Support Vector Machine Techniques for Nonlinear Equalization

Daniel J. Sebald, *Member, IEEE,* and James A. Bucklew

***Abstract*—The emerging machine learning technique called support vector machines is proposed as a method for performing nonlinear equalization in communication systems. The support vector machine has the advantage that a smaller number of parameters for the model can be identified in a manner that does not require the extent of prior information or heuristic assumptions that some previous techniques require. Furthermore, the optimization method of a support vector machine is quadratic programming, which is a well-studied and understood mathematical programming technique.**

Support vector machine simulations are carried out on nonlinear problems previously studied by other researchers using neural networks. This allows initial comparison against other techniques to determine the feasibility of using the proposed method for nonlinear detection. Results show that support vector machines perform as well as neural networks on the nonlinear problems investigated.

A method is then proposed to introduce decision feedback processing to support vector machines to address the fact that intersymbol interference (ISI) data generates input vectors having temporal correlation, whereas a standard support vector machine assumes independent input vectors. Presenting the problem from the viewpoint of the pattern space illustrates the utility of a bank of support vector machines. This approach yields a nonlinear processing method that is somewhat different than the nonlinear decision feedback method whereby the linear feedback filter of the decision feedback equalizer is replaced by a Volterra filter. A simulation using a linear system shows that the proposed method performs equally to a conventional decision feedback equalizer for this problem.

***Index Terms*—Decision feedback SVM, ISI, nonlinear equalization, support vector machine.**

I. Introduction

NONLINEAR equalization has many applications [1]–[4] and has remained a challenging analytical problem for several reasons. First, architectures for nonlinear equalization often become unmanageably complex very rapidly; thus, they require novel techniques for limiting their degrees of freedom to make them useful. This has come to be known in the literature as *the curse of dimensionality*—a phrase popularized by Bellman [5]. Second, the nonlinear system requiring equalization is often noninvertible, resulting in a drastic loss of information. Third, algorithms performing nonlinear equalization are often too computationally intensive to be run in real time.

Manuscript received July 19, 1999; revised July 5, 2000. The associate editor coordinating the review of this paper and approving it for publication was Prof. Colin F. N. Cowan.

The authors are with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI 53766 USA (e-mail: sebald@cae.wisc.edu).

Publisher Item Identifier S 1053-587X(00)09291-6.

This paper primarily addresses the first of these issues by investigating the use of support vector machines as a means to perform nonlinear equalization or, more appropriately, nonlinear *detection*. A support vector machine (SVM) uses training data as an integral element of the function estimation model as opposed to simply using training data to estimate parameters of an *a priori* model using maximum likelihood, which is the more traditional approach. The second issue, regarding loss of information, is a consequence of the nonlinear system and is not addressed in this paper. Although this is certainly an issue of interest to a system designer, the main issue here is the detection problem given a predefined nonlinear system. The third issue, regarding computational efficiency, is more general than the equalization problem. This is also certainly an issue of concern as the optimization technique associated with SVMs requires a considerable amount of computation. However, our intent is to investigate feasibility of the SVM method in the detection setting. Efficient SVM implementation is a very active area of research by other investigators, including [6]–[10].

In the case of equalization, it is desirable that a modem require a small set of training data to characterize the transmission channel, hence making better use of bandwidth. In addition, the model should be efficient for real-time applications. We feel that the concepts of generalized learning theory presented in [5], i.e., the idea that an algorithm should attempt to bound the estimation risk based upon empirical risk, is a strong motivation for investigating the use of SVMs for nonlinear equalization.[1] Consequently, SVMs train with relatively small amounts of data, and training is rather straightforward, requiring less *ad hoc* input from the designer. Once training of the SVM has completed, the detection stage is efficient, comparable to Volterra filters and neural networks.

Section II presents the pattern recognition concept of detection and equalization then summarizes solutions using Volterra filters and neural networks. Section III contrasts the SVM against other nonlinear techniques and lays the mathematical groundwork. Section IV gives further details about setting up the equalization problem. Section V then presents simulation results using the model of Section IV. Three nonlinear system models from [12] are analyzed. Results for the first system are illustrated in terms of the pattern space. This gives insight into the decision boundaries that the SVM can construct. The second system introduces colored noise to show that the SVM works on this more general noise problem. The third system investigates the bit error rate (BER) as a function of signal-to-noise ratio (SNR) and compares it against the result

[1]An alternative interpretation of the theory of SVM optimization is a regularization problem [11].

in [12] using a neural network. Section VI presents the SVM bank and state machine idea, and Section VII simulates a linear system studied in [13] with detection done using an SVM with decision feedback and done using an SVM bank. Section VIII summarizes the paper and discusses some open issues.

II. Detection Viewed as Pattern Recognition

As pointed out in [12], equalization may be viewed as a classification problem. In such a scenario, the output of a communications channel can be grouped as a vector and used as the input to a classification machine whose output should match as best as possible some delayed version of the original signal entering the channel. The raw data (i.e., channel output) is transformed to a *pattern* space $\mathcal{P}$ [i.e., it is grouped as a state vector according to the expected amount of intersymbol interference (ISI)]. The pattern space is then transformed to a higher dimensional *feature* space $\mathcal{F}$ (usually of much higher dimension than $\mathcal{P}$, which is sometimes infinite dimensional) that incorporates the nonlinear nature of the model.

We pause to emphasize the difference between true equalization and detection via pattern classification for a digital communications system. Equalization attempts to map the nonlinear channel output, an element of $\mathbb{R}^M$, into $\mathbb{R}$ such that this mapping represents an inverse of the channel as closely as possible according to some accepted functional measure. Clearly, if this can be done effectively, then detection follows in a straightforward manner. On the other hand, detection only attempts to map the channel output into a finite alphabet, say $\{-1,+1\}$, such that this mapping is optimum in a statistical sense. The underappreciated point is that in cases where finding the system inverse only serves as an intermediate step to the goal of detecting digital symbols, the system designer usually finds that the direct detection problem is much easier to solve than the more general equalization problem.

Both Volterra filters [14]–[16] and neural networks [12], [17]–[20] have been studied as techniques for nonlinear equalization. In the case of Volterra filters, a multiplicative structure creates cross-products of all filter states. These cross-products are then weighted linearly, and the problem is to find the optimum weighting that minimizes some cost. The dimension of the model grows quickly, and it becomes necessary to apply some type of heuristic to limit that model. For example, Van Veen *et al.* [21] address the issue of excessive parameterization and propose a clever method to manage the model using diagonal coordinate system approximations. A recent paper by Carini *et al.* [22] proposes a new vector algebra for Volterra filters to address the stability of high dimensional systems.

In the case of neural networks, the feature space is generated via multiple layers of nonlinear functions (i.e., neurons) acting on either filter states (first layer) or outputs of the previous layer. The weightings of the neurons are trained by backpropagation, which is an iterative, gradient descent-like algorithm that is not guaranteed to find a global optimum but may instead tend to a local optimum solution. Researchers have found that the neural network is susceptible to overtraining and that the number of layers, the number of neurons per layer, and when to stop adapting must be determined in an *ad hoc* fashion [12], [20]. That is, the network may find a solution that minimizes the error on the training set very well, but in order to generalize for a test set, some heuristics must be introduced. A recent paper by Uncini *et al.* [4] proposes a complex-valued neural network with a spline activation function as a method for addressing the complexity and generalization problems.

