

# Relevance-Vector-Machine Quantization and Density-Function Estimation: Application to HMM-Based Multi-Aspect Target Classification

Nilanjan Dasgupta, Lawrence Carin and <sup>1</sup>Luise Couchman

Department of Electrical and Computer Engineering

Duke University

Durham, NC 27708-0291

<sup>1</sup>Naval Research Laboratory

Physical Acoustics, Code 7130

Washington, DC 20375-5000

**Abstract** – The relevance vector machine (RVM) is applied for feature-vector quantization (codebook design) and for density-function estimation in high-dimensional feature space. The RVM represents a Bayesian extension of the widely applied support vector machine (SVM). The use of RVMs for quantization and density-function estimation is explored with application to discrete and continuous HMMs, respectively, with comparisons provided to traditional Lloyd codebook design and Gaussian-mixture-model density-function estimation. The RVM-HMM algorithm is employed for multi-aspect target classification, although such is of general utility in HMM applications. Example results are presented for measured multi-aspect acoustic scattering data from several underwater elastic targets.

## I. Introduction

When performing data classification one typically must address the problems of feature-vector quantization [1] and density-function estimation [2]. Vector quantization plays a critical role in such algorithms as self-organizing feature maps (SOFMs) [3], while density-function estimation is important for Bayesian processing [4], as well as in other classifiers.

In this paper we address the problem of quantization and density-function estimation via a new algorithm: the relevance vector machine (RVM) [5]. An RVM represents an extension of the recently developed support vector machine (SVM) [6]. An SVM develops a functional mapping between an input vector  $\mathbf{x}$  and a target output  $t$  (often a scalar). For classification problems, the  $t$  represents the identity of the item to be classified (e.g.  $t = \pm 1$  for the binary-hypothesis problem), while for regression  $t$  is often a continuous function of  $\mathbf{x}$  (e.g. density-function estimation). In training an SVM one is given a set of training-data pairs  $\{\mathbf{x}_n, t_n\}_{n=1, N}$ , and the algorithm learns the function  $f(\mathbf{x}) = \sum_{n=1}^N \alpha_n t_n K(\mathbf{x}, \mathbf{x}_n) + b$  for mapping the testing vector  $\mathbf{x}$  to the target  $t = f(\mathbf{x})$ . In this expression the strength of the Lagrange multiplier  $\alpha_n$  reflects the importance of training example  $\mathbf{x}_n$ ,  $K(\mathbf{x}, \mathbf{x}_n)$  is a kernel that quantifies the similarity between  $\mathbf{x}$  and  $\mathbf{x}_n$  in a prescribed sense, and  $b$  is an offset. For an SVM the expression  $K(\mathbf{x}, \mathbf{x}_n)$  must be a Mercer kernel [6], which implies that  $K(\mathbf{x}, \mathbf{x}_n)$  is characteristic of an inner product between  $\phi(\mathbf{x})$  and  $\phi(\mathbf{x}_n)$  in the space defined by a general mapping  $\phi(\cdot)$ . Most of the Lagrange multipliers are zero, except those along the support of the decision surface [7].

While the SVM is a very attractive classifier, manifesting performance comparable to or better than nearly all classifiers, based on experience with a wide range of data sets [7], it does have some drawbacks. For example, the restriction to Mercer kernels can be limiting, and while the number of support vectors (training examples  $\mathbf{x}_n$  with nonzero  $\alpha_n$ ) is typically a small *percentage* of the training data, the number of support vectors grows linearly with the available training data [8]. The RVM is based on a Bayesian formalism, with which we again learn functional mappings of the form

$f(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$ . However, now  $K(\mathbf{x}, \mathbf{x}_n)$  is viewed simply as a basis function,

and therefore it need not be a Mercer kernel. Moreover, the weights  $w_n$  are typically mostly zero, as in the SVM. However, in the RVM the non-zero weights correspond to the most “relevant” training data, in the sense that they capture the data’s underlying distribution – the “relevant” data does not in general reside along the decision boundary, and therefore for

large data sets the RVM typically yields a sparser representation than the SVM [7] (more weights  $w_n$  are zero than are Lagrange multipliers  $\alpha_n$ ).

For the work reported here, the RVM has a set of particularly important attributes. First, this algorithm seeks to determine the “most relevant” training data for classification, without requiring one to specify *a priori* the number of relevance vectors desired or expected. This is a particularly important attribute for vector quantization (VQ) [1], since most VQ algorithms require one to set the number of codes at the start (although this number can be refined while training) [9]. In addition, the sparse rendering of the function  $f(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$  is of importance for density-function estimation, based on empirical histogram data. Here the target function  $t$  is the histogram, and the function  $f(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$  yields a smooth approximation to the density function. If, for example,  $K(\mathbf{x}, \mathbf{x}_n)$  is representative of a multivariate Gaussian density function centered at  $\mathbf{x}_n$ , then  $f(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$  is analogous to the widely used Gaussian-mixture model (GMM) [10]. The advantage of the RVM is that one need not specify *a priori* the number of Gaussian mixtures, while such is typically required in most expectation-maximization (EM) [11] approximations to the GMM. The RVM therefore offers significant potential in two important areas of classifier design: quantization and density-function design.

We consider these applications of the RVM within a hidden Markov model (HMM), since a discrete HMM requires vector quantization and a continuous HMM requires state-dependent density-function estimation [12]. The performance of an RVM for HMM design is compared with performance accrued via traditional Lloyd encoding [13] (discrete HMM) and via GMM density-function estimation [10] (continuous HMM).

While use of RVMs for discrete and continuous HMMs is of general interest, including for speech processing [10], in this paper we focus on the use of an HMM for

multi-aspect target classification. In this setting the HMM is employed to fuse the information in  $M$  backscattered waveforms, corresponding to viewing an unknown target at  $M$  target-sensor orientations. The use of an HMM for such applications has been addressed in several recent publications [12,14], and therefore only essential details are discussed here. We have previously considered multi-aspect target classification via discrete and continuous HMMs [12]. In this paper we consider the use of the RVM to optimize the design of such classifiers.

The remainder of the paper is organized as follows. In Sec. II we summarize use of an HMM as a multi-aspect target classifier. A brief summary of the feature-extraction techniques is presented in this section. The relevance vector machine (RVM) is discussed in Sec. III, for classification and regression. The application of the RVM to quantization and density-function estimation is discussed in Sec. IV, applicable to discrete and continuous HMMs. A performance analysis of the RVM-based discrete and continuous HMM is discussed in Sec. V, for noise-free and noisy (cluttered) data, with comparison to more-traditional approaches (Lloyd encoding and Gaussian mixture models). Conclusions and further research are discussed in Sec. VI.

## II. Discrete and Continuous HMMs

Assume that we measure a sequence of  $M$  waveforms, each of which is mapped to a feature vector, yielding the sequence of feature vectors  $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M\}$ . Moreover, assume that the  $k$ th HMM,  $T_k$ , is composed of  $N_k$  states. The likelihood of the feature-vector sequence  $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M\}$  given target  $T_k$  is expressed as

$$p(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M | T_k) = \sum_{i=1}^{N_k} \sum_{m=1}^{N_k} \cdots \sum_{l=1}^{N_k} p(\mathbf{v}_1 | s_i, T_k) p(\mathbf{v}_2 | s_m, T_k) \cdots p(\mathbf{v}_M | s_l, T_k) p(s_i, s_m, \dots, s_l | T_k) \quad (1)$$

where  $s_i$  represents the  $i$ th state of target  $T_k$ , and  $p(\mathbf{v}_m | s_i, T_k)$  represents the likelihood of observing feature vector  $\mathbf{v}_m$  in state  $s_i$ . As a consequence of the Markov-model assumption, we have  $p(s_i, s_m, \dots, s_l | T_k) \approx \pi_i^k a_{im}^k \cdots a_{jl}^k$  where, for target  $T_k$ ,  $\pi_i^k$  represents the probability that state  $s_i$  is sampled on the first observation, and  $a_{im}^k$  represents the

probability of transitioning from state  $s_i$  to  $s_m$  on consecutive measurements. Based on this underlying Markov-model assumption, the multiple summations in (1) are computed efficiently via the forward-backward algorithm [15,16]. Alternatively, instead of summing over all possible state sequences, one can determine the most likely state sequence (and associated likelihood) via the Viterbi algorithm [17].

The principal distinction between a discrete and continuous HMM is found in computation of  $p(\mathbf{v}_m | s_i, T_k)$ . In a discrete HMM, each continuous (arbitrary) feature vector  $\mathbf{v}_m$  is mapped to one of a set of discrete (finite) codes. The set of codes define a codebook  $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K\}$ , with each code  $\mathbf{c}_k$  residing in feature space. The codebook may be designed via numerous approaches [1,9], although the Lloyd algorithm [13] is most often applied, with the final set of codes dictated by the available training data. Once the codebook is in place, there are also many forms of quantization one may use [18], although typically the mapping  $\mathbf{v}_m \rightarrow \mathbf{c}_m$  is effected in a nearest-neighbor sense:  $\|\mathbf{v}_m - \mathbf{c}_m\| \leq \|\mathbf{v}_m - \mathbf{c}_k\|, \forall k \neq m$ . After having realized this quantization, the required likelihoods simplify to the state-dependent probability-mass functions (pmfs)  $p(\mathbf{c}_m | s_i, T_k)$ . The principal utility of the discrete HMM *vis-à-vis* its continuous counterpart is found in only requiring the pmfs  $p(\mathbf{c}_m | s_i, T_k)$ , as opposed to the more-general density functions  $p(\mathbf{v}_m | s_i, T_k)$ . This algorithmic simplification is vitiated by the distortion [1] realized in the mapping  $\mathbf{v}_m \rightarrow \mathbf{c}_m$ .

