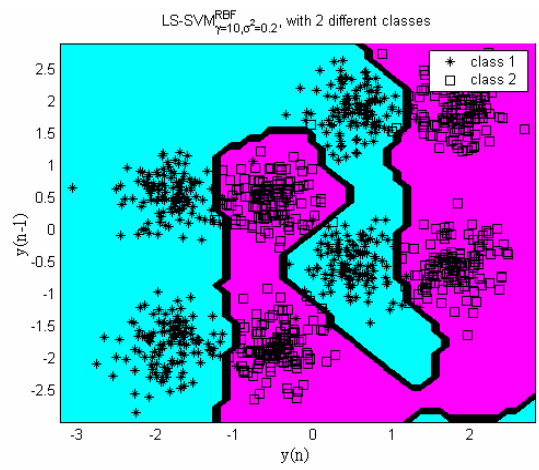

Figure 5. Time-invariant channel, data clusters from LS-SVM, SNR=10db,1000 samples of *y(n)*, and decision boundary

Table 1. Input and desired channel states for m=2 and d=0

NO	x(n)	x(n-1)	x(n-2)	y'(n)	y'(n-1)
1	1	1	1	1.8375	1.8375
2	1	1	-1	1.8375	-0.5125
3	1	-1	1	-0.5125	0.5125
4	1	-1	-1	-0.5125	-1.8375
5	-1	1	1	0.5125	1.8375
6	-1	1	-1	0.5125	-0.5125
7	-1	-1	1	-1.8375	0.5125
8	-1	-1	-1	-1.8375	-1.8375

4.3. The time-varying channel problem

Let the nonlinear time-invariant channel transfer function be as given by Eq. (16) where a_1=0.5 and a_2=1. Since we assume the channel is time-varying, a_1 and a_2 are two time-varying coefficients. These time-varying coefficients were generated by passing white Gaussian noise through a Butterworth low pass filter (LPF). The example was centered at a_1=0.5 and a_2=1, and the input for the Butterworth filter was a white Gaussian sequence with standard deviation (std) β. Applying the function provided by Matlab, we can generate a second-order low pass digital Butterworth filter with cutoff frequency β=0.1. Adjusting the time-varying coefficients, we plotted the coefficients and the corresponding channel states, as shown in Figs. 6(a) and 6(b), respectively. The resulting BER curves of the Bayesian, LS-SVM, MLP, and WNN equalizers under different SNRs are shown in Figure 7. Our method obtained a better BER curve.

According to the experimental results, we see that the classification rate of the LS-SVM equalizer is better than that of some of the