III. SVM

An SVM is a method for separating clouds of data in the feature space $\mathcal{F}$ using an *optimal* hyperplane. By consequence of Cover's theorem [23], a nonlinear mapping from the pattern space $\mathcal{P}$ to the higher dimensional feature space $\mathcal{F}$ is more likely to create linearly separable clouds of data. The SVM nonlinearly maps inner products of the pattern space data, as opposed to the data itself, via a kernel. That is, the pattern space maps to a nonunique generalized surface in the feature space [24]. Thus, the projection does not have the full dimensionality of the feature space, and there are mechanisms for controlling the capacity of the SVM. Furthermore, data points near the optimal hyperplane called *support vectors* are used as a basis for the model. Consequently, an SVM is a nonparametric learning machine for which capacity can be controlled and whose solution is familiar to optimization theory.

A. Dual Optimization Problem

For the classification problem, the training set consists of vectors from the pattern space $\mathbf{x}_i \in \mathbb{R}^M$ and to each vector a classification $y_i \in \{-1,+1\}$, $i = 1,\cdots,L$. This data is completely known beforehand. In the case of the telecommunications problem, it would be the training data periodically sent for purposes of characterizing the transmission channel. For a full derivation of the pattern recognition and optimal hyperplane terminology, see [5], [23], and [24]. In summary, given an input vector $\mathbf{x}$, an SVM classifies according to

$$\hat{y} = \text{sign}\,\{f(\mathbf{x})\}$$

where $\hat{y}$ is the estimate to the classification, and

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i\in S} \alpha_i y_i \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}) + b \\ &= \sum_{i\in S} \alpha_i y_i K(\mathbf{x}_i,\mathbf{x}) + b. \end{aligned} \tag{1}$$

Here, $\{\alpha_i\}$ are Lagrange multipliers, S is a set of indices for which $\mathbf{x}_i$ is a *support vector*, i.e., a vector for which $\alpha_i \neq 0$ after optimization, and $K(\cdot,\cdot)$: $\mathbb{R}^M \times \mathbb{R}^M \mapsto \mathbb{R}$ is a kernel satisfying the conditions of Mercer's theorem [5], [25]. We see in (1) that after training, only a subset of the training data enters the model (i.e., data reduction) and operations are only performed on data in the pattern space and not in the feature space (i.e., more manageable than previously studied nonlinear techniques).

According to Vapnik [5], for training data that is nonseparable, the dual optimization problem is to maximize

$$W(\alpha) = \sum_{i=1}^{L} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{L} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) \tag{2}$$

under constraints

$$\sum_{i=1}^{L} \alpha_i y_i = 0 \tag{2'}$$

$$0 \le \alpha_i \le C, \qquad i = 1, \cdots, L. \tag{2''}$$

This is a quadratic programming (QP) problem that may be solved with traditional optimization techniques [26]. Choice of parameter C is a current research area of others. In Wahba *et al.* [11], it is considered a regularization parameter for which generalized approximate cross validation is utilized. Here, we choose C empirically and find that the solution is not overly sensitive to its value.

B. QP Formulation

The QP algorithm of most mathematics software packages takes the general form [27]

$$\min\{\tfrac{1}{2}\mathbf{z}^T\mathbf{Q}\mathbf{z} + \mathbf{c}^T\mathbf{z}\text{: } \mathbf{a}_i^T\mathbf{z} = b_i, i \in \mathcal{E}, \mathbf{a}_i^T\mathbf{z} \le b_i, i \in \mathcal{I}\} \tag{3}$$

where

- $\mathbf{z}$ variable to be optimized;
- $\mathcal{E}$ index set for equality constraints;
- $\mathcal{I}$ index set for inequality constraints.

Although the $\{\alpha_i\}$s play the role of Lagrange multipliers, here, they are treated as the optimized variable $\mathbf{z}$ in order to implement constraints (2′) and (2″). To put the dual formula (2) into a form suitable for the QP problem, begin by defining the vectors

$$\boldsymbol{\alpha} = [\alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_L]^T$$
$$\mathbf{y} = [y_1 \quad y_2 \quad \cdots \quad y_L]^T$$

and the matrix $\mathbf{Y} = \text{diag}\{\mathbf{y}\}$ and arranging the vectors of the pattern space into an $M \times L$ matrix as

$$\mathbf{X} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_L].$$

The QP problem is usually programmed as a minimization. Expanding the negative of (2) and rearranging terms leads to

$$-W(\alpha) = -\sum_{i=1}^{L} \alpha_i + \frac{1}{2} \sum_{i,j=1}^{L} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$
$$= -\mathbf{1}_{L,1}^T \boldsymbol{\alpha} + \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{Y} K(\mathbf{X}, \mathbf{X}) \mathbf{Y}^T \boldsymbol{\alpha}$$

where $\mathbf{1}_{m,n}$ is an $m \times n$ matrix of ones, and the matrix kernel $K(\mathbf{X}, \mathbf{X})$ is a component-wise application of the kernel, i.e., $K(\mathbf{X}, \mathbf{X})_{m,n} = K(\mathbf{x}_m, \mathbf{x}_n)$. Then, we can easily define the following:

$$\mathbf{z} = \boldsymbol{\alpha}, \quad \mathbf{Q} = \mathbf{Y}\, K(\mathbf{X}, \mathbf{X})\, \mathbf{Y}^T, \quad \mathbf{c} = -\mathbf{1}_{L,1}.$$

Having defined $\mathbf{z}$ as such, the constraints of (2′) and (2″) may be expressed compactly as $\mathbf{A}\mathbf{z} \le \mathbf{b}$ with

$$\mathbf{A} = \begin{bmatrix} \mathbf{y}^T \\ \mathbf{I}_L \\ -\mathbf{I}_L \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0 \\ C\mathbf{1}_{L,1} \\ \mathbf{0}_{L,1} \end{bmatrix}$$

where $\mathbf{0}_{m,n}$ is an $m \times n$ matrix of zeros, and $\mathbf{I}_m$ is a size m identity matrix. It is taken as understood for this matrix representation that the first constraint is an equality constraint, whereas the remaining $2L$ constraints are inequality constraints.

C. Calculating the Affine Offset

The affine constant b in (1) is implicitly determined by the QP solution. The property of support vectors and the Karush-Kuhn-Tucker conditions [24] for the primal optimization problem indicate that for vectors in which the optimized Lagrangian coefficient satisfies $0 < \alpha_j < C$, (1) equals y_j^{-1}, which is equivalent to y_j in this case. This corresponds to support vectors on the *margins* of the optimal hyperplane. Call the set of indices for these support vectors S_{m}. Then, in the case where S_{m} is not empty, b is taken as the average

$$b = \frac{1}{|S_{\text{m}}|} \sum_{\substack{j \in S_{\text{m}} \\ S_{\text{m}} \neq \emptyset}} \left(y_j - \sum_{i \in S} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) \right) \tag{4}$$

where $|S_{\text{m}}|$ is the cardinality of S_{m}. Occasionally, the solution to the QP problem, even when data is separable, only has support vectors for which $\alpha_i = C$. Unfortunately, (4) does not apply in that case, and the affine constant b must come directly from the QP numerical method.