For a continuous HMM, no quantization is applied and one attempts to model the state-dependent densities  $p(\mathbf{v}_m | s_i, T_k)$  directly. This is typically effected via a Gaussian-mixture model

$$p(\mathbf{v} | s_i, T_k) \approx \sum_{j=1}^{J_{ik}} a_j^{i,k} G[\mathbf{v}; \mathbf{m}_j^{i,k}, \mathbf{I}(\sigma_j^{i,k})^2] \quad (2)$$

where  $G[\mathbf{v}; \mathbf{m}_j^{i,k}, \mathbf{I}(\sigma_j^{i,k})^2]$  represents a multivariate Gaussian density function, with mean (in feature space)  $\mathbf{m}_j^{i,k}$  and diagonal covariance  $\mathbf{I}(\sigma_j^{i,k})^2$ . To satisfy the properties of a

density function we require  $\sum_{j=1}^{J_{i,k}} a_j^{i,k} = 1 \forall i,k$ , where  $J_{i,k}$  represents the number of Gaussians used to represent state  $s_i$  of target  $T_k$ .

From the above discussion we note that the principal distinction between the discrete and continuous HMMs is found in the fact that the former performs vector quantization, with state-dependent density functions represented via pmfs, while the latter attempts to model the state-dependent density functions directly, usually in terms of a Gaussian-mixture model (GMM). This therefore introduces two problems of ubiquitous interest in pattern recognition: vector quantization and continuous density-function estimation. We address these issues via an extension of the newly introduced relevance vector machine (RVM) [5,19], and examine RVM performance in the context of HMM-based multi-aspect classification (see the Appendix). As discussed in the subsequent sections, by determining the most “relevant” training data, the RVM autonomously determines the number and position of codes required for quantization of a given data set. Further, these “relevant” training examples may be used to determine the number of Gaussians required in the GMM approximation to a given density function. Consequently, the RVM, as applied here, addresses the problems of determining the required number of codes and mixtures required respectively for discrete and continuous density-function estimation [19].

### III. Relevance Vector Machine

The support vector machine (SVM) [6] has become widely established as one of the leading approaches to pattern recognition and machine learning. It expresses predictions in terms of a linear combination of kernel functions [8] centered on a subset of the training data, known as support vectors. The SVM makes “hard” classifications, rather than generating the *likelihood* that given data is associated with a particular class. Recently, Tipping [5] has formulated a probabilistic model whose functional form is equivalent to the SVM. It achieves comparable recognition accuracy to the SVM while making probabilistic predictions [5,19], with the facility to utilize arbitrary basis

functions (e.g. non-Mercer kernels [7]). Moreover, while the SVM defines support vectors along the decision boundary, the RVM selects “relevant” training examples that capture the underlying statistical distribution of the training data. The relevant vectors are not necessarily along the decision boundary, and it has been found that the number of RVM relevant vectors is often much smaller than the required number of SVM support vectors, for representative data sets [5].

Given a set of training examples  $\{\mathbf{x}_n, t_n\}_{n=1, N}$  ( $t_n$  is real-valued for regression and represents labels for classification, e.g.  $t = \pm 1$  in the binary case), we wish to learn a model of the dependency of the targets  $t_n$  on the inputs  $\mathbf{x}_n$ , with the objective of making accurate predictions of  $t$  for previously unseen values of  $\mathbf{x}$ . The prediction for  $t(\mathbf{x})$  is of the form

$$t(\mathbf{x}; \mathbf{w}) = w_0 + \sum_{i=1}^N w_i K(\mathbf{x}, \mathbf{x}_i) \quad (3)$$

where the output is a linearly-weighted sum of  $N$  generally nonlinear *kernel* functions. Hence, training involves estimation of appropriate values for the parameters (or weights)  $\mathbf{w} = (w_0, w_1, w_2, \dots, w_N)$ .

The RVM represents a Bayesian framework for obtaining good generalization performance to classification and regression tasks while the inferred predictors are often very sparse (most weights are zero). While the SVM tries to minimize the training error of the data under the constraint of maximum smoothness (for regression), an RVM uses a fully probabilistic framework and introduces priors on the model weights, governed by a set of hyperparameters, one associated with each weight. Sparsity is achieved since the posterior distributions of many of the weights are sharply (indeed infinitely) peaked around zero. The training vectors associated with non-zero weights are called ‘relevance’ vectors, motivated by the principle of *automatic relevance determination* [20]. Relevance-vector classifiers have demonstrated comparable performance to SVMs, often using significantly fewer kernel functions. This leads to reduced model complexity and often to better generalization.

## A. Sparse Bayesian learning for regression

Given a data set of input-target pairs  $\{\mathbf{x}_n, t_n\}_{n=1,N}$  and assuming that the targets are samples from a model, with additive noise,

$$t_n = y(\mathbf{x}_n; \mathbf{w}) + \varepsilon_n \quad (4)$$

where the parameters  $\varepsilon_n$  are independent samples from a noise process, assumed for algorithmic simplicity to be iid zero-mean Gaussian with variance  $\sigma^2$ . Thus  $p(t_n | y(\mathbf{x}_n), \sigma^2) = N(t_n | y(\mathbf{x}_n), \sigma^2)$ , where the notation specifies a Gaussian distribution over  $t_n$  with mean  $y(\mathbf{x}_n)$  and variance  $\sigma^2$ . The likelihood of the data set can be written as

$$p(\mathbf{t} | \mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-N/2} \exp\left(-\frac{\|\mathbf{t} - \Phi\mathbf{w}\|^2}{2\sigma^2}\right) \quad (5)$$

where  $\mathbf{t} = (t_1 \dots t_N)$ ,  $\mathbf{w} = (w_0 \dots w_N)$  and  $\Phi$  is the  $N \times (N+1)$  ‘design’ matrix with  $\Phi = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \dots, \phi(\mathbf{x}_N)]^T$ , wherein  $\phi(\mathbf{x}_n) = [1, K(\mathbf{x}_n, \mathbf{x}_1), K(\mathbf{x}_n, \mathbf{x}_2), \dots, K(\mathbf{x}_n, \mathbf{x}_N)]^T$ ,  $K$  being the kernel. A preference for a sparse representation is encoded in the RVM by making a zero-mean Gaussian prior distribution over  $\mathbf{w}$ ,

$$p(\mathbf{w} | \boldsymbol{\alpha}) = \prod_{i=0}^N N(w_i | 0, \alpha_i^{-1}), \quad (6)$$

with  $N+1$  hyperparameters  $\boldsymbol{\alpha} = \{\alpha_0, \dots, \alpha_N\}$ . Although one may choose pattern-specific priors on the  $\alpha$ ’s, we have used uniform hyperpriors [5] in our research. Once the priors are defined, the posterior over all unknowns given the data may be computed using Bayes’ rule as follows

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{t}) = \frac{p(\mathbf{t} | \mathbf{w}, \boldsymbol{\alpha}, \sigma^2) \cdot p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)}{p(\mathbf{t})} \quad (7)$$

Given a new test vector,  $\mathbf{x}^*$ , predictions are made for the corresponding target  $t^*$  in terms of the predictive distribution as

$$p(t^* | \mathbf{t}) = \int p(t^* | \mathbf{w}, \boldsymbol{\alpha}, \sigma^2) p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{t}) d\mathbf{w} d\boldsymbol{\alpha} d\sigma^2. \quad (8)$$

Since the posterior  $p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{t})$  cannot be computed directly, it is decomposed as,

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 | \mathbf{t}) = p(\mathbf{w} | \mathbf{t}, \boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}, \sigma^2 | \mathbf{t}) \quad \text{with} \quad p(\mathbf{t} | \mathbf{w}, \boldsymbol{\alpha}, \sigma^2) \equiv p(\mathbf{t} | \mathbf{w}, \sigma^2) \quad (9)$$

and the posterior distribution over the weights can be computed analytically since its normalizing integral,  $p(\mathbf{t} | \boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{t} | \mathbf{w}, \sigma^2) p(\mathbf{w} | \boldsymbol{\alpha}) d\mathbf{w}$  is a convolution of two Gaussian distributions. The posterior distribution over the weights is thus given by

$$\begin{aligned} p(\mathbf{w} | \mathbf{t}, \boldsymbol{\alpha}, \sigma^2) &= \frac{p(\mathbf{t} | \mathbf{w}, \sigma^2) \cdot p(\mathbf{w} | \boldsymbol{\alpha})}{p(\mathbf{t} | \boldsymbol{\alpha}, \sigma^2)} \\ &= (2\pi)^{-(N+1)/2} |\boldsymbol{\Sigma}|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{w} - \boldsymbol{\mu})\right\} \end{aligned} \quad (10)$$

where the posterior covariance and mean are  $\boldsymbol{\Sigma}$  and  $\boldsymbol{\mu}$  respectively with

$$\begin{aligned} \boldsymbol{\Sigma} &= (\sigma^2 \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \mathbf{A})^{-1} \\ \boldsymbol{\mu} &= \sigma^2 \boldsymbol{\Sigma} \boldsymbol{\Phi}^T \mathbf{t} \end{aligned} \quad (11)$$

with  $\mathbf{A} = \text{diag}(\alpha_0, \alpha_1, \dots, \alpha_N)$ .

