SVM-based Learning for Multiple Model Estimation

Vladimir Cherkassky and Yunqian Ma

Department of Electrical and Computer Engineering

University of Minnesota

Minneapolis, MN 55455

{cherkass,myq}@ece.umn.edu

Abstract:

This paper presents new constructive learning methodology for multiple model estimation. Under multiple model formulation, training data are generated by several (unknown) statistical models, so existing learning methods (for classification or regression) based on a single model formulation are no longer applicable. We describe general framework for multiple model estimation using SVM methodology. The proposed constructive methodology is analyzed in detail for regression formulation. We also present several empirical examples for multiple-model regression formulation. These empirical results illustrate advantages of the proposed multiple model estimation approach.

1. Introduction

This paper describes constructive learning methods for multiple model regression formulation proposed in [Cherkassky and Ma, 2002]. Under this formulation, available (training) data $(\mathbf{x}_i, y_i), i = 1,2,..., n$ are generated by several (unknown) regression models, so the goal of

learning is two-fold, i.e., partitioning of available data into several subsets and estimating regression models for each subset of available data. Hence, the problem of multiple model estimation is inherently more complex than traditional supervised learning where all training data are used to estimate a single model. Conceptually, there are two principal approaches for designing constructive learning methods for multiple model estimation:

- (1) First partition available data into several subsets, then estimate model parameters for each subset of data;
- (2) First estimate a dominant model using all available data, and then partition the data into several subsets.

Under the *first approach*, the learning starts with a clustering step (unsupervised learning) followed by supervised learning on a subset of available data. Practical implementation of this approach is described in [Tanaka, 2001] using the framework of mixture density estimation, where each (hidden) model is modeled as a component in a mixture. The main practical limitations of this approach are as follows:

- Inherent complexity of density estimation (with finite samples). There is theoretical and empirical evidence that density estimation is much harder than supervised learning (regression) with finite samples [Vapnik, 1999, Cherkassky and Mulier, 1998];
- Moreover, the setting of multiple model estimation leads to clustering/density estimation in *local* regions of the input space. That is, for a given input value, there may be several distinctly different output (response) values, corresponding to different models. Hence, data partitioning (clustering) should be based on different response values, and this leads clustering/density estimation in *local regions* of the input space. Practical implementation of such clustering using a (small) portion of available data becomes very problematic with finite samples due to the curse of dimensionality.

Under the *second approach*, we apply robust regression estimator to all available data in order to estimate a dominant model (where a dominant model is a model that describes the majority of data samples). Clearly, this approach is better than clustering/density estimation strategy because:

- It is based on regression rather than density estimation formulation;
- It uses all available data (rather than a portion of the data) for model estimation.

Hence, in this paper we focus on implementation of the second approach. The main practical requirement for this approach is availability of robust regression algorithm, where 'robustness' refers to capability of estimating a single (dominant) model when available data are generated by several (hidden) models possibly corrupted by additive noise. This notion of robustness is somewhat different from traditional robust estimation techniques. This is because standard robust methods are still based on a single-model formulation, where the goal of robust estimation is resistance (of estimates) with respect to unknown noise models. Recently, robust statistical methods have been applied to computer vision problems that can be described using multiple model formulation. In these studies, existence of multiple models (in the data) is referred to as 'the presence of structured outliers' [Chen et al, 2000]. Empirical evidence suggests that traditional robust statistical methods usually fail in the presence of structured outliers, especially when the model instances are corrupted by significant noise [Chen et al, 2000]. This can be explained as follows. When the data are generated by several (hidden) models, each of the data subsets (structures) has the same importance, and relative to any one of them the rest of the data are 'outliers'. As a result, the notion of the breakdown point (in robust statistics) which describes processing the majority of data points loses its meaning under multiple-model formulation. Moreover, traditional robust estimation methods cannot handle more than 30% of outliers in the

data [Rousseeuw and Leroy, 1987]. Hence, we need to develop new constructive learning methodologies for multiple model estimation.

This paper describes new learning algorithms for multiple model estimation based on SVM methodology. The following simple example illustrates desirable properties of robust algorithms for multiple model estimation. Consider a data set comprising two (linear) regression models: dominant model M1 (70% of the data) and secondary model M2 (30% of the data) shown in Fig.1a. The data are corrupted by additive gaussian noise (with standard deviation 0.1). Results in Fig. 1b show the model estimated by (linear) SVM regression with insensitive zone ε=0.084, and the model estimated by ordinary least squares (OLS). Both estimation algorithms use all available data (generated by both models). OLS method produces rather inaccurate model, whereas SVM produces very accurate estimate of the dominant model M1. Further, data set in Fig. 1c shows another data set generated using the same dominant model M1 but completely different secondary model M2. Application of SVM (with the same ε-value) to this data set yields (almost) identical estimate of the dominant model M1, as shown in Fig. 1d. However, application of OLS to this data set yields an estimate of M1 (shown in Fig. 1d) that is completely different from the estimate in Fig. 1b. Note that the number of (hidden) models is unknown (to an estimation method), and we use two models only to simplify presentation. This example shows an existence of robust regression algorithm that can be used to accurately estimate a single (dominant) model from a data set generated by several models. Here robustness (in the context of multiple model estimation) refers to accurate estimation of the dominant model and the stability of such estimates in spite of significant potential variability of data generated by secondary model(s). Going back to example in Fig. 1: after the dominant mode M1 has been identified by a robust regression method, it may be possible to identify and remove data samples generated by M1, and then apply robust regression to the remaining data in order to estimate the next model.

Hence, we propose general methodology for multiple model estimation based on successive application of robust regression algorithm to available data, so that during each (successive) iteration, we estimate a single (dominant) model and then partition the data into two subsets. This iterative procedure is outlined next:

Table 1: PROCEDURE for MULTIPLE MODEL ESTIMATION

Initialization: Available data = all training samples.

• **Step 1: Estimate dominant model**, i.e. apply robust regression to available data, resulting in a dominant model M1 (describing the majority of available data).

• **Step 2: Partition available data** into two subsets, i.e. samples generated by M1 and samples generated by other models (the remaining data). This partitioning is performed by analyzing available data samples ordered according to their distance (residuals) to dominant model M1.

•**Step 3: Remove subset of data generated by dominant model from available data.**

Iterate: Apply Steps 1-3 to available data until some stopping criterion is met.

It is important to note here that the above procedure relies heavily on the existence of robust (regression) estimation algorithm that can reliably identify and estimate a dominant model (describing majority of available data) in the presence of (structured) outliers and noise. The existence of such robust regression method based on Support Vector Machine (SVM) regression has been demonstrated in the example shown in Fig.1. However, results in Fig. 1 are purely empirical and require further explanation, since the original SVM methodology has been developed for single model formulation. Even though SVM is known for its robustness, its application for multiple model estimation is far from being obvious.