D. Kernel Options

The kernel $K(\cdot, \cdot)$ corresponds to an inner product of vectors in the higher dimensional feature space if and only if Mercer's condition is met. Some common kernels meeting this condition are the polynomial generator

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + 1)^d \tag{5}$$

the Gaussian radial basis function

$$K(\mathbf{x}, \mathbf{z}) = e^{-\|\mathbf{x}-\mathbf{z}\|^2/2\sigma^2}$$

and the sigmoidal neural network function

$$K(\mathbf{x}, \mathbf{z}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{z} - \delta).$$

Preliminary studies on character recognition problems [5] suggest that the type of kernel utilized by the SVM is inconsequential as long as the capacity is appropriate for the amount of training data and complexity of the classification boundary. However, the precise effects of the kernel is still an issue for research [24]. In the simulations for this paper, the polynomial kernel (5) is used exclusively. This kernel is more efficient for real-time applications than the other standard kernels, which require computation of the exponential function. The polynomial order d is a parameter that controls the capacity of the SVM.

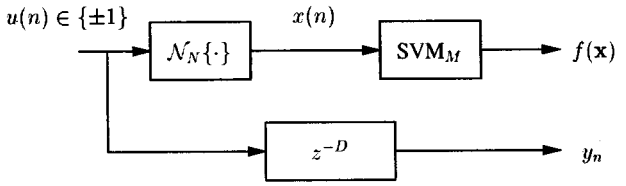


Fig. 1. Nonlinear system $\mathcal{N}$ is equalized by a SVM. The training data for the SVM is obtained from appropriately arranging the nonlinear system output $x(n)$ of the training sequence $u(n)$ and the desired output y_n, which is simply a delayed version of the training sequence.

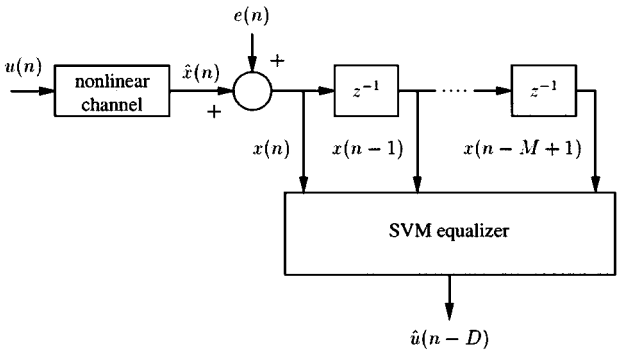


Fig. 2. Model of the nonlinear transmission system originating from Chen *et al.* [12].

The greater d is, the more complex classification boundary the SVM can create.

IV. SVM Model of Equalization

A discrete-time transmission channel can be made to fit the support vector model by grouping the output of the channel into vectors

$$\mathbf{x}(n) = [x(n) \quad x(n-1) \quad \cdots \quad x(n-M+1)]^T$$

and, for training purposes, taking the desired classification to be the input to the channel delayed by D samples, i.e., $y_n = u(n-D)$. This model is illustrated in Fig. 1, where the nonlinear channel $\mathcal{N}_N$ has an ISI of length N. A baseband model of a transmission channel is assumed, but we consider all data to be real valued in our analysis.

To train the SVM, let

$$\mathbf{X} = [\mathbf{x}(n-L+1) \quad \cdots \quad \mathbf{x}(n)]$$

i.e., $\mathbf{x}_i = \mathbf{x}(n-L+i)$, and

$$\mathbf{Y} = \text{diag}\{[u(n-L-D+1) \quad \cdots \quad u(n-D)]\}.$$

The Chen *et al.* model [12] is a pulse amplitude modulation (PAM) scheme illustrated in Fig. 2. It is an example of a cubic simple Wiener nonlinear model [28]. The transmitted data sequence $u(n)$ is an independent, equiprobable binary sequence taking values $\{-1, +1\}$. The output of the channel $x(n) \in \mathbb{R}$ is the sum of a deterministic, nonlinear function of $u(n), \hat{x}(n)$, and an additive noise $e(n)$. The equalizer has a feedforward delay of length $M-1$, i.e., the number of past channel outputs utilized. The goal of the equalizer is then to mimic the desired output $u(n-D)$. Call the equalizer detection output $\hat{u}(n-D)$.

The deterministic portion of the channel model consists of a linear, finite impulse response (FIR) filter followed by a polynomial nonlinearity. Let

$$\tilde{x}(n) = \sum_{k=0}^{N-1} h_k u(n-k) \tag{6}$$

where $\{h_k\}$ are the FIR filter coefficients, and let

$$\hat{x}(n) = \sum_{p=1}^{P} c_p \tilde{x}^p(n)$$

where $\{c_p\}$ are the polynomial coefficients.

Varying the delay D results in different performance of the equalizer because the correlation between $u(n-D)$ and $\mathbf{x}(n)$ changes with D. Since the channel has ISI of length N and the equalizer has feed-forward delay of length $M-1$, the equalizer output is dependent on $M+N-1$ channel inputs, i.e.,

$$\hat{u}(n-D) = f(u(n), u(n-1), \cdots, u(n-M-N+2)). \tag{7}$$

This means that the signal constellation for the detector has 2^{M+N-1} points. The constellation points are the noise-free channel outputs resulting from the various inputs and are classified according to the value of D. Let the noise-free channel outputs be grouped as a vector

$$\hat{\mathbf{x}}(n) = [\hat{x}(n) \quad \hat{x}(n-1) \quad \cdots \quad \hat{x}(n-M+1)]^T.$$

Then, the constellation sets can be expressed as

$$C_{\pm 1,D} = \{\hat{\mathbf{x}}(n) | u(n-D) = \pm 1\}.$$

Similarly, let the classification regions be defined as

$$R_{\pm 1,D} = \{\mathbf{x}(n) | \hat{u}(n-D) = \pm 1\}.$$

Note that the right-hand side of (7) depends on $u(n), \cdots, u(n-M-N+2)$. Under the assumption of independent $\{u(n)\}$, any detector having delay D greater than $D_{\max} = M+N-2$ is certain to yield poor results since $u(n-D)$ is then independent of $u(n), \cdots, u(n-M-N+2)$. The optimum classifier assumed in this study is the Bayesian maximum likelihood detector under conditions of equiprobable *a priori* probabilities and zero/one cost [25]. Let λ represent probability density functions. Then, the Bayesian classifier is

$$\lambda_{\mathbf{x}}(\mathbf{x} | u(n-D) = +1) \underset{H_0}{\overset{H_1}{\gtrless}} \lambda_{\mathbf{x}}(\mathbf{x} | u(n-D) = -1)$$

where hypothesis H_0 declares $\hat{u}(n-D)$ to be -1, hypothesis H_1 declares $\hat{u}(n-D)$ to be $+1$, and

$$\lambda_{\mathbf{x}}(\mathbf{x} | u(n-D) = \pm 1) = \sum_{\{j : \hat{\mathbf{x}}_j \in C_{\pm 1,D}\}} \frac{1}{2|C_{\pm 1,D}|} \lambda_{\mathbf{x}_j}(\mathbf{x})$$

where $\lambda_{\mathbf{x}_j}(\mathbf{x})$ is the probability density associated with random vector $\mathbf{x}_j = \hat{\mathbf{x}}_j + \mathbf{e}$, where $\mathbf{e}$ is an M-length random vector with components distributed similar to $e(n)$.