The hyperparameter posterior  $p(\boldsymbol{\alpha}, \sigma^2 | \mathbf{t})$  can be approximated by a delta-function at its most-probable values  $\alpha_{MP}$ . Thus, relevance vector ‘learning’ is the search for the hyperparameter posterior mode, *i.e.* the maximization of  $p(\boldsymbol{\alpha}, \sigma^2 | \mathbf{t}) \propto p(\mathbf{t} | \boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}) p(\sigma^2)$  with respect to  $\boldsymbol{\alpha}$  and  $\sigma^2$ . For the case of uniform hyperparameters, maximization of the term  $p(\mathbf{t} | \boldsymbol{\alpha}, \sigma^2)$  is given by,

$$\begin{aligned} p(\mathbf{t} | \boldsymbol{\alpha}, \sigma^2) &= \int p(\mathbf{t} | \mathbf{w}, \sigma^2) p(\mathbf{w} | \boldsymbol{\alpha}) d\mathbf{w}, \\ &= (2\pi)^{-N/2} |\sigma^2 \mathbf{I} + \boldsymbol{\Phi} \mathbf{A}^{-1} \boldsymbol{\Phi}^T|^{-1/2} \exp\left\{-\frac{1}{2} \mathbf{t}^T (\sigma^2 \mathbf{I} + \boldsymbol{\Phi} \mathbf{A}^{-1} \boldsymbol{\Phi}^T)^{-1} \mathbf{t}\right\} \end{aligned} \quad (12)$$

For  $\boldsymbol{\alpha}$ , differentiation of (12), equating to zero and rearranging, following the approach of [20], gives

$$\alpha_i^{new} = \frac{\gamma_i}{\mu_i^2} \quad \text{and} \quad \gamma_i \equiv 1 - \alpha_i \Sigma_{ii} \quad (13)$$

where  $\mu_i$  being the  $i^{\text{th}}$  posterior mean weight and  $\Sigma_{ii}$  is the  $i^{\text{th}}$  diagonal element of the posterior weight covariance computed with the current  $\boldsymbol{\alpha}$  and  $\sigma^2$  values. For the noise variance  $\sigma^2$ , differentiation leads to the re-estimate,

$$(\sigma^2)^{\text{new}} = \frac{\|\mathbf{t} - \Phi \mathbf{w}\|^2}{N - \sum_i \gamma_i} \quad (14)$$

where  $N$  refers to the number of training data examples. Re-estimated  $\boldsymbol{\alpha}, \sigma^2$  are used in the calculation of  $\Sigma$  and  $\boldsymbol{\mu}$ . During iterative re-estimation, many of the  $\alpha_i$  tend to infinity. Hence, the associated  $p(w_i | \mathbf{t}, \boldsymbol{\alpha}, \sigma^2)$  becomes highly peaked at zero. The corresponding basis functions are thus pruned, and sparsity is realized.

At convergence of the hyperparameter estimation procedure, the predictive distribution is represented as

$$p(t^* | \mathbf{t}, \boldsymbol{\alpha}_{MP}, \sigma_{MP}^2) = \int p(t^* | \mathbf{w}, \sigma_{MP}^2) p(\mathbf{w} | \mathbf{t}, \boldsymbol{\alpha}_{MP}, \sigma_{MP}^2) d\mathbf{w} \quad (15)$$

Since both terms in the integrand are Gaussian, this is readily computed, giving

$$p(t^* | \mathbf{t}, \boldsymbol{\alpha}_{MP}, \sigma_{MP}^2) = N(t^* | \mathbf{y}^*, \mathbf{I}\sigma_*^2), \quad (16)$$

where  $\mathbf{y}^* = \boldsymbol{\mu}^T \Phi(\mathbf{x}^*)$  and  $\sigma_*^2 = \sigma_{MP}^2 + \Phi(\mathbf{x}^*)^T \Sigma \Phi(\mathbf{x}^*)$ . The predictive mean is intuitively  $\mathbf{y}(\mathbf{x}^*; \boldsymbol{\mu})$ , or the basis functions weighted by the posterior mean weights, many of which will typically be zero.

## B. RVM-based classification

Relevance vector classification pursues an essentially identical Bayesian framework as regression with the difference being that the labels ( $t$ 's) are now discrete rather than real values. The objective is to predict the posterior probability  $p(x|i)$  for each class  $i$ , given the input  $\mathbf{x}$ . Since the estimated output  $y$  could assume any real value  $(-\infty, +\infty)$ , we define the posterior probability of an observation  $\mathbf{x}$  by applying the logistic sigmoidal link function  $\rho(y) = 1/(1 + e^{-y})$ , with  $y(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$ . Adopting the Bernoulli distribution for  $p(t|\mathbf{x})$ , the likelihood of an observed label  $\mathbf{t} = \{t_1, \dots, t_N\}$  is expressed as

$$p(\mathbf{t} | \mathbf{w}) = \prod_{n=1}^N \rho\{y(\mathbf{x}_n; \mathbf{w})\}^{t_n} [1 - \rho\{y(\mathbf{x}_n; \mathbf{w})\}]^{1-t_n} \quad (17)$$

where  $t_n \in \{0, 1\}$ . Note that, by this construct, the algorithm seeks to make  $y(\mathbf{x})$  large when

$\mathbf{x}$  is associated with class  $t=1$ , while making it small when associated with  $t=0$  (we employ a different definition of the binary-classification target function *vis-à-vis* the  $t = \pm 1$  used in the SVM [7]). The probability that  $y(\mathbf{x})$  is associated with class  $t=1$  is quantified as  $\rho(y(\mathbf{x}))$ , with the probability of being associated with class  $t=0$  expressed as  $1 - \rho(y(\mathbf{x}))$ . Since the marginal likelihood cannot be obtained in a closed form, unlike the regression case, an approximate method is chosen [20] which is based on the Laplace’s method. For a current, fixed value of  $\boldsymbol{\alpha}$ , the covariance of the posterior weights,  $\Sigma$ , is calculated as  $\Sigma = (\boldsymbol{\Phi}^T \mathbf{B} \boldsymbol{\Phi} + \mathbf{A})^{-1}$  where  $\mathbf{A} = \{\alpha_0, \alpha_1, \dots, \alpha_N\}$ . The ‘most probable’ weights  $\mathbf{w}_{MP}$  are defined as  $\mathbf{w}_{MP} = \Sigma \boldsymbol{\Phi}^T \mathbf{B} \mathbf{t}$ . Since the classification case is treated as “noise free” case,  $\mathbf{w}_{MP}$  is directly used in place of  $\boldsymbol{\mu}$  (from the regression case). The equivalent iterative optimization of hyperparameters follows as in regression.

#### IV. RVM Quantization and Density-Function Estimation

The RVM algorithm summarized in Sec. III has been applied for classification problems as well as for regression of one-dimensional signals [5]. In the subsequent section we demonstrate how the “relevance” vectors extracted via RVM may be applied as codes for quantization, and we demonstrate how RVM may be applied for density-function estimation in high-dimensional feature space.

##### A. Relevance-vector quantization

The Lloyd (also termed K-means) algorithm [13] is a well-established technique for vector quantization. Despite being a fast and efficient approach for codebook design, it may converge to local minima. Consequently, codebook initialization plays a crucial role in algorithm performance. In addition, one must set the codebook size in advance, before optimizing the design of the codes.

To address these issues, we have augmented the RVM-based classification technique to quantize the feature space. We assign a *single* class label (e.g. ‘+1’) to all feature vectors to be quantized. Specifically, for  $N$  training examples we have the labeled

data  $\{\mathbf{x}_n, 1\}_{n=1, N}$ . The quantization step seeks to discretely model the distribution of the training data in feature space, and here we seek to push  $\rho(y(\mathbf{x}_m)) \rightarrow 1$  for all  $\mathbf{x}_m \in \{\mathbf{x}_n\}_{n=1, N}$ ; this corresponds to making  $y(\mathbf{x}_m)$  as large as possible for all  $\mathbf{x}_m \in \{\mathbf{x}_n\}_{n=1, N}$ . The trained RVM model corresponds to a sparse set of relevance vectors, those with non-zero weights  $w_n$  in the representation  $y(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$ . The extracted relevance vectors are here treated as codes. It has been demonstrated that the relevance vectors capture the distribution of the underlying training data, without the need to set the number of codes *a priori*.

It is of interest to examine how this codebook design procedure is related to the Lloyd algorithm, which seeks to find a set of codes that minimize the average distortion between the training data and the codes. The RVM algorithm designs a set of weights  $w_n$

that make the function  $y(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n) + w_0$  as large as possible for  $\mathbf{x}$  a member of

the training set, *i.e.* we seek to maximize  $y(\mathbf{x}_m) = \sum_{n=1}^N w_n K(\mathbf{x}_m, \mathbf{x}_n)$  for all  $\mathbf{x}_m \in \{\mathbf{x}_n\}_{n=1, N}$ ,

where we set  $w_0=0$  for reasons discussed below. The kernel  $K(\mathbf{x}_m, \mathbf{x}_n)$  can be viewed as an inverse-distance measure, between training examples  $\mathbf{x}_m$  and  $\mathbf{x}_n$ , taking on its largest value when  $\mathbf{x}_m=\mathbf{x}_n$ . The RVM has two competing goals: maximize  $y(\mathbf{x}_m)$  for all  $\mathbf{x}_m \in \{\mathbf{x}_n\}_{n=1, N}$ , with this constrained by the hyperparameters [18] which reward a sparse representation. The constraint is important, since otherwise all  $w_n$  will be large, to maximize  $y(\mathbf{x}_m)$ . The algorithm implicitly defines a set of representative training data  $\mathbf{x}_n$ , here used as codes, that are relatively close to a neighborhood (cluster) of other training examples, with closeness quantified via the kernel. The output of this process is consistent with the motivations of the Lloyd algorithm, but here one need not set the number of codes *a priori* – the RVM determines the number of codes (relevance vectors) by balancing the two competing metrics in the constrained optimization problem. Moreover, initially the hyperparameters are set to give each training example equal

likelihood of being a “relevant” vector, and therefore we do not have to initialize the codes).

The kernel  $K(\mathbf{x}_m, \mathbf{x}_n)$  permits a general, nonlinear means of defining “closeness” in feature space. If one chooses, for example, a kernel based on the widely applied radial basis functions [21],  $K(\mathbf{x}_m, \mathbf{x}_n) = \exp(-\|\mathbf{x}_m - \mathbf{x}_n\|^2 / \eta^2)$ , then the distance measure is directly analogous to the Euclidean distortion typically employed in Lloyd codebook design. However, other kernels can be employed to generalize the definition of closeness in feature space.

As alluded to above, one necessary modification to the RVM classification scheme deals with the offset  $w_0$ . In a binary classification problem,  $w_0$  behaves as a scaling (bias) of the threshold between the two hypotheses. The use of RVM for quantization involves data solely from a *single* class. Therefore, we have eliminated  $w_0$  from the design matrix while training the RVM model. The new design matrix  $\Phi$  is of size  $N \times N$  and represented as  $\Phi = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \dots, \phi(\mathbf{x}_N)]^T$ , where  $\phi(\mathbf{x}_n) = [K(\mathbf{x}_n, \mathbf{x}_1), K(\mathbf{x}_n, \mathbf{x}_2), \dots, K(\mathbf{x}_n, \mathbf{x}_N)]^T$ .