In the next section we provide conceptual and theoretical justification for using SVM method in the context of multiple model estimation, i.e. we explain why SVM regression can be used in Step 1 of an iterative procedure outlined above. Section 3 describes details and implementation of the partitioning Step 2. In addition, Section 3 provides guidelines on selection of meta-parameters for SVM regression, important for practical application of SVM. Section 4 presents empirical results for multiple model estimation. These results show successful application of the proposed SVM-based multiple model estimation for both linear and nonlinear regression models. Section 5 presents a clustering algorithm based on multiple model estimation, where the goal (of learning) is to partition available (training) data into several subsets, such that each subset is generated by a different model. Finally, conclusions are given in Section 6.

2. SVM Regression for robust model estimation

It is well known that Support Vector Machine (SVM) methodology is robust under standard single-model estimation setting [Vapnik, 1999]. That is, SVM approach works well for estimating indicator function (pattern recognition problem) and for estimating real-valued function (regression problem) from noisy sparse training data. In this section, we demonstrate SVM robustness under multiple model estimation setting, i.e. we explain why SVM regression provides stable and accurate estimates of the dominant model when available (training) data are generated by several (hidden) models.

First, we review standard (linear) SVM regression formulation [Vapnik, 1995]. The goal of regression is to select the 'best' model from a set of admissible models (aka approximating functions) $f(\mathbf{x}, \boldsymbol{\omega})$, where $\boldsymbol{\omega}$ denotes (generalized) set of parameters. The best model provides

good prediction accuracy (generalization) for future (test) samples, and its selection is performed via minimization of some loss function (aka empirical risk) for available training data $(\mathbf{x}_i, y_i), i = 1,2,..., n$. The main feature of SVM regression responsible for its attractive properties is the notion of ε -insensitive loss function:

$$L(y, f(\mathbf{x}, \boldsymbol{\omega})) = \begin{cases} 0 & if \mid y - f(\mathbf{x}, \boldsymbol{\omega}) \mid \leq \varepsilon \\ \mid y - f(\mathbf{x}, \boldsymbol{\omega}) \mid \leq \varepsilon, & otherwise \end{cases} \quad (1)$$

Here the linear nature of the loss function accounts for SVM robustness whereas the ε -insensitive zone leads to sparseness of SVM regression models [Vapnik, 1995].

Let us consider (for simplicity) linear SVM regression:

$$f(\mathbf{x}, \boldsymbol{\omega}) = < \boldsymbol{\omega}, \mathbf{x} > + b \quad (2)$$

SVM approach to linear regression amounts to (simultaneous) minimization of ε -insensitive loss function (1) and minimization of the norm of linear parameters $\| \boldsymbol{\omega} \|^2$ [Vapnik, 1995]. This can be formally described by introducing (non-negative) slack variables $\xi_i, \xi_i^* \;\; i = 1,...n$, to measure the deviation of training samples outside ε -insensitive zone. Thus SVM regression can be formulated as minimization of the following functional:

$$\frac{1}{2} \| \boldsymbol{\omega} \|^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i^*) \quad (3)$$

$$\text{Subject to constraints } \begin{cases} y_i - < \boldsymbol{\omega}, \mathbf{x}_i > - b \leq \varepsilon + \xi_i \\ < \boldsymbol{\omega}, \mathbf{x}_i > + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i \, , \xi_i^* \geq 0, i = 1,..., n \end{cases}$$

The constant C determines the trade off between the model complexity (flatness) and the degree to which deviations larger than ε are tolerated in optimization formulation. This optimization problem can be transformed into the dual problem [Vapnik, 1995], and its solution is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) < \mathbf{x}_i, \mathbf{x} > + b \tag{4}$$

with coefficient values in the range $0 \le \alpha_i^* \le C$, $0 \le \alpha_i \le C$. In representation (4), typically only a fraction of training samples appear with non-zero coefficients, and such training samples are called support vectors. For most applications, the number of support vectors (SVs) n_{SV} is usually smaller than the number of training samples. Thus, with ε -insensitive loss, SVM solutions are typically sparse.

For nonlinear regression problem, SVM approach performs first a mapping from the input space onto a high-dimensional feature space and then performs linear regression in the high-dimensional feature space. The SVM solution is

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}) + b \tag{5}$$

where the $K(\mathbf{x}_i, \mathbf{x})$ is a kernel function. The choice of the kernel functions and kernel parameters is determined by a user and is (usually) application-dependent. In this paper, we use RBF kernels

$$K(\mathbf{x}_i, \mathbf{x}) = \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}_i \|^2}{2p^2}\right) \tag{6}$$

where p is RBF width parameter.

Next, we explain why SVM regression is suitable for estimating the dominant model under multiple model formulation. We assume, for simplicity, linear SVM formulation (4); however similar arguments hold for nonlinear SVM as well. The objective function in (3) can be viewed as a primal problem, and its dual form can be obtained by constructing Lagrange function and introducing a set of (dual) variables [Vapnik, 1995]. For the dual form, the so called Karush-Kuhn-Tucker (KKT) conditions hold at the optimal solution, which state that the product between dual variables and constraints has to vanish:

$$\alpha_i\,(\varepsilon+\xi_i-y_i+<\boldsymbol{\omega},\mathbf{x}_i>+b)=0$$

$$\alpha_i^*\,(\varepsilon+\xi_i{}^*-y_i-<\boldsymbol{\omega},\mathbf{x}_i>-b)=0 \qquad (7)$$

$$(C-\alpha_i)\xi_i=0$$

$$(C-\alpha_i^*)\xi_i{}^*=0$$

We may further analyze properties of coefficients $\alpha_i^{(*)}$ (dual variables) in the SVM solution evident from KKT conditions [Smola and Schölkopf, 1998]. First, only samples with corresponding $\alpha_i^{(*)}=C$ lie outside the ε –insensitive zone. Second, condition $\alpha_i\alpha_i^*=0$, implies that dual variables α_i and α_i^* cannot be simultaneously be nonzero, since nonzero slack cannot happen in both directions. Let us analyze contribution of training samples in SVM solution (4). As shown in Fig. 2, all data samples can be divided into 3 subsets: data points inside the ε -tube (labeled as '×' in Fig.2.), data points on the ε -tube (border) (labeled as '○' in Fig. 2.) and data points outside the ε -tube (label as '□'in Fig.2.). Note that data samples inside the ε -tube cannot be support vectors, whereas data samples on the ε -tube border and outside the ε -tube are the support vectors but they have different values of the slack variables $\xi_i^{(*)}$ and dual variables $\alpha_i^{(*)}$, as summarized in Table 2.