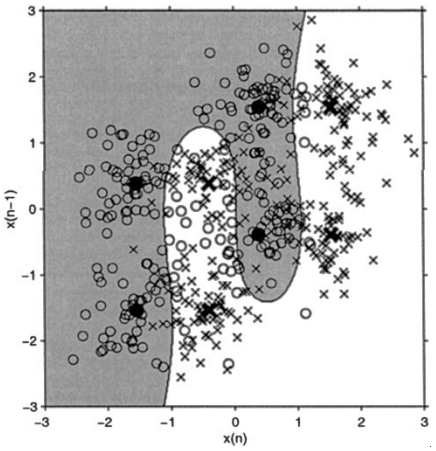


Fig. 3. Example of typical classification regions of an SVM and associated training points with channel $\hat{x}(n) = \bar{x}(n) - 0.9\ \bar{x}^3(n)$, $\bar{x}(n) = u(n) + 0.5u(n-1)$, and Gaussian white noise of power $\sigma_e^2 = 0.2$ and equalizer dimension $M = 2$, polynomial kernel order $d = 3$, constraint $C = 5$, and lag $D = 0$.

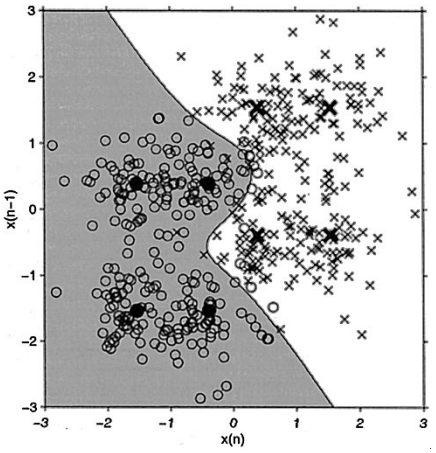


Fig. 4. Example of typical classification regions of an SVM and associated training points with the channel described in Fig. 3, except the equalizer lag is now $D = 1$.

V. Simulations and Results [2]

The first simulation is with the nonlinear channel $\hat{x}(n) = \tilde{x}(n) - 0.9\tilde{x}^3(n)$, $\tilde{x}(n) = u(n) + 0.5u(n-1)$, additive white Gaussian noise of power $\sigma_e^2 = 0.2$, and SVM parameters $M = 2$, $C = 5$ and $d = 3$ with polynomial kernel. Results are an average of ten trials with 500 samples in the training set and 5000 samples in the test set. With this setup, $D_{\max} = 2$. Example SVM classification regions for $D = 0, 1$, and 2 are given in Figs. 3–5, respectively. Region $R_{+1,D}$ is shaded, whereas $R_{-1,D}$ is left unshaded. Included on the pattern space is the signal constellation where $C_{+1,D}$ is marked by a large $\bullet$, and $C_{-1,D}$ is marked by a large $\times$. Training data is also displayed, using a small $\circ$ to indicate $u(n-D) = +1$ and a small $\times$ to indicate $u(n-D) = -1$.

The initial impression is that the SVM classifier constructs a good boundary with respect to the training data it is given. Data clouds of the two types are consistently separated in a logical manner. Chen *et al.* [12] gives optimum boundaries for the

[2]All simulations were done with MATLAB 5.3 with an interface to PATH 3.0 [29] for solving the QP optimization.

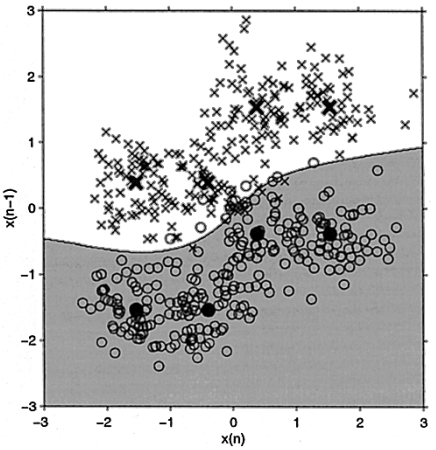


Fig. 5. Example of typical classification regions of an SVM and associated training points with the channel described in Fig. 3, except the equalizer lag is now $D = 2$.

TABLE I
SIMULATION STATISTICS FOR THE SVM $\hat{x}(n) = \bar{x}(n) - 0.9\ \bar{x}^3(n)$, $\bar{x}(n) = u(n) + 0.5\ u(n-1)$, AND GAUSSIAN WHITE NOISE OF POWER $\sigma_e^2 = 0.2$ AND EQUALIZER DIMENSION $M = 2$, POLYNOMIAL KERNEL ORDER $d = 3$, AND CONSTRAINT $C = 5$. THE NUMBER OF TRAINING POINTS IS 500, AND THE NUMBER OF TEST POINTS IS 5000

	$D=0$	$D=1$	$D=2$
$P_{\text{e,train}}$	0.163	0.0458	0.0366
$P_{\text{e,test}}$	0.172	0.0589	0.0421
$P_{\text{e,unproc}}$	0.602	0.905	0.503
$\sigma_{P_{\text{e,train}}}$	0.0166	0.00727	0.00800
$\sigma_{P_{\text{e,test}}}$	0.00722	0.00473	0.00575
$\sigma_{P_{\text{e,unproc}}}$	0.00825	0.00408	0.00506
$\lvert S\rvert$	205	66.5	51.1
$\lvert S_{\text{m}}\rvert$	10.0	9.40	9.30

$D = 0$ and $D = 1$ scenario. In the case of $D = 0$, the SVM classification regions have the general shape of the optimum regions; however, they tend to be more curved in nature. The optimum regions for $D = 0$ are rather complex. The SVM classification regions for $D = 1$ are more near that of the optimum given in [12].

Statistics for the simulations are given in Table I: probabilities of error, P_{e}; associated standard deviations of the probabilities, $\sigma_{P_{\text{e}}}$; total number of support vectors, $|S|$; and number of margin support vectors, $|S_{\text{m}}|$. Surprisingly, delay $D = 2$ produces the best results. One might have guessed a delay of $D = 0$ or 1 would perform better because the impulse response of (6) is only of length 2 for this channel. Most encouraging is that the standard deviations for probability of error for the SVM are approximately one eighth of the probability of error, and the error probabilities for training and test data are approximately the same. This confirms that the SVM is very good at general-

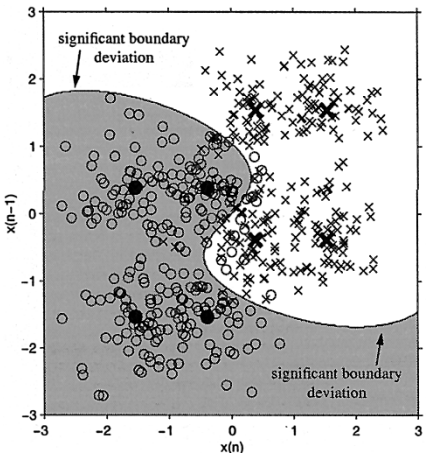


Fig. 6. Example of a type of SVM error in which the classification boundary does not match the optimum in the region where empirical data is unlikely. The system is that described in Fig. 4.