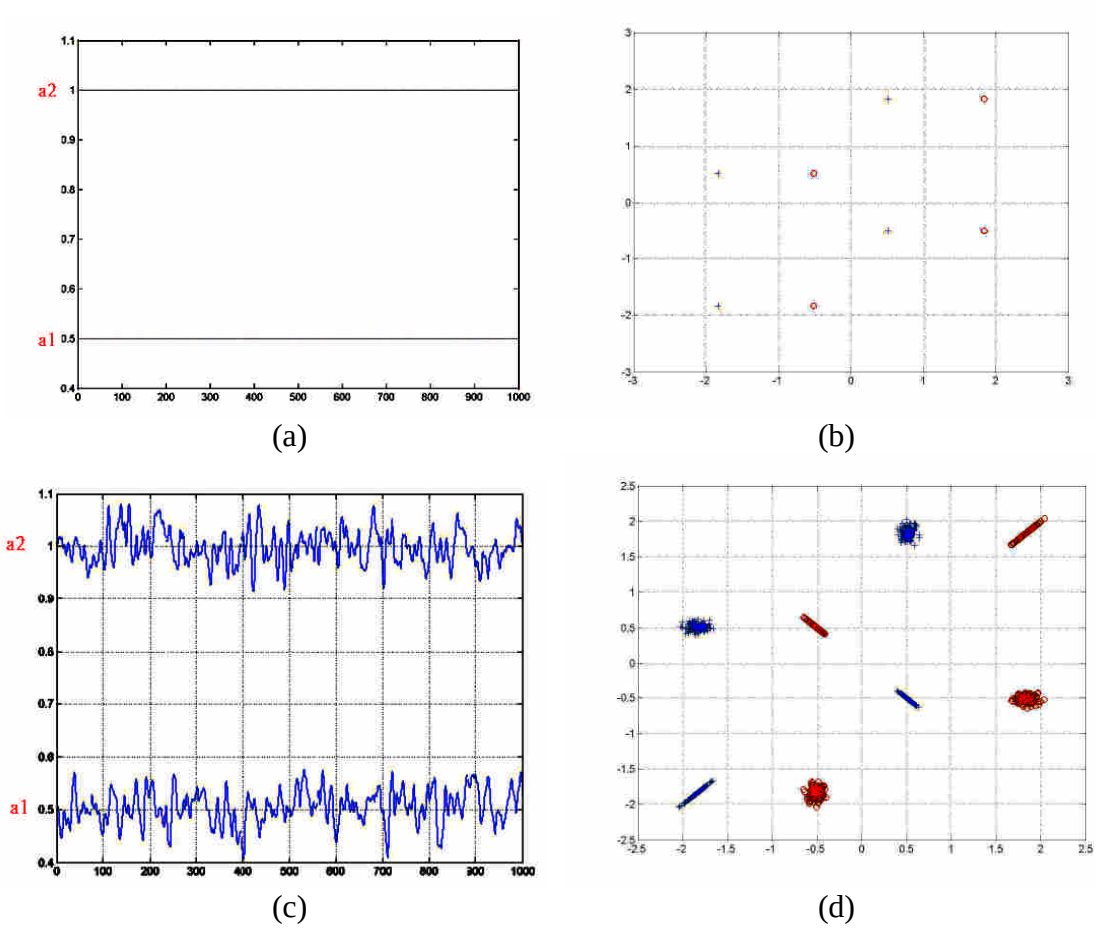

Figure 6. (a) An example of a time-invariant channel; (b) channel states (noise free) of the time-invariant channel; (c) an example of a time-varying channel with β =0.1; and (d) channel states (noise free) of the time-varying channel

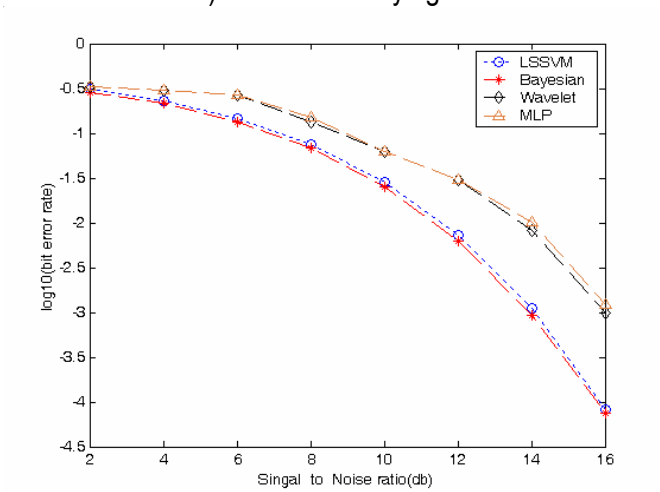

Figure 7: Comparisons of bit-error-rate curves for the Bayesian, LS-SVM, MLP and WNN equalizers in a time-varying channel with β = 0.1

other existing models for channel equalization problems. Table 2 shows the hardware computation time for training and testing when the channel's SNR was 10db. The training time for the Bayesian equalizer was zero because it did not need to be trained. The WNN with the simultaneous Perturbation method [20] was adopted to solve the channel equalization problem and converged after 2000 iterations. The MLP also used the traditional back-propagation algorithm, and its convergence happened after 1000 iterations.

We can see that the training and testing times for the LS-SVM equalizer are faster than those for the WNN and MLP equalizers. For this reason, the LS-SVM equalizer is more suitable for adaptive communication channel equalization.

Table 2. The hardware computation times for various equalizers

Process \ Model	Bayesian (sec)	LS-SVM (sec)	WNN (sec)	MLP (sec)
Training time (Time varying channel)	-	15.36	5045	2011
Testing time (Time varying channel)	511.24	398.45	660.48	543.8
Training time (Time invariant channel)	-	14.97	4583	1997
Testing time (Time invariant channel)	505.99	399.31	602.42	560

5. Conclusion

In this paper, we adopted least squares support vector machines (LS-SVM) for adaptive communication channel equalization. The LS-SVM equalizer has the advantage of being able to be implemented with relatively low complexity and good performance. We applied the LS-SVM equalizer to the time-invariant and time-varying channel problems. Simulation results showed that the bit error rate of the LS-SVM equalizer is close to the Bayesian equalizer and better than the MLP and WNN equalizers.