Table 2: Values of Slack Variables and dual variables for different subsets

	Sample Location	SV	$\xi_i^{(*)}$	$\alpha_i^{(*)}$
Subset 1	Inside the ε -tube	Not SV	$\xi_i^{(*)}=0$	$\alpha_i^{(*)}=0$
Subset 2	On the ε -tube	Is SV	$\xi_i^{(*)}=0$	$\alpha_i^{(*)}\in(0,C)$
Subset 3	Outside the ε -tube	Is SV	$\xi_i^{(*)}>0$	$\alpha_i^{(*)}=C$

Recall that the coefficients $\boldsymbol{\omega}$ in the (linear) SVM solution (4) are calculated as

$$\boldsymbol{\omega} = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)\mathbf{x}_i \tag{8}$$

where non-zero contribution is provided only by support vectors, which are the data points in subset 2 (on the ε -tube) and subset 3 (outside the ε -tube). Further, the value of coefficient $\boldsymbol{\omega}$ is determined by $\alpha_i^{(*)}$ and $\mathbf{x}$ -value of training samples, however for samples in subset 3, the value $\alpha_i^{(*)} = C$ (constant) does not depend on the y -values of training samples. Hence, data points from subset 3 give the same contribution to SVM solution regardless of their y -values, i.e. independent of how far away their y -values are from the ε -tube. This property enables robust SVM estimation of the dominant model during multiple model estimation. For example in Fig.2, consider two '□' points, labeled as 'Point 1' and 'Point 2'. Although their y -values are quite different, their $\mathbf{x}$ -values are very close (or equal) and the corresponding $\alpha_i^{(*)} = C$, so their contributions to SVM solution (8) are (approximately) the same.

Similarly, one can analyze contribution of data samples to the bias term in SVM solution. Following [Smola and Schölkopf, 1998] the bias term (b) is given by:

$$b = y_i - < \boldsymbol{\omega}, \mathbf{x}_i > -\varepsilon \quad \text{for } \alpha_i \in (0, C) \tag{9}$$

$$b = y_i - < \boldsymbol{\omega}, \mathbf{x}_i > +\varepsilon \quad \text{for } \alpha_i^* \in (0, C)$$

where the constraint $\alpha_i^{(*)} \in (0, C)$ corresponds to data points in Subset 2 (on the border of ε -tube). Hence, the points outside the ε -tube (in subset 3) do not contribute to the bias, i.e. outliers (samples outside the ε -tube) have no affect on the value of bias.

In summary, our analysis of (linear) SVM regression presented above indicates that:

- SVM regression model depends mainly on SVs on the border of ε -insensitive zone;

- SVM regression solution is very robust to 'outliers' (i.e. data samples outside ε-insensitive zone). In particular, SVM solution does not depend on the y-values of such outliers.

These properties make SVM very attractive for its use in an iterative procedure for multiple model estimation described in Section 1, where a robust estimator applied to all training data needs to provide a reliable estimate of the first dominant model. The main practical issue is specifying conditions under which SVM regression would yield an accurate estimate of the dominant model, under multiple model setting. To answer this question, recall that an SVM model depends mainly on SVs on the border of ε-insensitive zone. Hence, SVM regression would provide an accurate estimate of the dominant model only if these SVs are generated by the dominant model. This will happen only if all (or most) samples in subset 1 (inside the ε-tube) and samples in subset 2 (on the ε-tube) are generated by the dominant model. Since the SVM model is estimated using all training data (from several models), the last condition implies that the majority of the data (say, over 55%) should be generated by dominant model. The requirement that majority of available data should be generated by a dominant model is standard in robust statistics [Lawrence and Arthur, 1987]. Here we simply derived this condition for SVM algorithm in the context of multiple model estimation.

3. SVM methodology for multiple model estimation

This section describes practical algorithms (based on SVM regression) for multiple model estimation. These algorithms are based on the iterative procedure described in Section 1. However, practical implementation of this procedure requires addressing the following issues:

- How to set meta-parameters of SVM regression;

- How to partition the data into two subsets (after the dominant model had been estimated).

In order to simplify presentation, all descriptions in this paper assume that training data are generated by two models, i.e. model M1 (dominant model) and model M2 (minor model). The goal is to accurately estimate the dominant model in the first iteration of an iterative procedure given in Section 1. Then the second model M2 is estimated in the second iteration of an algorithm. Generalization to data sets with multiple models is straightforward.

Selection of SVM meta-parameters. Next we discuss proper setting of ε (insensitive zone) and *C* (regularization parameter) in SVM regression for estimating the dominant model in Step 1 of an iterative procedure given in Section 1. There are many proposals for setting SVM meta-parameters for standard single-model estimation [Vapnik, 1995; Cherkassky and Mulier, 1998; Schölkopf et al, 1999; Hastie et al, 2001]. However, most theoretical prescriptions for setting meta-parameters are based on restrictive assumptions and in practice SVM meta-parameters are often selected via resampling [Schölkopf et al, 1999]. In this paper, however, we are interested in selecting meta-parameters for multiple model estimation setting. Recently, [Cherkassky and Ma, 2002] proposed analytical selection of SVM meta-parameters (for standard single-model regression formulation), as detailed next. For SVM regression, the values of meta-parameters are:

$$C = \max(|\ \bar{y} + 3\sigma_y\ |, |\ \bar{y} - 3\sigma_y\ |) \tag{10}$$

where $\bar{y}$ is the mean of the training response values, and σ_y is the standard deviation of the training response values;

$$\varepsilon(\sigma, n) = 3\sigma\sqrt{\frac{\ln n}{n}} \tag{11}$$

where σ is the standard deviation of additive noise and *n* is the number of training samples.

Further, it can be shown that the value of ε-parameter plays the most important role for SVM regression, whereas SVM solutions are rather insensitive to regularization parameter as long as this parameter is larger than the value given by (10) [Cherkassky and Ma, 2002]. This insensitivity to regularization parameter values is particularly true for linear SVM regression formulation (3). In other words, one can use very large value of regularization parameter in (3), so that SVM solution depends only on proper setting of ε. So in the remainder of the paper we shall be only concerned with selection of ε. In order to apply (11) for multiple model estimation, consider (for simplicity) only linear SVM. Then in order to estimate dominant model M1 we should know the standard deviation σ_1 of additive noise in the dominant model and the number of samples generated by the dominant model M1. Hence, we may consider two possibilities:

- First, we assume that the noise level for each (hidden) model is available or can be somehow estimated (using a priori knowledge). In this case, we simply use (11) for selecting the value of ε.
- Second, the noise level (standard deviation) and the number of samples for each model is not known.

Let us consider the second (more difficult) possibility. In this case, selection of ε relies on the requirement that majority of available is generated by the dominant model (being estimated by SVM). Hence, we need to select ε-value such that most of the data (say 55%) lies inside the ε-tube. This can be done by trial-and-error approach (i.e., trying different ε-values and examining support vectors in SVM estimates) or using a more systematic approach called ν-SVM [Schölkopf et al, 1998]. This approach effectively implements SVM regression having prespecified number of support vectors specified by parameter ν (i.e., a given fraction of the total number of samples). In the context of multiple model estimation, the requirement that 55% of the

data is inside the insensitive zone is equivalent to specifying ν =0.45, i.e. that 45% of the data is outside the ε-tube.