The constrained-optimization discussed above favors a sparse representation. The balance between sparseness and maximizing  $y(\mathbf{x}_m) = \sum_{n=1}^N w_n K(\mathbf{x}_m, \mathbf{x}_n)$  for all  $\mathbf{x}_m \in \{\mathbf{x}_n\}_{n=1, N}$  is not made explicit in the form, for example, of a Lagrangian (as in an SVM [7]), rather sparseness is invoked via the hyperparameters [5]. We have found it useful to make the following augmentation of the hyperparameters, to assure that the codebook is not overly sparse. The importance of each relevance vector (or code) is defined in terms of its distance from  $L$  of its nearest neighbors (determined on each RVM iteration, and quantified via the kernel). The modification of the hyperparameters  $\alpha$  is given by

$$\alpha_i^* = \frac{\alpha_i}{P_i} \quad \forall i = 1, \dots, N \quad (18a)$$

$$P_m = \frac{\lambda_m}{\sum_{m=1}^M \lambda_m} \quad \text{and} \quad \lambda_m = \sum_{\mathbf{x}_n \in S_m} \frac{1}{K(\mathbf{x}_m, \mathbf{x}_n)} \quad (18b)$$

where  $S_m$  defines a set of the  $L$  closest training examples to  $\mathbf{x}_m$ , as defined by the kernel. The further apart the members of  $S_m$ , the larger the parameter  $\lambda_m$ , implying that the hyper-parameter  $\alpha_i^*$  is made smaller than it otherwise would be. From (6) we note that this reduction of  $\alpha_i^*$  increases the likelihood that training data in less-densely populated regions of feature space may be employed as relevant vectors (codes). In the results presented in Sec. V, we have set  $L=3$ .

## B. Novelty detection

Assume again that we are given a set of labeled data  $\{\mathbf{x}_n, 1\}_{n=1, N}$  characteristic of a particular (single) phenomena labeled  $S$ , and our goal is to determine the likelihood that a new vector  $\mathbf{x}$  is associated with  $S$ , defined as  $p(\mathbf{x}|S)$ . For example,  $S$  may be characteristic of a particular HMM state. As we discuss in the next section, a rigorous means of estimating  $p(\mathbf{x}|S)$  is to generate a histogram based on the available training data  $\{\mathbf{x}_n, 1\}_{n=1, N}$ , from which regression is used to form a smooth estimate of  $p(\mathbf{x}|S)$ . We first suggest a simpler alternative, representative of a simple extension of the RVM quantization algorithm discussed in the previous section.

Assume that RVM quantization is effected as discussed in the previous section, from which we yield the representation  $y(\mathbf{x}) = \sum_{n=1}^N w_n K(\mathbf{x}, \mathbf{x}_n)$ . In Sec. IVA the RVM was employed to develop a codebook  $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K\}$ , and the probability of any new vector  $\mathbf{x}$  is effected in a two-step process: we first quantize  $\mathbf{x}$  in terms of one of the codes in  $\mathbf{C}$ , and then a pmf is used to quantify the probability of observing the associated code (and, implicitly, the original vector  $\mathbf{x}$ ). Alternatively, we can quantify the probability of  $\mathbf{x}$  as  $p(\mathbf{x}|S) \approx \rho(y(\mathbf{x}))$ , recalling that  $\rho(y) = 1/(1 + e^{-y})$  is the sigmoidal function defined previously. By construction, using this approach  $p(\mathbf{x}|S) \approx 1$  for  $\mathbf{x} \in S$ , and ideally

$p(\mathbf{x}|S) \approx 0$  otherwise. It is important to note that  $p(\mathbf{x}|S)$  as defined here does not represent a likelihood in a rigorous sense (e.g.  $\int p(\mathbf{x}|S) d\mathbf{x} \neq 1$ ), but it has been found to be a relatively effective and simple means of distinguishing between  $\mathbf{x} \in S$  and  $\mathbf{x} \notin S$ .

While this construction is not rigorously a density function, we note that it serves well as a “novelty detector”. Given a test vector  $\mathbf{x}$ , we compute  $\rho(\mathbf{y})$ , which we threshold with the scalar  $T$ . If  $\rho(\mathbf{y}) > T$  then  $\mathbf{x}$  is deemed to be associated with the training data (associated with the aforementioned set  $S$ ), and otherwise it is characterized as “novel” (*i.e.*, it is representative of a class of data not seen while training). Note that the quantization approach in Sec. IVA cannot be used as a novelty detector, since any test vector  $\mathbf{x}$  is necessarily mapped via quantization into one of the codes in  $\mathbf{C}$ , whether or not it is similar to the data  $\{\mathbf{x}_n\}_{n=1,N}$  used to generate the codebook  $\mathbf{C}$ . Novelty detectors have numerous applications [22], and it is of interest to note that while a novelty detector is designed here simply via RVM, SVM-based novelty detectors are less naturally defined (e.g. previous SVM novelty detectors have employed an *ad hoc* hyper-sphere construction [23]).

### C. RVM density-function estimation

Given a set of observations, a straightforward means of estimating a density function is via a histogram. The histogram can be viewed as a uniform quantizer of the continuous feature space. The number of histogram bins required to represent a distribution increases exponentially with the dimension of the feature space. For example, if *each* feature dimension is discretized uniformly into  $B$  bins, one requires  $B^D$  bins to represent a  $D$ -dimensional feature space. For large  $B$  and  $D$ , as found typically, the estimated distribution is suboptimal in presence of limited data.

We consider an approach to density-function estimation that builds upon quantization concepts discussed in Sec. IVA. We discretize the feature space into a set of codes  $\{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K\}$ . Using available training data, we calculate the pmf representative of the probability of observing each code, where  $p(\mathbf{c}_k)$  represents the probability of code

$k$ . We now have the pairs  $\{\mathbf{c}_k, p(\mathbf{c}_k)\}_{k=1,K}$ , representing a quantized form of the underlying density function, from which we seek the continuous density function  $p(\mathbf{x})$ .

We can view this as a regression problem: assume we are given samples  $\{\mathbf{c}_k, p(\mathbf{c}_k)\}_{k=1,K}$ , where  $\mathbf{c}_k$  sample feature space and  $p(\mathbf{c}_k)$  represents a mapping from the feature vector to a scalar. Given an arbitrary feature vector  $\mathbf{x}$ , *not* in  $\mathbf{C}$ , regression addresses the problem of determining the associated scalar, yielding the density function  $p(\mathbf{x})$ .

As discussed in the Introduction, flexibility in choosing a kernel makes the RVM an attractive candidate for regression-based estimation of the density function, given a set of representative input-output pairs. Although one may choose to use any problem-specific kernel, we here use the radial basis function (RBF) as a kernel for RVM-based quantization (Sec. IVA) and density estimation. Given a set of training features  $\{\mathbf{c}_k, p(\mathbf{c}_k)\}_{k=1,K}$ , regressive modeling corresponds to estimating

$$p(\mathbf{x}) \approx \sum_{j=1}^K w_j G(\mathbf{x}; \mathbf{c}_j, \mathbf{I} \sigma^2) \quad (19)$$

Note that the RVM density-function estimation chooses the most-relevant codes  $\mathbf{c}_j$ , with associated non-zero weights  $w_j$ . Therefore, when performing codebook design, for ultimate use in density-function estimation via (19), it is desirable that the codebook not be overly sparse.

We must assure that the  $p(\mathbf{x})$  in (19) integrates to one. Since the RBF basis functions can be integrated in closed form, for a  $D$ -dimensional feature space  $\int d \mathbf{x} G(\mathbf{x}; \mathbf{c}_j, \mathbf{I} \sigma^2) = (\sqrt{2\pi\sigma^2})^D$ , and therefore we normalize each of the weights as  $w'_j = w_j / [K_s (\sqrt{2\pi\sigma^2})^D]$ , where  $K_s$  represents the number of relevance vectors used in (19). More specifically,  $K_s$  represents the number of codes in (19) for which the associated weight  $w_j$  is nonzero.

It is desirable to examine the similarities and differences between the density-function estimation summarized in (19), and the traditional GMM summarized in (2). The

GMM typically is designed with a fixed number of Gaussian mixtures, and the algorithm searches for the optimal Gaussian centroid and standard-deviation. The RVM regression summarized in (19), as implemented with RBF basis functions, does not require one to set the number of Gaussian mixtures *a priori*. For simplicity, in (19) we have used basis functions with a constant standard deviation  $\sigma$ , although one may readily employ multiple basis-function types (e.g. RBF basis functions with multiple, *discrete* variances) [24,25] if desired. Moreover, we note that the RVM regression is applicable to general basis functions, not only the RBF representation employed in (19).

## V. Results

We have investigated the utility of the RVM as a tool for feature-space quantization, novelty detection and density-function estimation, for discrete and continuous HMM multi-aspect classification. We consider measured acoustic scattering data from five underwater elastic targets [12]. The scattered fields are observed from  $M$  target-sensor orientations, assuming that the constant angular sampling rate is known, but that the target identity and orientation are unknown. Insonification is effected via an acoustic wave with bandwidth approximately 11-40 KHz, corresponding to relative target dimensions of  $2.9 \leq ka \leq 10.4$ , where  $k$  is the wavenumber and  $a$  is the average radius of the targets ( $k = \omega / c$ , where  $\omega$  is the angular frequency and  $c$  is the speed of sound in water). The targets are rotationally symmetric, and the backscattered fields are observed in a plane bisecting the target axes, with 360 time-domain responses sampled at an interval of  $1^\circ$ . Features are extracted from each of the  $M$  waveforms, as discussed in the Appendix. The initial set of results is for noise-free scattering data, with the performance for noisy data discussed at the end of this section.

### A. Discrete HMM via relevance-vector quantization

As discussed in Sec. IVA, the codebook derived via Lloyd code design [13] is a function of codebook initialization, while relevance-vector quantization (RVQ) does not suffer from such problems. To address the initialization dependence of Lloyd encoding, one typically considers multiple, random initializations, and the final codebook is

selected as that yielding the minimum distortion. To compare RVQ to Lloyd encoding, we consider scattering data from each of the five targets [14] separately, and address codebook design on the associated feature vectors (see Appendix). In Fig. 1 we plot the Euclidian distortion for a codebook of dimension  $K=30$ , for each of the five targets, with codebook design realized via RVQ and Lloyd. The features are “whitened” (normalized) in advance such that each feature component is of comparable amplitude. In Fig. 1 we present the distortion for 100 random initializations for Lloyd encoding, while a single result is shown for RVQ. We see from Fig. 1 that RVQ consistently outperforms Lloyd encoding, quantified in terms of distortion, for the same codebook dimension.