Reference

[1] Al-Mashouq, K. A. and I. S. Reed. 1994. The use of neural nets to combine equalization with decoding for severe intersymbol interference channels. *IEEE Transactions on Neural Networks*, 5, 6: 982-988.

[2] Chen, S., B. Mulgrew, and P. M. Grant. 1993. A clustering technique for digital communications channel equalization using radial basis function networks. *IEEE Transactions on Neural Networks*, 4, 4: 570-579.

[3] Liang, Q. and J. M. Mendel. 2000. Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters. *IEEE Transactions on Fuzzy Systems*, 8, 5: 551 - 563.

[4] Jagdish, C. Patra and Ranendra N. Pal 1995. A functional link artificial neural network for adaptive channel equalization. *Elsevier Science B. V. Signal Processing*, 43: 181-195.

[5] Kechriotis, G. and E. S. Manolakos. 1994. Using recurrent neural networks for adaptive communication channel equalization. *IEEE Transactions on Neural Networks*, 5, 2: 267-278.

[6] Gibson, G. J. and C. G. N. Cowan. 1990. On the decision regions of multilayer

perceptrons. *Proceedings of the IEEE,* 78: 1590-1594.

[7] Cristea, P., R. Tuduce, and A. Cristea. 2000. Time series prediction with wavelet neural networks. *Proceedings of IEEE Neural Network Applications in Electrical Engineering*.

[8] Zhang, J., G. G. Walter., Y. Miao, and W. N. W. Lee. 1995. Wavelet neural networks for function learning. *IEEE Trans. on Signal Processing,* 43: 1485-1497.

[9] Cortes, C. and V. Vapnik. 1995. Support vector networks. *Machine Learning,* 20, 3: 273-297.

[10] Muller, K.-R., S. Mika., G. Ratsch., K. Tsuda, and B. Scholkopf. 2001. An introduction to kernel-Based learning algorithms. *IEEE Transactions on Neural Networks.* 12, 2: 181-202.

[11] Suykens, J. A. K. and J. Vandewalle, 1999. Least squares support vector machine classifiers. *Neural Processing Letters.* 9, 3: 293-300.

[12] Cristianini, N. and J. Shawe-Taylor. 2000. "*An introduction to support vector machines and other kernel based learning methods"*. Cambrige University Press.

[13] Poggio, T. and F. Girosi. 1990. Networks for approximation and learning. *Proceeding of the IEEE,* 78, 9: 1481-1497.

[14] Vapnik, V. 1995. *"The Nature of Statistical Learning Theory."* Springer-Verlag New York, Inc.

[15] El-Naqa, I., Y. Yang., M. N. Wernick., N. P. Galatsanos, and R. M. Nishikawa. 2002. A support vector machine approach for detection of microcalcifications. *IEEE Transactions on Medical Imaging.* 21, 12: 1552-1563.

[16] Valentini, G., M. Muselli, and F. Ruffino. 2003. Bagged ensembles of support vector machines for gene expression data analysis. *Proceedings of the International Joint Conference on Neural Networks.* 3: 20-24.

[17] Kim, K. I., K. Jung., S. H. Park, and H. J. Kim. 2002. Support vector machines for texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 24, 11:1542-1550.

[18] Malinowski, M., M. Jasin´ski, and M. P. Kazmierkowski. 2004. Simple direct power control of three-phase PWM rectifier using space-vector modulation (DPC-SVM). *IEEE Transactions on Industrial Electronics.* 51, 2:447-454.

[19] Suykens, J. A. K., L. Lukas, and J. Vandewalle. 2000. Sparse least squares support vector machine classifier. *ESANN'2000 European Symposium on Artificial Neural Networks:* 37-42.

[20] Maeda, Y. and R. J. P. De Figueiredo. 1997. Learning Rules for Neuro-Controller via Simultaneous Perturbation. *IEEE Transaction on Neural Networks.* 8, 5:1119-1130.