Remarkably, the ability of SVM to accurately estimate the dominant model is not very sensitive to the chosen width of ε-insensitive zone. For example, let us apply (linear) SVM to the data set shown in Fig.1, in order to estimate the dominant model M1. Assuming the noise standard deviation σ_1 =0.1 is known, the value of ε-insensitive zone according to (11) should be ε =0.084. This value has been used to generate regression estimates shown in Fig.1. In practice, we can only know/use crude estimates of the noise level (and hence crude ε-values). So we try to estimate the dominant model for data set in Fig. 1a using SVM regression with three different ε-values (ε =0.084, 0.042 and 0.126). These values are chosen as a half and one-and-a-half of the value ε =0.084 specified by (11). Fig. 3 shows SVM estimates of the dominant model for different ε-values; clearly these estimates are almost identical, in spite of significant variations in ε-values. Hence, using inaccurate values of σ_1 and n_1 (the number of samples) for estimating the value ε of via (11) should not affect accurate estimation of the dominant model. For example, if the total number of samples is 100 (known number), then the (unknown) number of samples in the dominant model should be at least 50. According to (11), the difference between $\sqrt{\ln 50/50}$ and $\sqrt{\ln 100/100}$ is about 25%, so using inaccurate values of the number of samples should result in 25% variation in ε-values (in the worst case). This variation would not affect the accuracy of SVM regression estimates, as indicated by empirical results in Fig.3.

Data partitioning step. Following estimation of the dominant model, we need to partition available data into two subsets, i.e. data generated by dominant model and the remaining data (generated by other models). This is done by analyzing the (absolute value of) residuals between the training response values y_i and SVM estimates provided by dominant model $\hat{y}(\mathbf{x}_i)$:

$$\left|res_i\right| = | y_i - \hat{y}(\mathbf{x}_i) | \quad \text{for } i = 1,..., n \tag{12}$$

Namely, training samples with residual values smaller than certain threshold are generated by dominant model, and samples with large absolute values of residuals are generated by other model(s). Empirically we found that a good threshold equals twice the standard deviation of additive noise in the dominant model M1:

$$\text{If } \left|res_i\right| < 2\sigma \text{ then } (\mathbf{x}_i, y_i) \in M1 \tag{13}$$

Here we assume that the noise level σ is known a priori or can be (accurately) estimated from data. In fact, the noise level (its standard deviation) can be readily estimated from data as outlined next. Let us plot the histogram of residuals $res_i = y_i - \hat{y}(\mathbf{x}_i)$ for training data. Since SVM provides very accurate estimates of the dominant model (in Step 1) and the majority of the data are produced by the dominant model, these samples will form a large cluster (of residuals) symmetric around zero, whereas samples generated by other models will produce a few more smaller clusters. Then the standard deviation of noise (in the dominant model) can be estimated via standard deviation of residuals in a large cluster. Further, empirical results (in Fig.3) indicate that overall quality of multiple model estimation procedure is not sensitive to accurate knowledge/estimation of the noise level.

4. Empirical results

This section describes empirical results for multiple model estimation procedure using synthetic data sets. We only show examples where training samples $Z = (\mathbf{x}_i, y_i), i = 1,2,...,n$ are generated by two models, i.e.

- Model M1 generates n_1 samples according to $y = r_1(\mathbf{x}) + \delta_1$

- Model M2 generates n_2 samples according to $y = r_2(\mathbf{x}) + \delta_2$, so that $n_1 + n_2 = n$.

Note that the same algorithm can been successfully applied when the data are generated by larger number of models (these results are not shown here due to space constraints).

In all examples the input values of the training data are generated as random samples from a uniform distribution. Both hidden models are defined in the same domain in the input space. We use additive gaussian noise to generate training data for the examples presented in this section. However, the proposed method works well with other types of noise; also the standard deviation (of noise) may be different for different models. To simplify the presentation, the standard deviation of noise is assumed to be known (to the algorithm); however in practical settings the noise level can be estimated from data, as described in section 3.

The first example assumes that both models are linear, hence we apply linear SVM regression method (without kernels). In this example we have 100 training samples generated as follows:

- Model M1: $y = r_1(x) + \delta_1$, where $r_1(x) = 0.8x + 2$, $x \in [0,1]$, $n_1 = 60$ (major model).

- Model M2: $y = r_2(x) + \delta_2$, where $r_2(x) = 0.2x + 1$, $x \in [0,1]$, $n_2 = 40$ (minor model).

We consider two noise levels: $\sigma_1 = \sigma_2 = 0.1$ (small noise) and $\sigma_1 = \sigma_2 = 0.3$ (large noise). The training data sets (with small noise and with large noise) are shown in Fig. 4, with samples generated by major model labeled as '+', and samples generated by minor model labeled as '□'. These data are hard to separate visually (by a human eye) in the case of large noise. However, the proposed method can accurately estimate both models, and separate the training data, as shown in Fig.4. As expected, the model estimation accuracy is better for low noise; however even with large noise the model estimates are quite good.

In second example, both models are nonlinear. Hence we use nonlinear SVM regression. An RBF kernel function (6) with width parameter p=0.2 is used in this example. Available training data (total of 100 samples) are generated as follows:

- Model M1: $y = r_1(x) + \delta_1$, where $r_1(x) = \sin(2\pi x)$, $x \in [0,1]$, n_1=70

- Model M2: $y = r_2(x) + \delta_2$, where $r_2(x) = \cos(2\pi x)$, $x \in [0,1]$, $n_2 = 30$.

Again, we consider two noise levels: $\sigma_1 = \sigma_2$ =0.1 (small noise) and $\sigma_1 = \sigma_2$ =0.3 (large noise). The training data sets (with small noise and with large noise) are shown in Fig. 5. Results in Fig. 5 indicate that the proposed method provides very accurate model estimates, even in the case of large noise.

Finally, we show an example of multiple model estimation for higher-dimensional data. We consider linear models in a 4-dimensional input space, so that available training data are generated as follows:

- Model M1: $y = r_1(\mathbf{x}) + \delta_1$, where $r_1(\mathbf{x}) = x_1 + x_2 + x_3 + x_4$, $\mathbf{x} \in [0,1]^4$, $n_1 = 60$ (major model).

- Model M2: $y = r_2(\mathbf{x}) + \delta_2$, where $r_2(\mathbf{x}) = 6 - x_2 - x_3 - x_4$, $\mathbf{x} \in [0,1]^4$, $n_2 = 40$ (minor model).

Training data are corrupted by additive noise with standard deviation $\sigma_1 = \sigma_2$ =0.1.

For this data set, we illustrate the data partitioning step in Fig.6. Results in Fig. 6 show the distribution of residual values, i.e. the difference between response values and M1 model estimates (normalized by standard deviation of noise) calculated according to (12). Residual values for the first 60 samples (generated by model M1) are on the left-hand side of Fig. 6, and the next 40 samples (generated by model M2) are on the right-hand side of Fig.6. Partitioning of data samples is performed using residual values according to (13) using threshold value 2, and

this threshold is indicated as a horizontal line in the middle of Fig.6. That is, samples below this line are assumed to originate from M1, and samples above this line are assumed to originate from M2. As expected, data partitioning is not very accurate, since some samples from M1 are classified as samples from M2, and vice versa. This is because the two models actually provide the same (or very close) response values in a small region of the input space; so perfectly accurate classification is not possible. However, the proposed multiple model estimation procedure provides very accurate model estimates for this data set. Namely, the estimates obtained for models M1 and M2 are:

For model M1: $\hat{y}_1(\mathbf{x}) = 0.01 + 1.14x_1 + 0.95x_2 + 1.08x_3 + 1.08x_4$ MSE (for M1) = 0.0277;

For model M2: $\hat{y}_2(\mathbf{x}) = 5.93 - 0.07x_1 - 0.93x_2 - x_3 - 0.99x_4$ MSE (for M2) = 0.0044.