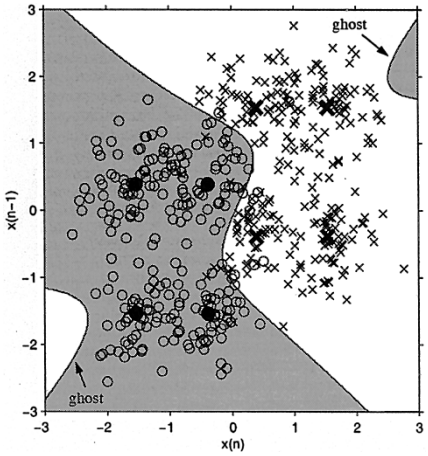


Fig. 7. Example of a type of SVM error in which an obvious region of misclassification appears in the outer regions of the classification space. The system is that described in Fig. 4.

izing. Last, note that there is correlation between the probability of error and number of support vectors. This relationship concerns the severity of the nonlinear boundary for the different pattern spaces as a function of delay D.

Depending on the value chosen for the optimization parameter C, the SVM classification regions $R_{-1,D}$ and $R_{+1,D}$ can exhibit peculiar artifacts. Examples of this are given in Figs. 6 and 7, where $D = 1$. The boundary separating $C_{-1,1}$ from $C_{+1,1}$ in Fig. 6 deviates greatly from the optimum boundary in the portion of the pattern space where data is unlikely to occur. Since the SVM training set does not include data from these locations, it is understandable that the boundary is, in general, arbitrary there. A recent paper by Lyhyaoui *et al.* [30] includes an analysis of boundary behavior for support vector-like classifiers. In Fig. 7, the boundary is constructed in accord with the optimum boundary. However, portions of $R_{-1,1}$ *ghost* into $R_{+1,1}$. Neither of these behaviors have a significant influence on BER. Furthermore, the artifacts can be lessened by decreasing C, where the tradeoff is an increase in the number of support vectors.

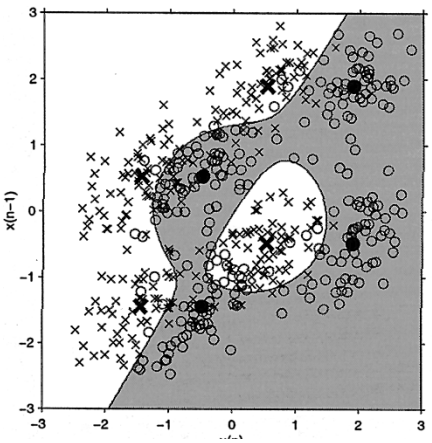


Fig. 8. Example of typical classification regions of an SVM and associated training points with channel $\tilde{x}(n) = \bar{x}(n) + 0.1\,\bar{x}^2(n) + 0.05\bar{x}^3(n)$, $\bar{x}(n) = 0.5\,u(n) + u(n-1)$, and Gaussian colored noise of power $\sigma_e^2 = 0.2$, correlation $\rho = 0.48$ (i.e., $\xi = 0.75$), and equalizer dimension $M = 2$, lag $D = 0$, polynomial kernel order $d = 3$ and constraint $C = 5$.

We now test the behavior of the SVM with colored noise as an input. Chen *et al.* [12] consider zero-mean, stationary correlated noise $e(n)$ with correlation matrix

$$\mathbf{\Sigma} = \begin{bmatrix} [e^2(n)] & E[e(n)e(n-1)] \\ E[e(n-1)e(n)] & E[e^2(n)] \end{bmatrix} = \sigma_e^2 \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}. \tag{8}$$

The method used in simulations for generating colored noise having statistic (8) was an FIR filter

$$e(n) = \frac{\sigma_e}{\sqrt{1+\xi^2}} w(n) + \frac{\sigma_e \xi}{\sqrt{1+\xi^2}} w(n-1)$$

where $w(n) \sim N(0,1)$ and uncorrelated, and

$$\xi = \frac{1}{2\rho} \pm \sqrt{\frac{1}{4\rho^2} - 1}.$$

Fig. 8 shows an example SVM classifier for channel $\tilde{x}(n) = \bar{x}(n) + 0.1\,\bar{x}^2(n) + 0.05\,\bar{x}^3(n)$, $\bar{x}(n) = 0.5u(n) + u(n-1)$, $\sigma_e^2 = 0.2$, $\rho = 0.48$ (i.e., $\xi = 0.75$), $M = 2$, $D = 0$, $d = 3$, and $C = 5$. The number of training points was again 500. The optimum Bayesian solution is given in [12]. Again, the SVM chooses a decision boundary similar to the optimum and is logical in terms of the training data. The optimum $R_{-1,0}$ for this example includes a disconnected region, but the SVM cannot match the polygon nature of the optimum. When approximately $C < 2$, the SVM does not generate disconnected regions but instead consistently creates spoon-like regions.

Last, a third example compares the SVM BER against the optimum BER as a function of SNR for the channel $\tilde{x}(n) = \bar{x}(n) + 0.2\,\bar{x}^2(n)$, $\bar{x}(n) = 0.3482\,u(n) + 0.8704\,u(n-1) + 0.3482\,u(n-2)$, $\rho = 0.48$ (i.e., $\xi = 0.75$), $M = 3$, $D = 1$, $d = 3$, and $C = 0.1$. Let γ be the SNR. Then, $\sigma_e^2 = 1/\gamma$ since the transmitted symbols have unit energy. The number of training samples was again 500, whereas the number of trials and test samples were varied to compensate for greater relative variance for low BER estimates. The variance scaling method of

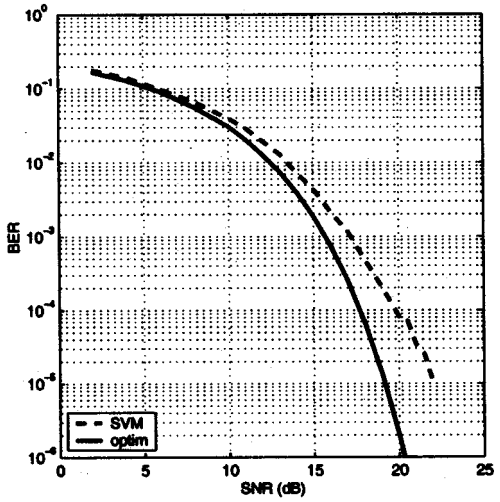


Fig. 9. Average BER for the SVM detector versus the Bayesian detector for the channel $\tilde{x}(n) = \bar{x}(n) + 0.2\,\bar{x}^2(n)$, $\bar{x}(n) = 0.3482\ u(n) + 0.8704\,u(n-1) + 0.3482\ u(n-2)$, and Gaussian colored noise correlation $\rho = 0.48$ (i.e., $\xi = 0.75$) and equalizer dimension $M = 3$, lag $D = 1$, polynomial kernel order $d = 3$, and constraint $C = 0.1$.

importance sampling, first proposed by Hahn and Jeruchim [31] and discussed for simulating communication links in Smith *et al.* [32], was utilized to speed computation of the BER for each individual trial. The number of samples and choice of scaling depend on the SNR. The BER results, which are given in Fig. 9, show that the SVM requires approximately 2.0–2.5 dB more SNR to match the Bayesian performance. Above 15 dB SNR, the SVM result is essentially the same as the neural network solution given in [12].