In the context of multi-aspect target classification via discrete HMMs, we design a single codebook for all five targets [14], and then five discrete HMMs are constructed, one for each target. Let  $\mathbf{O}$  represent the sequence of observed features, and therefore the HMMs quantify the likelihood of  $\mathbf{O}$  for target  $T_k$ ,  $p(\mathbf{O}|T_k)$ . The observation sequence  $\mathbf{O}$  is associated with target  $T_i$  if  $p(\mathbf{O}|T_i) > p(\mathbf{O}|T_k)$ ,  $\forall T_k$ . In the results presented here the HMMs are trained using scattered-waveform sequences beginning with even angles, and testing is performed using sequences beginning with odd angles. In Fig. 2 we plot the average classification error for the discrete HMM, as a function of the sequence length. As expected, classification performance improves with increasing sequence length. The results in Fig. 2 correspond to RVQ codebook design, and we have found that such performance is essentially identical to discrete-HMM results computed using Lloyd codebook design (to within numerical variability). This suggests that the discrete HMM is not overly sensitive to relatively small differences in distortion between two distinct codebooks for the same data set. In Fig. 2, and in subsequent examples, the angular sampling rate between consecutive waveforms in the sequence is  $\delta\theta = 5^\circ$ .

## B. Continuous HMM

In the next set of examples we compare implementation of the continuous HMM when the state-dependent density functions  $p(\mathbf{v}_m | s_i, T_k)$  in (1) are computed via a GMM

representation and via the RVM density-function estimation discussed in Sec. IV. Recall that for an HMM used for multi-aspect target classification [14], each state is characteristic of a set of contiguous target-sensor orientations for which the statistics of the associated feature vectors are stationary. To examine the effectiveness of GMM and RVM to model the state-dependent feature statistics, in Fig. 3 we consider data for Target 1, from the set of five elastic targets [14]. The data was partitioned into five states, corresponding to angular sectors  $[0^\circ, 29^\circ]$ ,  $[30^\circ, 59^\circ]$ ,  $[60^\circ, 70^\circ]$ ,  $[71^\circ, 80^\circ]$  and  $[81^\circ, 90^\circ]$ . For each state a GMM and RVM model was developed to characterize variation of the state-dependent feature-vector statistics. We then submitted each feature vector to the five state-dependent density functions, to examine whether the model associates the feature vector with the appropriate state (in a maximum-likelihood sense). In Fig. 3 the vertical axis identifies the angle of the feature vector, and the horizontal axis denotes the five states, and the colors represent the state-dependent likelihood. We see from Fig. 3 that the RVM yields a relatively “clean” partitioning of the data into states, while the GMM yields some inconsistencies between states. It is important to emphasize, however, that the HMM training subsequently optimizes the state partitions and the state-dependent statistics [11], so inaccuracies seen in Fig. 3, which represent an *initialization* to the HMM training, can be addressed via HMM training. Nevertheless, the results in Fig. 3 indicate that the RVM regression performs well in the context of density-function estimation, using the algorithm discussed in Sec. IV.

In Fig. 4 we present continuous-HMM classification performance as a function of sequence length, as in Fig. 2. Comparing Figs. 2 and 4, we note a significant improvement in performance yielded by the continuous HMM, *vis-à-vis* its discrete counterpart. Three curves are plotted in Fig. 4, for three different means of modeling the state-dependent statistics: (1) traditional GMM; (2) RVM-based density-function estimation, as described in Sec. IVC; and (3) RVM-based “novelty detection”, as discussed in Sec. IVB. In the context of (3), the probability that feature vector  $\mathbf{x}$  is associated with state  $S$  is modeled via the sigmoidal function discussed in Sec. IV,  $p(\mathbf{x}|S) = \rho(y(\mathbf{x}))$ . Approach (3) is a balance between simple vector quantization (with the associated distortion) and the relative complexity of full density-function estimation.

We see from Fig. 4 that the three approaches yield comparable performance, although the two RVM-based density-function estimation algorithms consistently provides superior performance. It is noteworthy that the relatively simple sigmoidal model of the density function yields performance comparable to that of the more-rigorous density-function estimation.

The results in Figs. 2 and 4 have considered noise-free data. We also considered classification when additive *colored* noise was added to the data. The colored noise was generated by convolving white Gaussian noise with the incident pulse, generating random data characteristic of incident-wave scattering from a random environment (of interest for modeling scattering off the sea bottom of surface) [12]. We have performed detailed studies of HMM performance on such noisy data [14], and therefore an extensive discussion of such is not reported here. In summary, for noisy data we have found HMM performance relatively independent of whether one employs traditional HMM design (Lloyd discretization or GMM density function estimation) or the RVM approach developed here. We should emphasize, however, that the “traditional” HMM designs employed here are based on extensive experience [14] in choosing such parameters as the number of Lloyd codes and the number of state-dependent Gaussian mixtures. This experience is accrued primarily through trial and error. It is therefore notable that the RVM-based HMMs yield comparable performance, since such issues as choosing the proper number of codes and the appropriate number of Gaussian mixtures is done autonomously via the RVM algorithm (by choosing the “relevant” training data).

### **C. HMM-based novelty detection**

In the previous paragraph we discussed the results of studies with *additive* noise or clutter. Often a more-interesting problem involves distinguishing a target of interest from a “false” target. A false target is a localized scatterer, yielding a sequence of scattered waveforms, just as the “true” targets of interest. An example of a false target may be a rock or a man-made scatterer (but not one of the targets of interest). We assume that the classifier(s) are trained to identify the “true” targets of interest, but that the

“false” targets have not been seen prior to testing (since the false targets constitute an infinite class). To examine the utility of the RVM-based HMM for distinguishing true targets (seen while training) from false targets (not seen while training), we have performed the following test. We designed five HMMs for the five targets considered in the previous example. In addition, we consider six false targets, described in [26], representative of the following underwater targets: (1) a blunt-bullet-shape filled metallic object (the flat nose is 11.2 cm in diameter, the flat end is 27.3 cm in diameter, and the length is 156.8 cm); (2) a truncated-cone-shape filled plastic object (the small end is 50 cm in diameter, the large end is 98 cm in diameter, and the height of the cone is 47 cm); (3) a 50 gallon water-filled drum (85 cm long and 57 cm in diameter); (4) a irregular shaped limestone rock; (5) a somewhat smooth granite rock; and (6) a wooden log that has been saturated in water (the log is roughly cylindrical, it is 114.5 cm long, one end is cut smooth and is about 18 cm in diameter and the other end is very irregular and is roughly 9 cm in diameter). All the false targets are submerged in water and insonified in a horizontal plane. Each target response is sampled at a  $5^{\circ}$  angular interval, resulting in 72 backscattered responses per target. All false-target responses have the same frequency support as the underwater elastic shells discussed previously. These six targets were not seen by the classifiers prior to testing (i.e., ideally they are deemed by the classifiers as “novel”).

We consider binary classification: the target under test is or is not one of the five shells seen while training. The performance is quantified in terms of the receiver operating characteristic (ROC) shown in Fig. 5. We have five trained HMMs, one for each shell target considered in the previous examples. Given an unknown sequence of observations, we calculate the likelihood of the given observation sequence conditioned on each trained HMM. The maximum of the five HMM outputs is compared against a variable threshold, yielding the probability of detection as a function of the false alarm rate. We see in Fig. 5 that the continuous HMMs, with RVM-based density-function estimation, yields encouraging performance in separating targets seen before (the shells) from the false targets not seen previously by the classifier. As expected, as the length of the data sequence  $M$  increases, we observe an improvement in classification performance.

## VI. Conclusions

The relevance vector machine (RVM) [5] has been examined as a tool to address two ubiquitous operations in pattern recognition: quantization and density-function estimation. The RVM seeks to determine the “most-relevant” training data for each of these operations. For quantization, the RVM attempts to determine those feature vectors that are most representative of the training data  $\{\mathbf{x}_n\}_{n=1,N}$ , with the RVM kernel (or basis function) used to quantify the similarity of any two feature vectors. An advantage of RVM codebook design is that one need not specify *a priori* the number of anticipated codes.

The quantized data yields a probability mass function (pmf) that quantifies the probability of observing each code, as defined by the training data. If we have  $K$  codes, this yields the pairs  $\{\mathbf{x}_k, p(\mathbf{x}_k)\}_{k=1,K}$ , which we have used as training data for an RVM-based estimation of the underlying density function, via RVM regression. If one employs a radial basis function (RBF) kernel [27], the RVM density-function estimation is closely related to the widely employed Gaussian-mixture model (GMM) [11]. The advantage of the RVM design is that one need not set *a priori* the number of Gaussian mixtures. The RVM can also be extended to arbitrary basis functions (not just RBF), and therefore it is more general than GMMs. It is also more general than SVM-based regression (which requires Mercer kernels) [28].

We have employed RVM-based codebook design and density-function estimation in discrete and continuous HMMs, respectively, for multi-aspect target classification [12]. Measured acoustic-scattering data from several underwater targets have been considered. It has been demonstrated that the RVM-based design compares very favorably with the performance of highly optimized HMMs (e.g. the codebook size and number of GMMs was optimized via an extensive trial-and-error procedure [12]).