Clearly, the above estimates are very close to the target functions used to generate noisy data. The MSE measure indicates the mean-squared-error between regression estimates and the true target function (for each model) obtained using 500 independently generated test samples.

5. Clustering using multiple model estimation

In many applications, the goal of data modeling (assuming that data are generated by several models) is to cluster/partition available data into several subsets, corresponding to different generating models. This goal is concerned mainly with accurate partitioning of the data, rather than with accurate estimation of the (hidden) regression models, even though these two objectives are highly correlated. Example shown in Fig. 6 illustrates the problem: even though data partitioning (implemented by proposed algorithm) is not very accurate, the algorithm produces very accurate and robust estimates of regression models. In this section we show how to improve the accuracy of data partitioning under multiple model estimation formulation.

For example, consider nonlinear data set described in Section 4 and shown in Fig. 5c. For this data set, some samples are very difficult to assign to an appropriate model, especially in regions of the input space where the models have similar response values. For this data set, the proposed multiple model estimation algorithm correctly classifies 90.4% of samples generated by major model M1, and 53.5% of samples generated by minor model M2. However, the accuracy of data partitioning can be further improved using simple post processing procedure described next. This procedure uses regression estimates provided by proposed multiple model estimation algorithm. Let us denote regression estimate for major model M1 as $\hat{y}^{(1)}(\mathbf{x})$, and regression estimate for minor model M2 as $\hat{y}^{(2)}(\mathbf{x})$. Then each training sample $(\mathbf{x}_i, y_i), i = 1,2,...,n$ can be assigned to one of the two models based on the (absolute) value of residuals:

$$\left|res_i^{(1)}\right| = | y_i - \hat{y}_i^{(1)}(\mathbf{x}_i) | \qquad \text{and} \qquad \left|res_i^{(2)}\right| = | y_i - \hat{y}_i^{(2)}(\mathbf{x}_i) |$$

That is:

$$\text{If } \left|res_i^{(1)}\right| < \left|res_i^{(2)}\right| \text{ then } (\mathbf{x}_i, y_i) \in M1 \text{ else } (\mathbf{x}_i, y_i) \in M2 \tag{14}$$

Effectively, such post processing method implements 'nearest neighbor' classification using (absolute value of) residuals. Applying prescription (14) for partitioning the data set shown in Fig. 5c yields classification accuracy 92.6% for samples generated by M1, and classification accuracy 80% for samples generated by M2. Hence, data re-partitioning technique (14) gives better accuracy than data partitioning produced by the original multiple model estimation procedure.

In conclusion, we comment on applicability and implications of clustering/data partitioning approach described in this section. This approach to clustering assumes that training data are generated by several models, and the clustering relies heavily of accurate estimates of (regression) models obtained by robust SVM-based algorithm. Hence, the problem setting itself

combines supervised learning (i.e. estimation of regression models) and unsupervised learning (i.e. data partitioning or clustering). We expect this approach to clustering to outperform traditional clustering techniques for applications that can be described using multiple model formulation. Finally, proposed 'nearest neighbor' rule (14) for data partitioning assumes that both (hidden) models have the same noise level (or standard deviation), and the same 'misclassification' cost for both models. These assumptions hold true for the data set in Fig. 5c, and this explains improved classification accuracy for this example. In many applications, however, the noise levels and misclassification costs for different (hidden) models are not the same, and one should adjust the rule (14) to account for these differences.

6. Summary and discussion

This paper presents a new algorithm for multiple model estimation. The proposed method is based on SVM learning adapted to multiple model formulation. Empirical results presented in this paper demonstrate that SVM-based learning can be successfully applied to multiple model regression problems. In addition, we introduced a new clustering/data partitioning method suitable for multiple model formulation.

Future related work may focus on applications of the proposed methodology to real-world problems, ranging from computer vision (motion analysis) to financial engineering. As discussed in [Cherkassky and Ma, 2002], such applications should be based on a thorough understanding of each application domain, necessary for a meaningful specification/parameterization of (hidden) models. Additional research may be concerned with better understanding of robustness of SVM methodology, and comparing it with traditional robust methods in the context of multiple model estimation.

Finally, we point out that the proposed learning method assumes that all (hidden) models are defined on the same domain (i.e., the same region in the input space). In situations where different models are defined in different (disjoint) regions of the input space, the proposed algorithm can not be successfully applied. Instead, one should use well-known (tree) partitioning algorithms such as CART, mixture of experts and their variants [Hastie et al, 2001]. These algorithms effectively partition the input space into several (disjoint) regions and estimate output (response) values in each region of the input space. Here it may be interesting to note that tree partitioning algorithms are based on a single model formulation, so they tend to enforce smoothness at the region boundaries. It may be possible to develop learning algorithms for regression in disjoint regions using multiple model formulation, and then compare its accuracy with traditional tree partitioning methods.

REFRENCES

[1] V. Cherkassky & Y. Ma, "Multiple Model Estimation: A New Formulation for Predictive Learning", IEEE Trans. Neural Networks (under review), 2002

[2] M. Tanaka, "Mixture of Probabilistic Factor Analysis Model and Its Application", in *Proc. ICANN 2001*, LNCS 2130, pp. 183-188, 2001

[3] V. Vapnik. *The Nature of Statistical Learning Theory (2nd ed.)*. Springer, 1999

[4] V. Cherkassky & F. Mulier, *Learning from Data: Concepts, Theory and Methods*, Wiley1998

[5] H. Chen, P. Meer and D. Tyler, "Robust Regression for Data with Multiple Structures", in *CVPR 2001, Proc. IEEE Computer Society Conf.,* pp 1069-1075, 2001

[6] P. Rousseeuw and A. Leroy, *Robust Regression and Outlier Detection*, Wiley, NY, 1987

[7] V. Vapnik, *The Nature of Statistical Learning Theory,* Springer, 1995

[8] K. Lawrence and J. Arthur, *Robust Regression- analysis and applications*, New York : M. Dekker, 1990

[9] A. Smola and B. Schölkopf, "A Tutorial on Support Vector Regression", NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London, UK, 1998

[10] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, Springer, 2001

[11] B. Schölkopf, J. Burges and A. Smola, ed., "*Advances in Kernel Methods: Support Vector Machines*", MIT Press, 1999

[12] V. Cherkassky and Y. Ma, "Selection of Meta-Parameters for Support Vector Regression", *Proc. ICANN-2002* (to appear), 2002

[13] B. Schölkopf, P. Bartlett, A. Smola, and R. Williamson, "Support Vector regression with automatic accuracy control", in L. Niklasson, M. Bodén, and T. Ziemke (Eds), *Proc. ICANN'98*, Springer, pp 111-116, 1998

FIGURE CAPTIONS

Fig. 1: Comparing robust method vs least squares estimation of the dominant model.
(a) First data set (dominant model 70% of data samples, secondary model 30% of samples),
(b) Estimates of dominant model M1 by robust method vs. least squares, for data set (a),
(c) Second data set (dominant model 70% of data samples, secondary model 30% of samples),
(d) Estimates of dominant model M1 by robust method vs. least squares, for data set (b).