VI. Combating Temporal Dependence

The results of the previous section suggest that SVMs are a robust, straight-forward manner to do nonlinear processing for the detection problem. The nonlinear boundaries can result in a reduced BER. Yet the nature of the SVM (and, consequently, neural-networks and Volterra filters) does not take into account the temporal relationship among input vectors $\{\mathbf{x}(n)\}$ due to ISI. Rather, an input vector is treated as though it were independent of all other input vectors. Because of the manner in which the input vector is constructed, i.e., basically a state vector of the channel output, this independence assumption clearly is not valid.

It has been shown for linear channels [13] that, depending on the nature of the ISI, significantly better performance can often be achieved by incorporating previously detected symbols into the detector. This concept is exemplified in the decision feedback equalizer (DFE). The concept of the linear DFE, which is shown in Fig. 10, is to pass the previously detected symbols through feedback filter $A(f)$ to create a signal approximating the ISI. Subtracting this signal from the feed-forward filter $B(f)$ output theoretically removes ISI. As for application of the DFE idea in nonlinear scenarios [2], [33], replace the linear feedback and feed-forward filters of Fig. 10 by nonlinear Volterra series channel approximations. Uncini *et al.* [4] point out this method's sensitivity to noise.

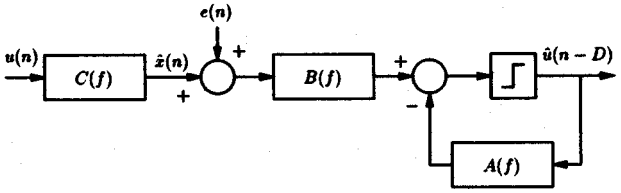


Fig. 10. Decision feedback equalizer in which past decisions are fed back through filter $A(f)$ to cancel ISI.

The decision feedback idea can be incorporated into an SVM by simply lengthening its input vector by appending previous SVM outputs. That is, let

$$\mathbf{x}(n) = [x(n) \quad x(n-1) \quad \cdots \quad x(n-M+1) \quad \hat{u}(n-D-1) \quad \cdots \quad \hat{u}(n-D-M')]^T$$

where M' is the number of previously detected symbols fed back. We call this approach the decision feedback SVM (DFSVM). A DFSVM with correct decisions fed back is called the perfect DFSVM (PDFSVM).

The most thorough study of applying nonlinear equalization is Chen *et al.* [34]. They used a radial basis function (RBF) network with adaptive centers to construct the Bayesian solution having a nonlinear decision boundary. They also described an important property of the pattern space (*observation* space in [34]) related to previously detected symbols and proposed a novel method of utilizing this property in their system. We now reiterate these ideas and show how our system can be modified in a similar way.

Reconsider the constellations of Figs. 3–5. Notice that the locations of points on each constellation (i.e., the eight bigger, filled points) are all the same but are assigned different symbols based on the delay D. The case of $D = 2$ in Fig. 5 results in the best performance because its constellation is such that the minimum distance between points in $C_{+1,2}$ and $C_{-1,2}$ is greater than the distance between points in classes for other delays. Furthermore, its classification boundary is simpler. Therefore, any method that reduces and simplifies the signal constellation will improve BER's and simplify nonlinear models. Utilizing previously detected symbols will do just that.

Let us switch to analyzing a channel model studied by Proakis in [13] for DFE. That channel is linear, i.e., $\hat{x}(n) = \tilde{x}(n)$ with output

$$\tilde{x}(n) = 0.407\ u(n) + 0.815\ u(n-1) + 0.407\ u(n-2) \quad (9)$$

and the signal constellation for delay $D = 1$ is shown in Fig. 11. Because of the fact that opposite symbols in the constellation lie on the same point (i.e., the input $\{+1, -1, +1\}$ is indistinguishable from $\{-1, +1, -1\}$), there is little hope that even a nonlinear boundary could significantly decrease the BER for this example. However, Fig. 12 illustrates how the constellation is reduced when conditioned on previous symbols being known *a priori*. Two properties are immediately clear.

1) For this example, no longer do opposing symbols lie on the same point. In general, distances between classes of symbols will typically increase.

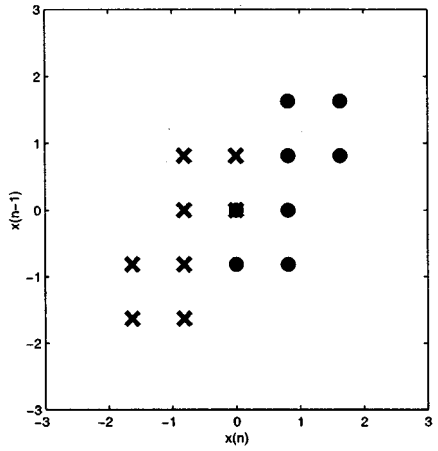


Fig. 11. Constellation diagram for linear channel (9) for delay $D = 1$, assuming $u(n-2)$ and $u(n-3)$ are unknown.

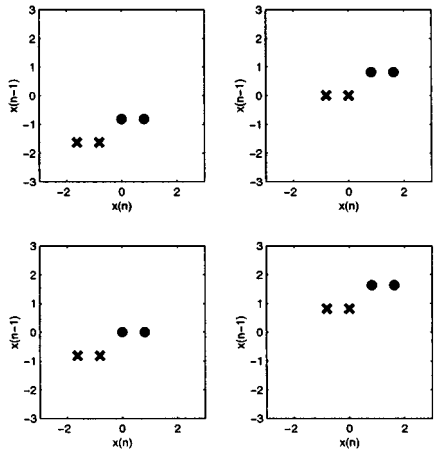


Fig. 12. Various constellations for delay $D = 1$ in different states of $u(n-2)$ and $u(n-3)$ for the linear channel (9).

2) Typically, the severity of the decision boundary decreases, in which case, the SVM becomes simpler and more efficient.

As the delay D increases and the length of the ISI N and dimension of the input M remain constant, fewer symbols will be known (estimated) *a priori* from the perspective of decision feedback. That is, the pattern space simplification is less significant for larger D. The relationship is that the maximum number of useful detected symbols, or *states*, N_s is related to the delay, ISI length, and input dimension as

$$N_s = N + M - D - 2. \tag{10}$$

In (10), $N_s = 0$ when $D = D_{\max}$. Note that in [34], the term *states* is used to refer to the RBF centers, whereas here, the term refers to previously detected symbols.

The logical manner to utilize this property of pattern spaces is to construct a bank of SVMs (or any other of the nonlinear machines discussed in this paper), each of which is trained using an appropriate subset of the training data based on previous symbols being known *a priori*. Then, in the detection stage, a state machine having previously decided symbols as an input selects which SVM to use when deciding the current symbol. The idea is illustrated in Fig. 13. The method may appear computationally intensive, but in fact, the simpler decision boundary for each individual SVM requires less training data per SVM (roughly the same amount overall) and less support vectors per SVM (roughly the same amount overall). This implies that the SVM-bank (SVMB) method is as efficient as the DFSVM approach. In fact, the two methods involve the same principle. However, the SVMB better handles the finite alphabet of the feedback decision, whereas the DFSVM is more appropriate for real or discrete amplitude data.