## Appendix: Features Used in HMM for Target Classification

We assume access to multiple scattered waveforms collected at multiple target-sensor orientations. The scattered waveforms are assumed measured in the time-domain, wherein feature extraction is performed, although the approach is also of utility for frequency-domain scattering data. Features are extracted from each of  $M$  signals, corresponding to scattered waveforms observed at  $M$  target-sensor orientations. We have extracted temporal moments and matching-pursuits [29] parameters to concisely represent the features of the time-domain scattering data. While extracting the temporal moments, we normalize the transient waveform, such that it can be viewed as a density function. The central higher-order moments yield information on the temporal extent and overall shape of the scattered waveform. For discrete time-domain signals  $y(t_k)$ ,  $k=1, 2, \dots, K$ , the temporal probability density function  $p(t_k)$  and the central moments  $m_l$  are calculated as

$$p(t_k) = \frac{y^2(t_k)}{\sum_{k=1}^K y^2(t_k)} \quad (\text{A.1a})$$

$$m_1 = \sum_{k=1}^K t_k p(t_k) \quad m_{l>1} = \sum_{k=1}^K (t_k - m_1)^l p(t_k) \quad (\text{A.1b})$$

The three moment-based features we use for target classification are the standard deviation, skewness and kurtosis, defined respectively as  $\sigma = \sqrt{m_2}$ ,  $m_3 / \sigma^3$  and  $m_4 / \sigma^4$ . The first moment  $m_1$  (or mean) is highly sensitive to target-sensor distance, and therefore it is not used as a feature.

Matching pursuits (MP) is a well-known feature-extraction technique [5], in which a dictionary of parametric waveforms is predefined. For acoustic-wave scattering, we design a matching-pursuits dictionary, composed of general wavefronts and resonances [3], matched to the anticipated scattering physics. Wavefronts have localized temporal support, while resonances capture global properties of the waveform. We have performed three iterations of the matching-pursuits algorithm to extract the three most significant dictionary elements from each scattered waveform. The use of three MP

iterations is based on the properties of the scattering data considered here (see Sec. V), while in general the appropriate number of iterations is data dependent.

The complete feature vector associated with each transient target response  $s_n$  is given by

$$\gamma_n = \{\alpha_1, \omega_1, \alpha_2, \omega_2, \tau_2 - \tau_1, \alpha_3, \omega_3, \tau_3 - \tau_1, \sqrt{m_2}, m_3 / \sigma^3, m_4 / \sigma^4\} \quad (\text{A.2})$$

where  $\omega$ ,  $\alpha$  and  $\tau$  represent respectively the frequency, decay rate and the time delay of the  $i$ th extracted waveform from the physics-based MP dictionary (see [29]). Note that we use the *relative* time delay  $\tau_k - \tau_{k-1}$  between consecutive extracted time-domain MP dictionary elements, rather than the absolute time delay (e.g.  $\tau_k$ ). The former is a characteristic of target geometry, while the latter is dictated by the variable target-sensor distance.

## References

- [1] J. Makhoul, S. Raucos and H. Gish, "Vector quantization in speech coding," *Proc. IEEE*, Vol. 73, pp. 1551-1558, Nov. 1985.
- [2] H.L. Van Trees, *Detection, Estimation, and Modulation Theory*. J. Wiley and Sons, New York, NY, 1968.
- [3] T. Kohonen, *Self-Organizing Maps*. Springer Series in Information Sciences, Vol. 30, Third, extended edition, 2001.
- [4] A. Gelman, J. B. Carlin, H. S. Stern and D. B. Rubin, *Bayesian Data Analysis*, Chapman & Hall, 1995.
- [5] M. E. Tipping, "Sparse bayesian learning and the relevance vector machine," *Journal of Machine Learning Research 1*, pp. 211-244, 2001.
- [6] V. N. Vapnik, S. E. Golowich, and A. J. Smola, "Support vector method for function approximation, regression estimation and signal processing," in *Advances in Neural Information Processing Systems 9*. MIT Press, 1997.
- [7] B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors. "Advances in Kernel Methods: Support Vector Learning," MIT Press, 1999a.
- [8] B. Scholkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Mueller, G. Raetsch, and A.

- J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Networks*, Vol. 10, pp. 1000-1017, 1999.
- [9] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Comm.*, Vol. 28, pp. 84-95, Jan. 1980.
- [10] J. Picone, "Continuous speech recognition using hidden Markov models," *IEEE Acoust. Speech Signal Proc. Mag.*, Vol. 7, pp. 26-41, July 1990.
- [11] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, Vol. 7, pp. 257-285, Feb. 1989.
- [12] P. Runkle, L. Carin, L. Couchman, T. J. Yoder, J. Bucaro, "Multiaspect target identification with wave-based matched pursuits and continuous hidden Markov models," *IEEE Trans. Pattern Analysis Machine Intell.*, Vol. 21, pp. 1371-1378, Dec. 1999.
- [13] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, pp. 128-137, 1982.
- [14] P. R. Runkle, P. K. Bharadwaj, L. Couchman, and L. Carin, "Hidden Markov models for multi-aspect target classification," *IEEE Trans. Signal Proc.*, Vol. 47, pp. 2035 –2040, July 1999.
- [15] J. Deller, J. Proakis and J. Hansen, *Discrete-Time Processing of Speech Signals*. Englewood Cliffs, Prentice-Hall, NJ, 1993.
- [16] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Annals of Mathematical Statistics*, vol. 41, pp. 164-171, 1970.
- [17] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, Vol. 13, pp. 260-269, April 1967.
- [18] Li-Yi Wei and M. Levoy, "Fast Texture Synthesis using Tree-structured Vector Quantization," SIGGRAPH, *Proc. Computer Graphics*, 2000.
- [19] A. C. Faul and M. E. Tipping, "Analysis of Sparse Bayesian Learning," *Advances in Neural Information Processing Systems 14*. MIT Press.
- [20] D. J. C. MacKay, "The evidence framework applied to classification networks," *Neural Computation*, Vol. 4, pp. 720-736, 1992.
- [21] I. T. Nabney, "Efficient training of RBF networks for classification," *Proc. Ninth*

- Int. Conf. Artificial Neural Networks (ICANN99)*, IEE, pp. 210-215, 1999.
- [22] C. M. Bishop, “Novelty detection and neural Network validation,” IEE Proceedings on Applications of Neural Networks, May 1994.
- [23] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, Cambridge, UK, 2000.
- [24] P. A. Devijver and J. Kittler, *Pattern Recognition Theory and Applications*. NATO ASI Series, Springer-Verlag, 1987.
- [25] C. K. I. Williams, “Prediction with Gaussian processes: from linear regression to linear prediction and beyond,” in *Learning in Graphical Models*, M. I. Jordan, editor, MIT Press, pp. 599-621, 1999.
- [26] M.R. Azimi-Sadjadi, D. Yao, Q. Huang, and G.J. Dobeck, “Underwater target classification using wavelet packets and neural networks,” *IEEE Trans. Neural Networks*, vol. 11, pp. 784-794, May 2000.
- [27] T. S. Jaakkola and M. I. Jordan, “Bayesian logistic regression: a variational approach,” *Proc. Conf. Artificial Intelligence and Statistics*, Ft Lauderdale, FL, 1997.
- [28] J. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large Margin Classifiers*, A. J. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, MIT Press, 2000.
- [29] M. McClure and L. Carin, “Matching pursuits with a wave-based dictionary,” *IEEE Trans. Signal Proc.*, Vol. 41, pp. 2912-2927, Dec. 1997.

## Figure Captions

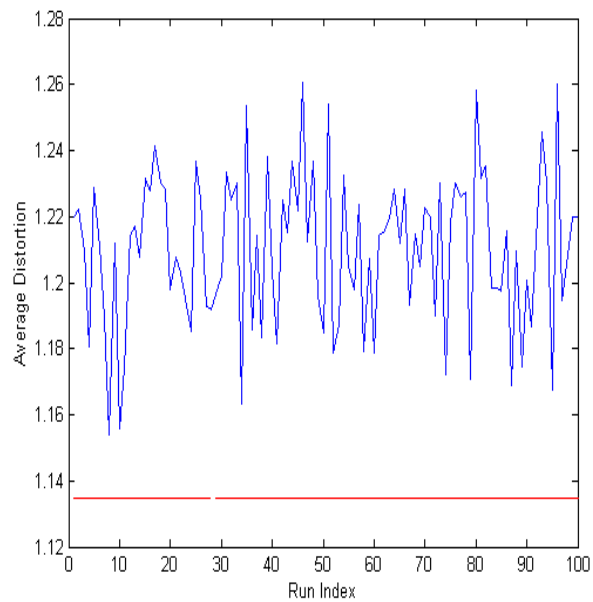
**Figure 1.** Distortion introduced by quantizing the features from five submerged elastic targets [14]. The horizontal axis identifies each of the 100 random initializations of the codebook, in the context of Lloyd quantization. The RVM-based quantization is deterministic, and therefore its distortion is constant. Results are shown for  $K=30$  codes. (a) Target 1, (b) target 2, (c) target 3, (d) target 4, (e) target 5

**Figure 2.** Probability of classification error for five elastic targets [14]. A discrete HMM is designed for each target, and the data is associated with that HMM that yields the largest likelihood. RVM-based codebook design is employed.

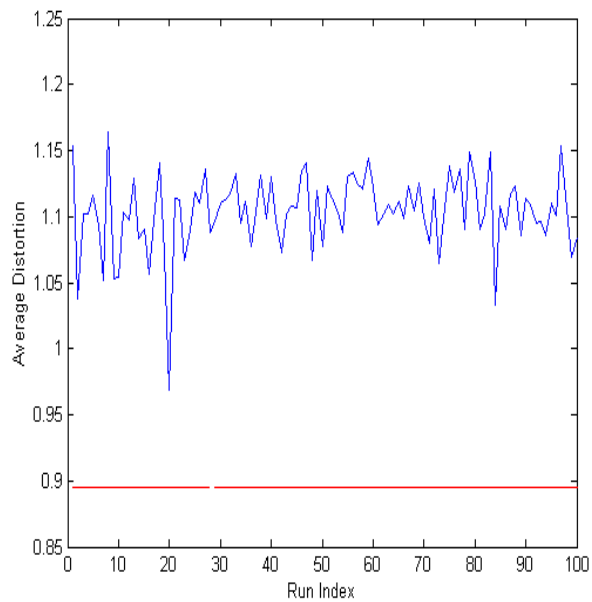
**Figure 3.** Density functions are designed to characterize the features of the five states characteristic of Target 1. Subsequently, the feature vector associated with each target-sensor orientation (angle) for Target 1 is submitted to the density function associated with each state. Here is plotted (in color scale) the likelihood that the feature vector is associated with each of the five states. (a) GMM density-function estimation, (b) RVM-regression density-function estimation

**Figure 4.** Probability of classification error for five elastic targets [14], as in Fig. 2. A continuous HMM is designed for each target, and the data is associated with that HMM that yields the largest likelihood. Results are shown for state-dependent density function estimation via GMM, RVM-regression (Sec. IVC), and RVM novelty detection (Sec. IVB).