Fig. 2: Location of training data with respect to ε-insensitive tube showing three possible subsets of data.

Fig. 3. SVM estimates of dominant model do not depend on accurate selection of ε-values. Results show three SVM estimates for data set in Fig. 1(a), using 'optimal' ε (for this data set), 0.5ε, and 1.5ε:

Solid line --- 'optimal' ε=0.084
Dashed line --- ε=0.042
Dotted line --- ε=0.126

Fig. 4. Example of multiple model estimation procedure for linear models:
(a) Training data (with small noise)
(b) Model estimates for data set (a) obtained using proposed algorithm
(c) Training data (with large noise)
(d) Model estimates for data set (b) obtained using proposed algorithm.

Fig. 5. Example of multiple model estimation procedure for nonlinear models:

(a) Training data (with small noise)

(b) Model estimates for data set (a) obtained using proposed algorithm

(c) Training data (with large noise)

(d) Model estimates for data set (b) obtained using proposed algorithm.

Fig.6. Illustration of data partitioning Step 2 in the proposed algorithm for high-dimensional data set. Horizontal threshold line is used to partition the data into two subsets.

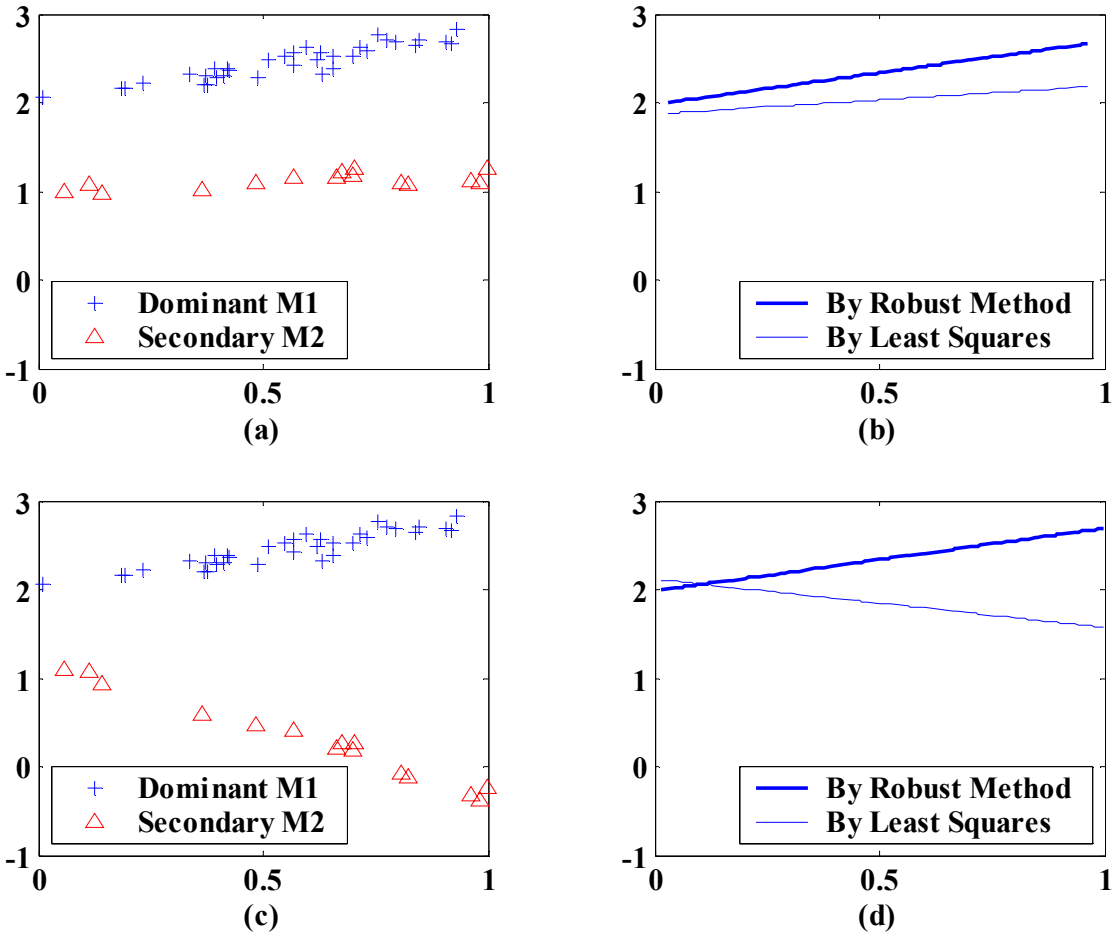

Fig.1. Comparing robust method vs least squares estimation of the dominant model.

(a) First data set (dominant model 70% of data samples, secondary model 30% of samples),

(b) Estimates of dominant model M1 by robust method vs. least squares, for data set (a),

(c) Second data set (dominant model 70% of data samples, secondary model 30% of samples),

(d) Estimates of dominant model M1 by robust method vs. least squares, for data set (b).

[Cherkassy and Ma]

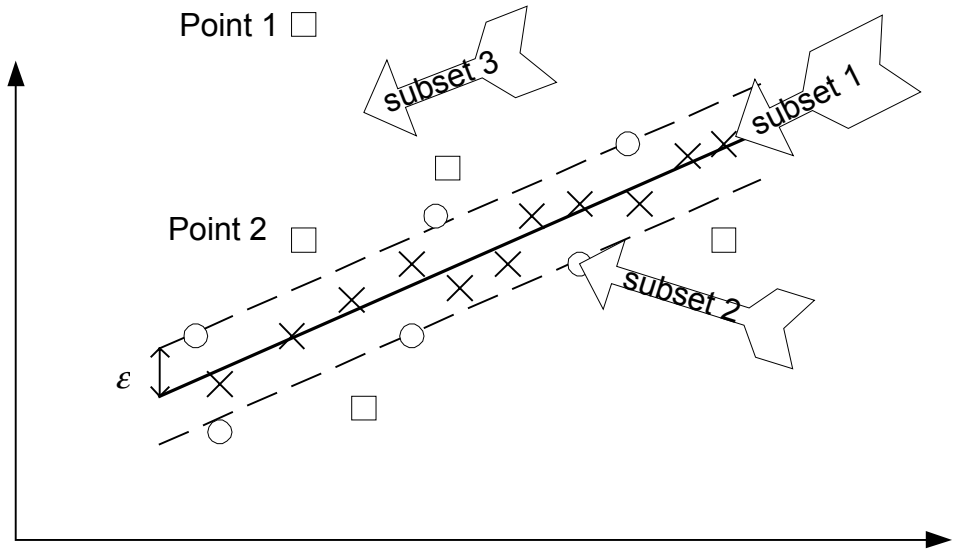

Fig. 2: Location of training data with respect to ε-insensitive tube showing three possible subsets of data.