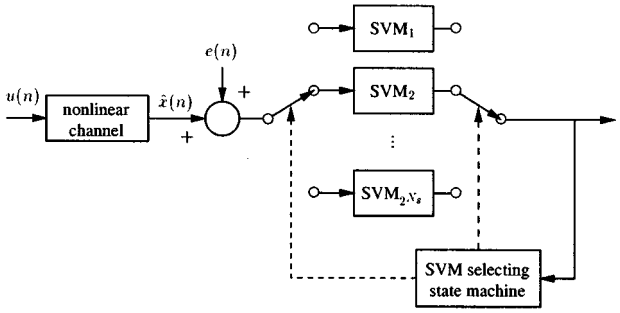


Fig. 13. Bank of SVMs with selection based on the state of previous decisions. Each SVM constructs a different decision boundary based on the constellation for a given state.

In contrast to the RBF approach of [34], the SVMB approach presented here requires no mechanism for finding the centers of RBF kernels and is more general in terms of the properties of noise. That is, although the method in [34] could be adapted for different noise distribution and correlation properties by choice of kernel, either such information would be required *a priori* or some method of density estimation would be required, which is a difficult problem. However, as was shown in the previous section, the general nature of the SVM means that detection performance will often be suboptimal. To be fair, however, in the case of RBF kernels, there is some uncertainty associated with determining centers, variances, and rotations (in the case of correlated noise) that can only degrade performance from the optimal.

Another striking difference between the two methods is that the RBF kernel is adaptive sample-by-sample, whereas an SVMB, as proposed here, can only be block adaptive. In addition, determining the number of RBF kernels is an issue for the system of [34], whereas determining an appropriate number of SVMs in the bank is an issue here. However, we find that having the right number of RBF kernels may be more critical to the system of [34] than the number of states is to the SVMB approach. The more general SVM may require only that the portion of the pattern space most harmful to performance be reduced, whereas the portion of the space less detrimental to performance can be handled by the nonlinear capabilities of the SVM.

VII. More Simulations and Results

To test the performance of the SVMB and DFSVM approaches, a simulation using channel model (9) is presented. The SNR is defined in [13] as

$$\gamma = \frac{1}{N_0} \sum_k |h_k|^2$$

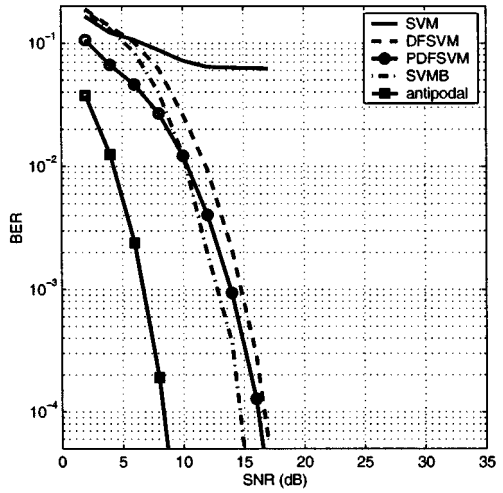


Fig. 14. Average BER for the SVM, SVMB, DFSVM, and PDFSVM detectors compared against the optimum binary, non-ISI detector for the linear channel (9) with added Gaussian noise and equalizer dimension $M = 2$, lag $D = 1$, polynomial kernel order $d = 3$, constraint $C = 0.5$, and DFSVM/PDFSVM feedback dimension $M' = 1$.

where N_0 is the variance for the additive noise of a discrete-time model of equalization, and $h_k \in \mathbb{C}$ is the impulse response of the channel. Thus, the variance of noise $e(n) \in \mathbb{R}$ in the following simulation is $N_o/2$, or more precisely

$$\sigma_e^2 = \frac{\sum_k h_k^2}{2\gamma}$$

where $h_k \in \mathbb{R}$. The optimum non-ISI binary signaling is

$$P_{\mathrm{e,opt}} = \tfrac{1}{2}\ \mathrm{erfc}\ (\sqrt{\gamma})$$

where $\mathrm{erfc}(\cdot)$ is the complimentary error function.

The parameters for the various types of SVM in the simulation were $M = 2$, $D = 1$, $d = 3$, and $C = 0.5$. For the DFSVM and PDFSVM, $M' = 1$. The added noise was white. The number of training samples was 500, and results were averaged over ten trials. Importance sampling was not used in this situation because of problems in large dimensionality due to feedback similar to those described in [32]. Fig. 14 shows the performance as a function of SNR for the various SVM approaches. The SVMB outperforms the DFSVM, requiring approximately 2.0 dB less SNR to achieve the same BER, and it performs about the same as the DFE of [13]. The DFSVM is a simple method with significantly improved performance compared to the SVM.

VIII. Conclusions

Simulations have shown that the SVM provides a robust method for addressing nonlinearities in communication channels exhibiting ISI. The method performs as well as neural networks and Volterra filters and has several advantages over these methods. The use of a bank of SVMs controlled by a state machine—a variation on a technique proposed previously be other researchers—allows incorporation of decision feedback. This significantly increases performance for certain channel scenarios. There are two open issues of concern for applying SVMs to nonlinear equalization: efficient implementation, and adaptive processing.

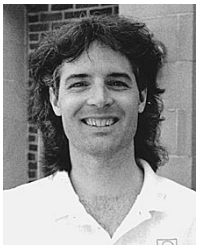
Although QP problems are well studied, the optimization method is somewhat computationally intensive. The SVM-bank approach reduces requirements because of simpler decision boundaries. However, faster optimization techniques are always beneficial. Optimization is an ongoing research problem in the area of mathematical programming, and the particular problem of SVMs is addressed in [8]. Another issue of efficiency is a method for reducing the number of nonmargin support vectors in the case of nonseparable data. This scenario arises when the optimal BER is abnormally high. Whenever a model can be constructed with fewer support vectors, the per-symbol classification becomes more efficient. The run-time complexity problem is studied in [9].

As it stands, the SVM can only be used in applications where a block of training data can be accumulated then the QP optimization performed. There currently is no well-studied sample-by-sample adaptive method for SVMs. An adaptive method would be an important contribution to the field. The goal would be to incorporate the regularization principle of SVMs into an adaptive algorithm. A recent paper by Martínez-Ramón *et al.* [35] proposes a variant of the SVM-like algorithm from [30] for equalization. The method chooses centers of an RBF network from the data using an adaptive clustering algorithm. This is the first attempt, that we are aware of, at a sample-by-sample adaptive methodology incorporating SVM principles.