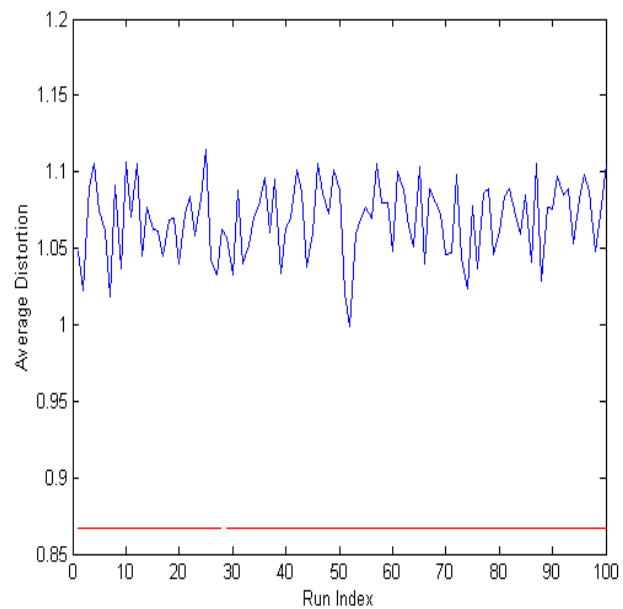
**Figure 5.** Receiver operating characteristic for distinguishing five known targets (the five shells considered in Figs. 1-4) – seen while training the classifiers – from six “false” targets [26], not seen when training the classifier. The continuous HMMs used to characterize the five shell targets employed RVM regression for state-dependent density-function estimation. Results are shown as a function of multi-aspect sequence length, using  $5^\circ$  angular sampling between consecutive measurements.



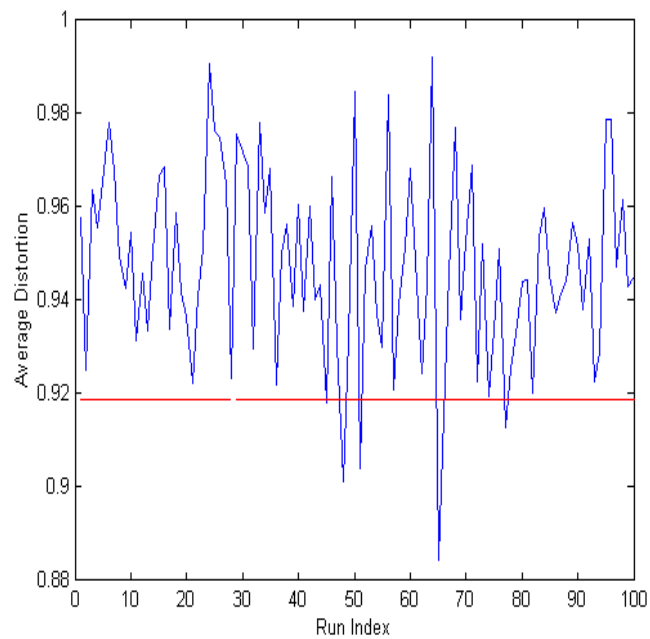
(a)



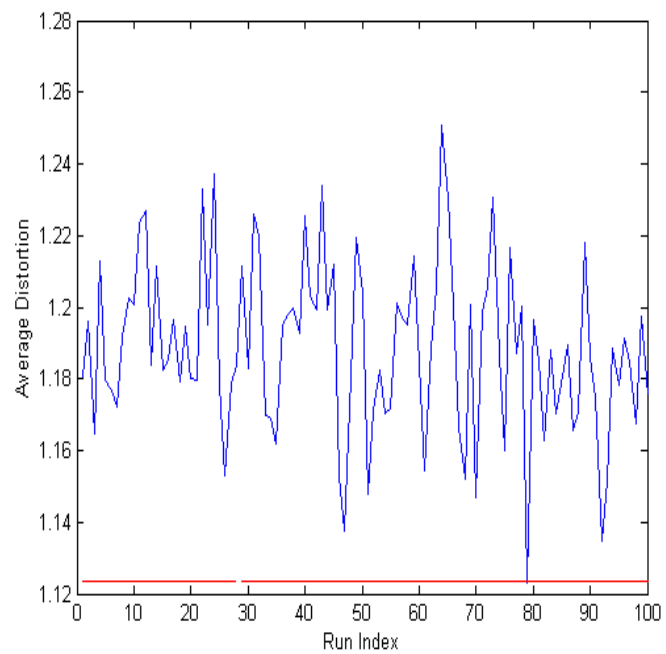
(b)



(c)



(d)



(e)

Figure 1

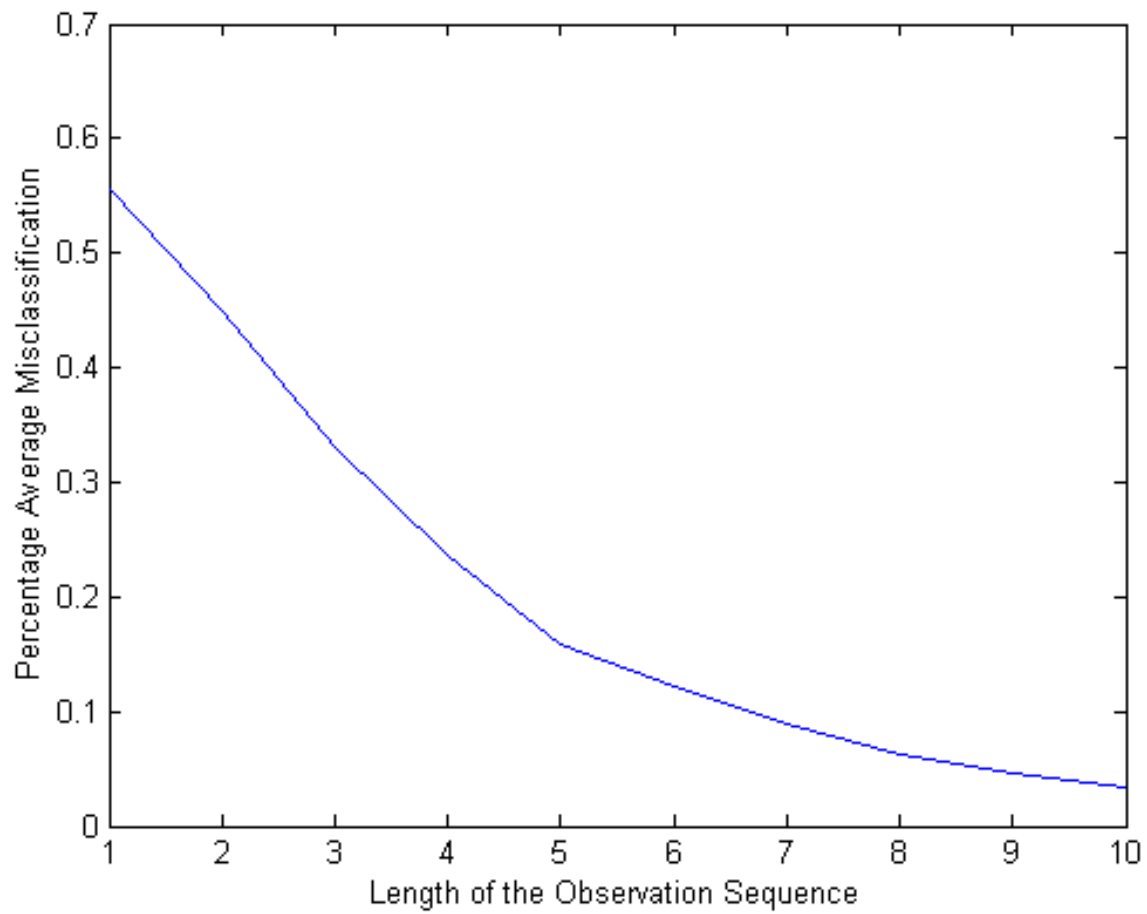
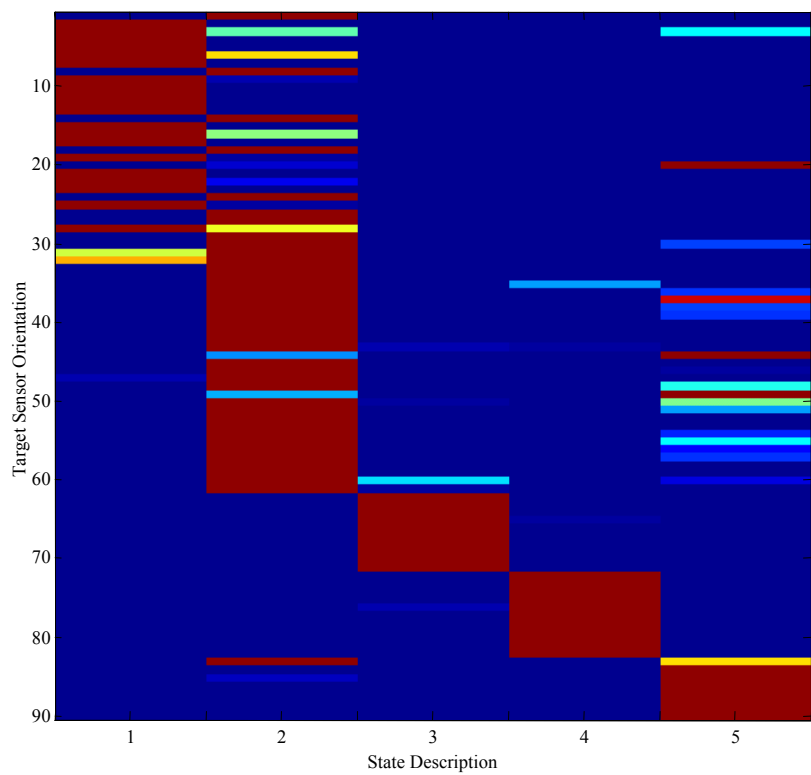
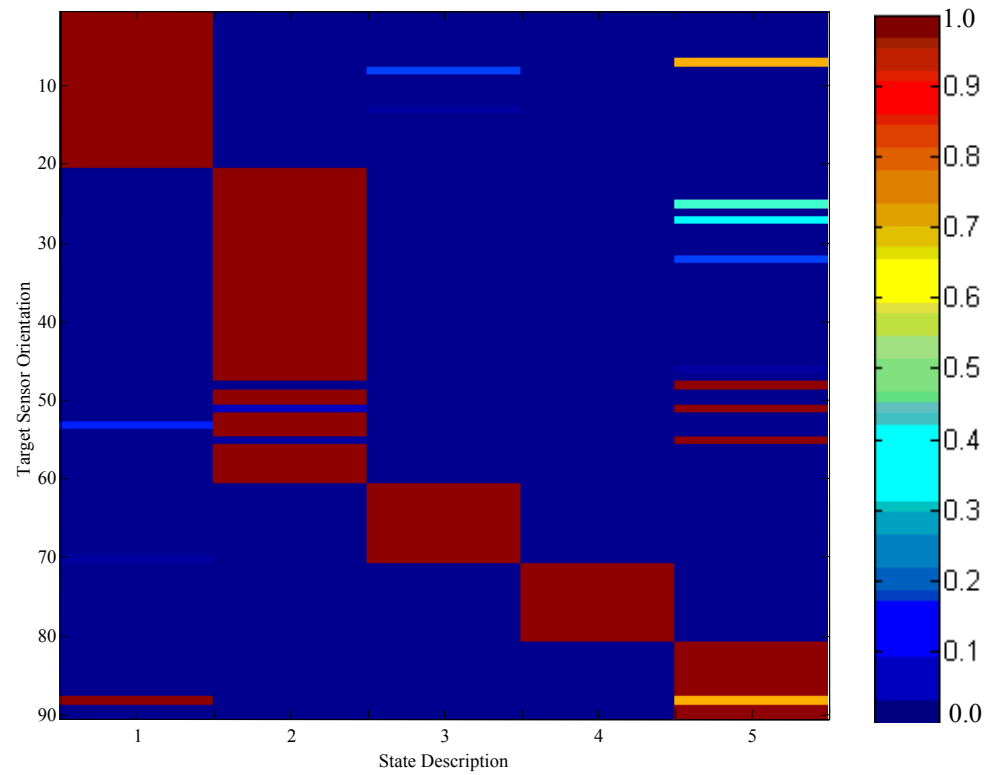