[Cherkassy and Ma]

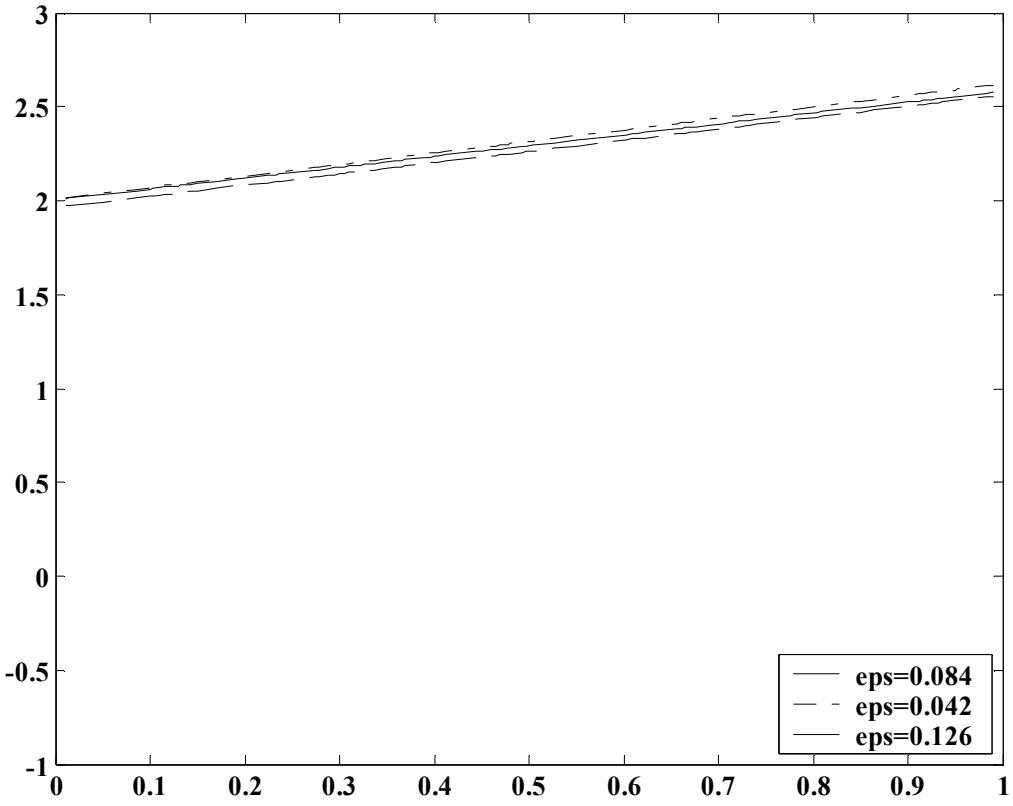

Fig. 3. SVM estimates of dominant model do not depend on accurate selection of ε-values. Results show three SVM estimates for data set in Fig. 1(a), using 'optimal' ε (for this data set), 0.5ε, and 1.5ε:

Solid line --- 'optimal' ε=0.084

Dashed line --- ε=0.042

Dotted line --- ε=0.126

[Cherkassy and Ma]

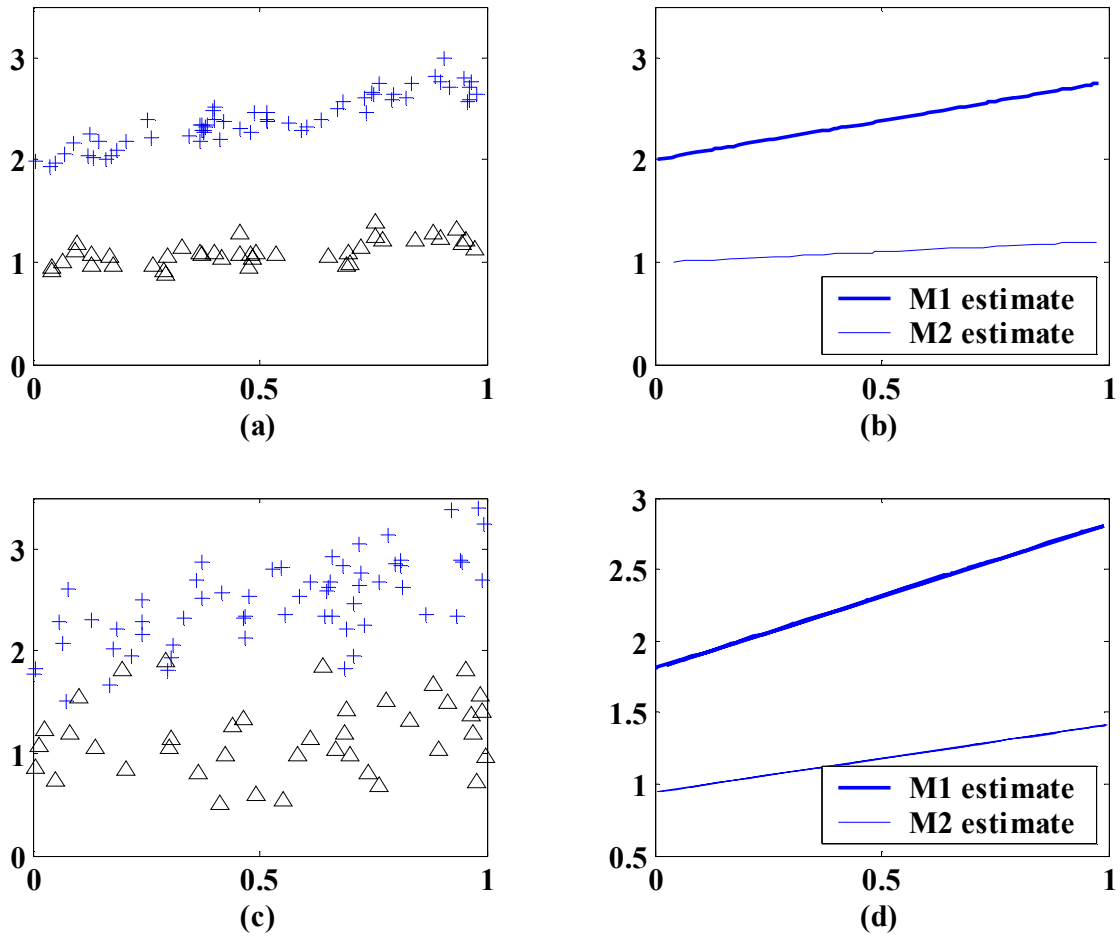

Fig. 4. Example of multiple model estimation procedure for linear models:

(a) Training data (with small noise)

(b) Model estimates for data set (a) obtained using proposed algorithm

(c) Training data (with large noise)

(d) Model estimates for data set (b) obtained using proposed algorithm.