References

[1] S. D. Personick, "Receiver design for digital fiber optic communication systems, I," *Bell Syst. Tech. J.*, vol. 52, pp. 843–874, July 1973.
[2] E. Biglieri, A. Bersho, R. D. Gitlin, and T. L. Lim, "Adaptive cancellation of nonlinear intersymbol interference for voiceband data transmission," *IEEE J. Select. Areas Commun.*, vol. SAC-2, no. 5, pp. 765–777, Sept. 1984.
[3] G. P. Agrawal, *Nonlinear Fiber Optics*, 2nd ed. San Diego, CA: Academic, 1995.
[4] A. Uncini, L. Vecci, P. Campolucci, and F. Piazza, "Complex-valued neural networks with adaptive spline activation function for digital radio links nonlinear equalization," *IEEE Trans. Signal Processing*, vol. 47, pp. 505–514, Feb. 1999.
[5] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[6] L. Kaufman, "Solving the quadratic programming problem arising in support vector classification," in *Advances in Kernel Methods: Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, ch. 10, pp. 147–167.
[7] T. Joachims, "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods: Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, ch. 11, pp. 169–184.
[8] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Learning*, B. Schökopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, ch. 12, pp. 185–208.
[9] E. E. Osuna and F. Girosi, "Reducing the run-time complexity in support vector machines," in *Advances in Kernel Methods: Support Vector Learning*, B. Schökopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, ch. 16, pp. 271–283.
[10] O. L. Mangasarian and D. R. Musicant, "Data discrimination via nonlinear generalized support vector machines," Compu. Sci. Dept., Univ. Wisconsin, Madison, WI, Tech. Rep. 99-03, Mar. 1999.
[11] G. Wahba, Y. Lin, and H. Zhang, "Generalized approximate cross validation for support vector machines, or, another way to look at margin-like quantities," Dept. Stat., Univ. Wisconsin, Madison, Tech. Rep. 1006, Apr. 1999.

[12] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant, "Adaptive equalization of finite nonlinear channels using multilayer perceptrons," *EURASIP Signal Process.*, vol. 20, no. 2, pp. 107–119, June 1990.
[13] J. G. Proakis, *Digital Communications*, 3rd ed. New York: McGraw-Hill, 1995.
[14] R. D. Nowak and B. D. Van Veen, "Tensor product basis appoximations for Volterra filters," *IEEE Trans. Signal Processing*, vol. 44, pp. 36–50, Jan. 1996.
[15] ——, "Volterra filter equalization: A fixed point approach," *IEEE Trans. Signal Processing*, vol. 45, pp. 377–388, Feb. 1997.
[16] S. Benedetto, E. Biglieri, and V. Castellani, *Digital Transmission Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1987.
[17] R. Parisi, E. D. Di Claudio, G. Orlandi, and B. D. Rao, "Fast adaptive digital equalization by recurrent neural networks," *IEEE Trans. Signal Processing*, vol. 45, pp. 2731–2739, Nov. 1997.
[18] G. Kechriotis, E. Zervas, and E. S. Manolakos, "Using recurrent neural networks for adaptive communication channel equalizaiton," *IEEE Trans. Neural Networks*, vol. 5, pp. 267–278, Mar. 1994.
[19] W. R. Kirkland and D. P. Taylor, "On the application of feedforward neural networks to channel equalization," in *Proc. IJCNN Int. Joint Conf. Neur. Net.*, vol. 2, New York, 1992, pp. II919–II924.
[20] G. J. Gibson, S. Siu, and C. F. N. Cowan, "The application of nonlinear structures to the reconstruction of binary signals," *IEEE Trans. Signal Processing*, vol. 39, pp. 1877–1884, Aug. 1991.
[21] B. D. Van Veen, R. D. Nowak, and G. M. Raz, "Diagonal coordinate system approximations for Volterra filters," *Signal Process.*, to be published.
[22] A. Carini, E. Mumolo, and G. L. Sicuranza, "V-vector algebra and its application to Volterra-adaptive filtering," *IEEE Trans. Circuits Syst. II*, vol. 46, pp. 585–598, May 1999.
[23] L. Devroye, L. Györfi, and G. Lugosi, *A Probablistic Theory of Pattern Recognition*. New York: Springer, 1996.
[24] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Disc.*, vol. 2, no. 2, pp. 1–47, 1998.
[25] H. Stark and J. W. Woods, *Probability, Random Processes, and Estimation Theory for Engineers*. Englewood Cliffs, NJ: Prentice-Hall, 1986.
[26] F. S. Hillier and G. J. Lieberman, *Introduction to Mathematical Programming*. New York: McGraw-Hill, 1995.
[27] J. J. Moré and S. J. Wright, *Optimization Software Guide of Frontiers in Applied Mathematics*. Philadelphia, PA: SIAM, 1993, vol. 14.
[28] R. Haber and L. Keviczky, *Nonlinear System Identification: Input-Output Modeling Approach*. Boston, MA: Kluwer, 1999, vol. 1.
[29] M. C. Ferris and T. S. Munson, "Interfaces to PATH 3.0: Design, implementation and usage," *Comput. Optim. Appl.*, vol. 12, pp. 207–227, 1999.
[30] A. Lyhyaoui, M. Martínez, I. Mora, M. Vázquez, J.-L. Sancho, and A. R. Figueiras-Vidal, "Sample selection via clustering to construct support vector-like classifiers," *IEEE Trans. Neural Networks*, vol. 10, pp. 1474–1481, 1999.
[31] P. Hahn and M. Jeruchim, "Developments in the theory and application of importance sampling," *IEEE Trans. Commun.*, vol. COMM-35, pp. 706–714, July 1987.
[32] P. J. Smith, M. Shafi, and H. Gao, "Quick simulation: A review of importance sampling techniques in communications systems," *IEEE J. Select. Areas Commun.*, vol. 15, pp. 597–613, May 1997.
[33] D. D. Falconer, "Adaptive equalization of channel nonlinearities in QAM data transmission systems," *Bell Syst. Tech. J.*, vol. 57, no. 7, pp. 2589–2611, Sept. 1978.
[34] S. Chen, B. Mulgrew, and S. McLaughlin, "Adaptive Bayesian equalizer with decision feedback," *IEEE Trans. Signal Processing*, vol. 41, pp. 2918–2927, Sept. 1993.
[35] M. Martínez-Ramón, J. L. Sancho-Gómez, C. Bousoño-Calzón, and A. R. Figueiras-Vidal, "Channel equalization via sample selection," in *Proc. Int. COST 254 Workshop Intell. Commun. Technol. Appl. Emphasis Mobile Commun.*, Neuchâtel, Switzerland, May 5–7, 1999.



Daniel J. Sebald (S'89–M'00) received the B.S. degree from the Milwaukee School of Engineering, Milwaukee, WI, in 1987 and the M.S. degree from Marquette University, Milwaukee, in 1992, both in electrical engineering. He is currently pursuing the Ph.D. degree in electrical engineering at the University of Wisconsin, Madison.

He is a registered P.E. in the state of Wisconsin and has worked for Camtronics Medical Systems, Hartland, WI, Nicolet Instrument Technologies, Madison, Xyte, Madison, and OB Scientific, Germantown, WI. His research interests include signal processing, image processing, communications, real-time DSP, and medical technology.

James A. Bucklew received the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1979.

He is currently a professor in the Department of Electrical and Computer Engineering and the Department of Mathematics at the University of Wisconsin–Madison. He is interested in the general area of statistical signal processing and applied probability and has published well over 100 papers in these areas. He is the author of *Large Deviation Techniques in Decision, Simulation, and Estimation*.

Dr. Bucklew served as associate editor for the IEEE TRANSACTIONS ON INFORMATION THEORY and for the IEEE TRANSACTIONS ON SIGNAL PROCESSING.