Figure 2



(a)



(b)

Figure 3

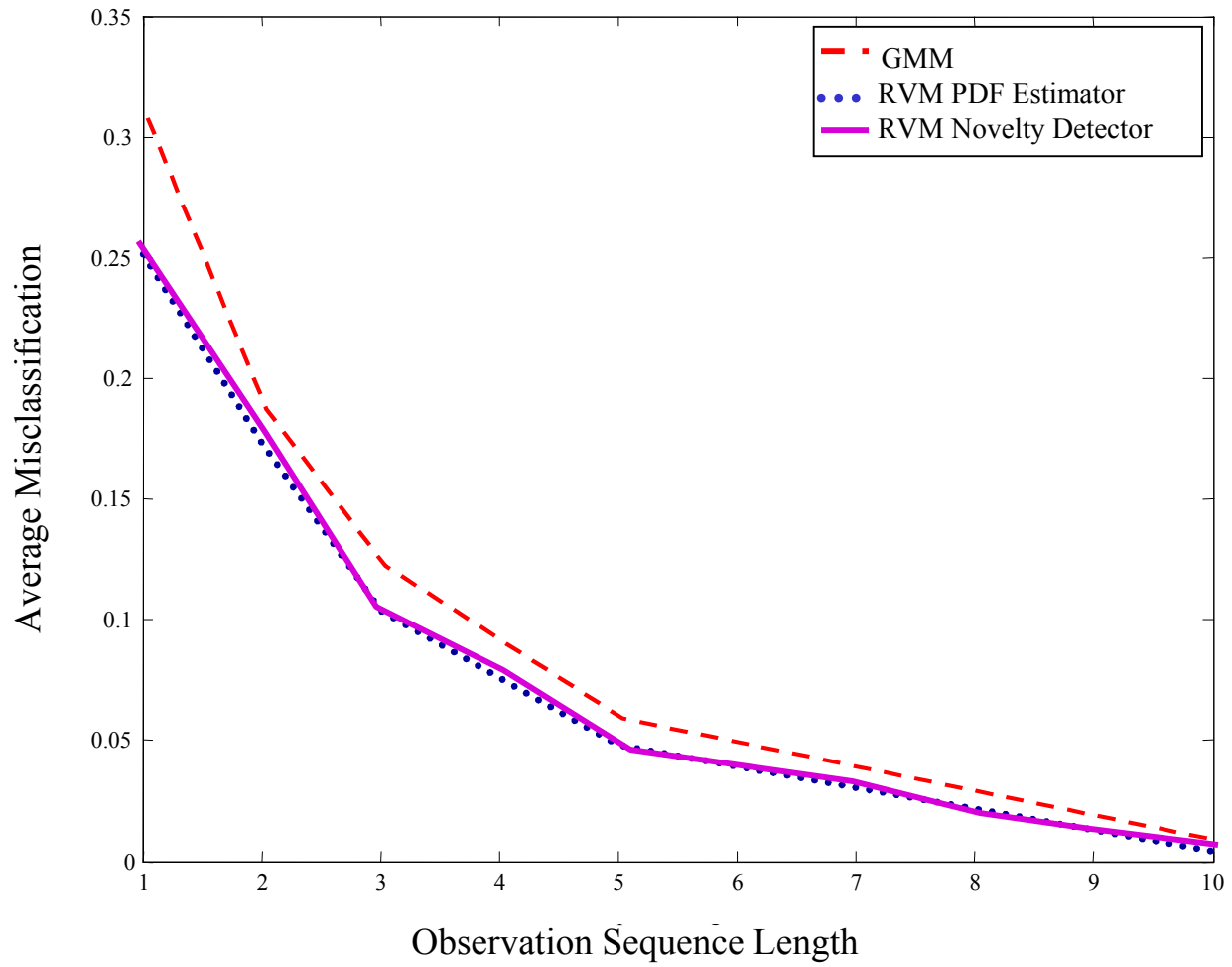


Figure 4

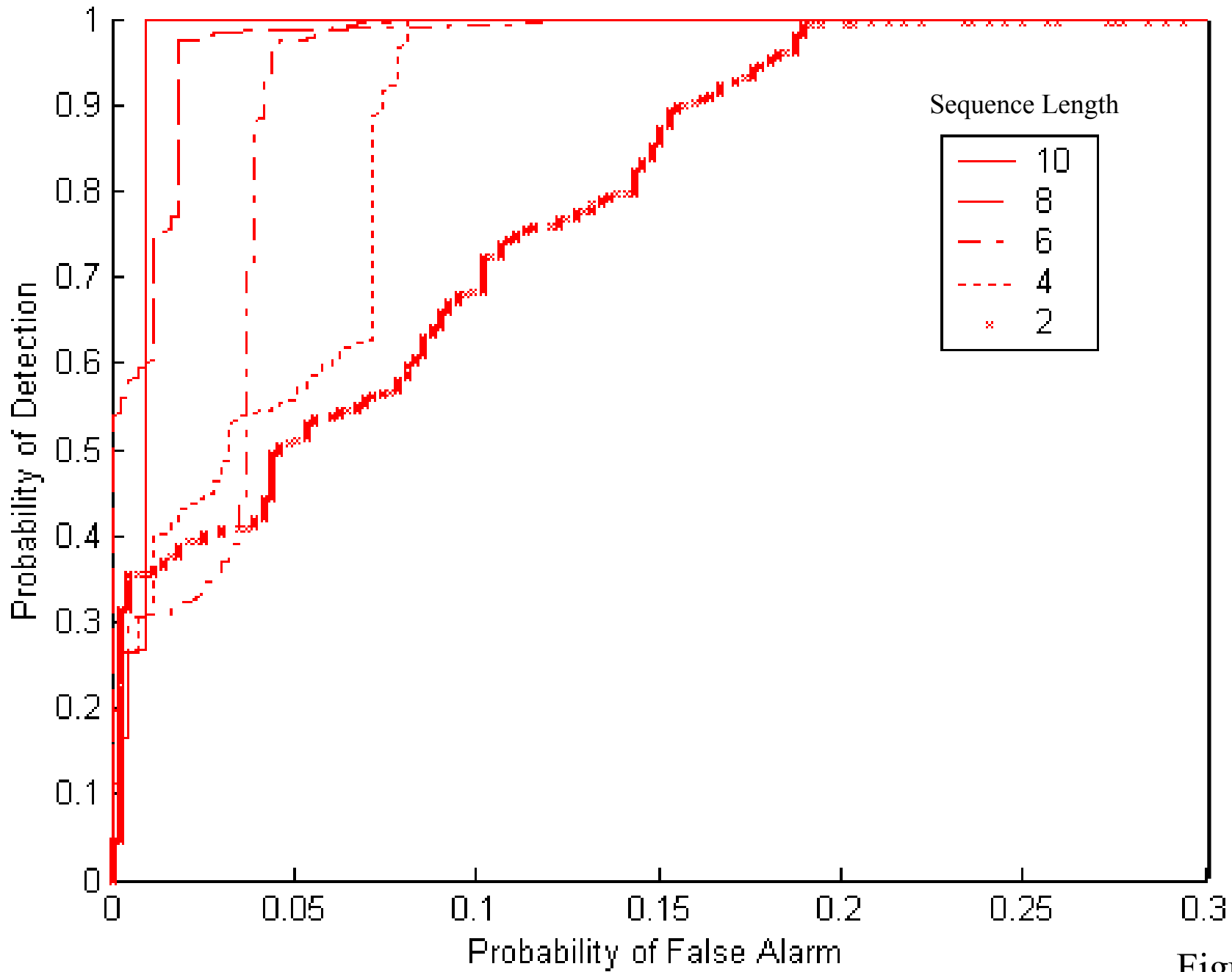


Figure 5