[Cherkassy and Ma]

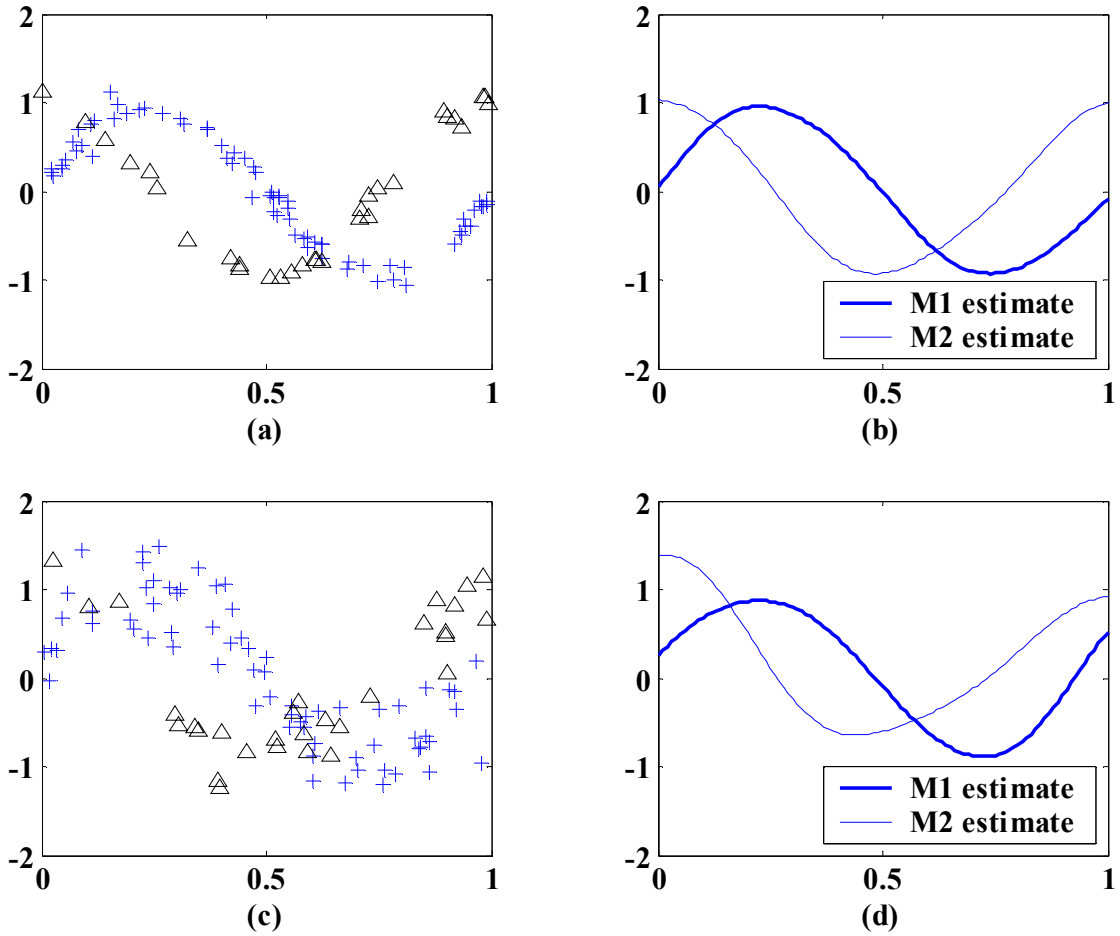

Fig. 5. Example of multiple model estimation procedure for nonlinear models:

(a) Training data (with small noise)

(b) Model estimates for data set (a) obtained using proposed algorithm

(c) Training data (with large noise)

(d) Model estimates for data set (b) obtained using proposed algorithm.

[Cherkassy and Ma]

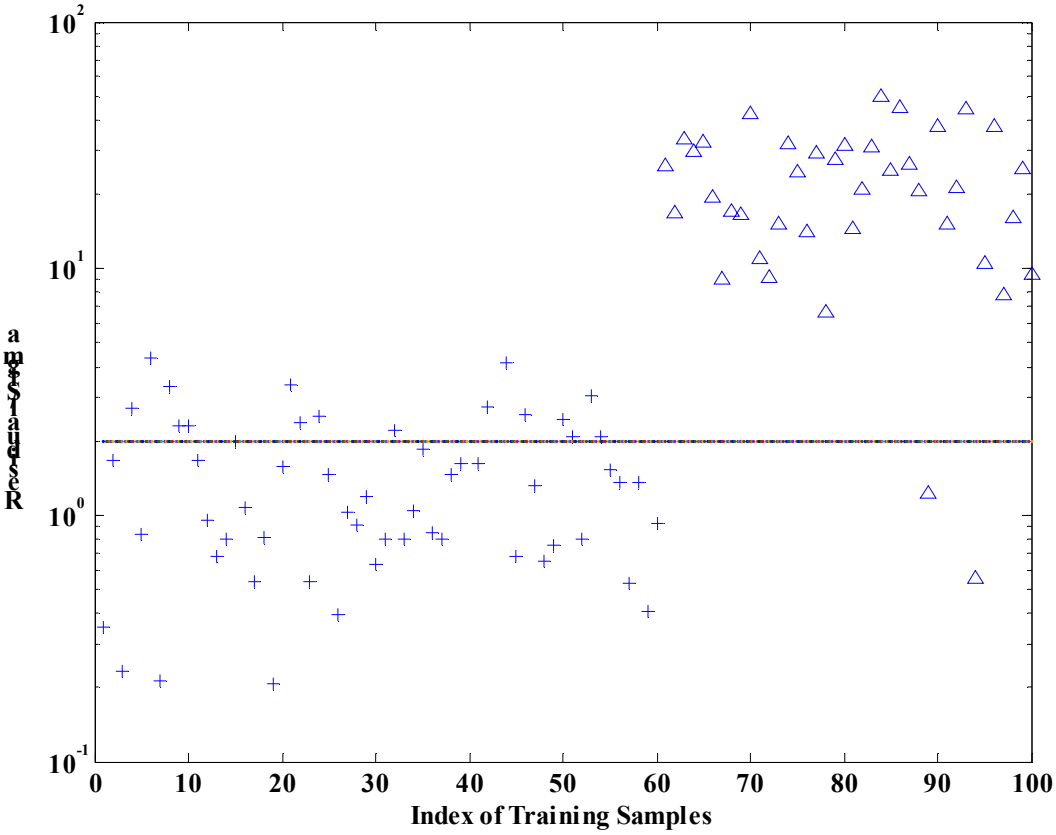

Fig.6. Illustration of data partitioning Step 2 in the proposed algorithm for high-dimensional data set. Horizontal threshold line is used to partition the data into two subsets.

[Cherkassy and Ma]