

# Robust Minimax Probability Machine Regression

**Thomas R. Strohmann**

**Gregory Z. Grudic**

*Department of Computer Science*

*University of Colorado*

*Boulder, CO 80309-0430, USA*

STROHMAN@CS.COLORADO.EDU

GRUDIC@CS.COLORADO.EDU

## Abstract

We formulate regression as maximizing the minimum probability ( $\Omega$ ) that the regression model is within  $\pm\epsilon$  of all future observations (i.e. outputs) of the true regression function. Our framework starts by posing regression as a *binary* classification problem, such that a solution to this single classification problem directly solves the original regression problem. Minimax probability machine classification (Lanckriet et al., 2002a) is used to solve the binary classification problem, resulting in a direct bound on the minimum probability  $\Omega$  that the regression model is within  $\pm\epsilon$  accurate. This minimax probability machine regression (MPMR) model assumes only that the mean and covariance matrix of the distribution which generated the regression data are known; no further assumptions on conditional distributions are required. Theory is formulated for determining when estimates of mean and covariance are accurate, thus implying robust estimates of the probability bound  $\Omega$ . Conditions under which the MPMR regression surface is identical to a standard least squares regression surface are given, allowing direct generalization bounds to be easily calculated for any least squares regression model (which was previously possible only under very specific, often unrealistic, distributional assumptions). We further generalize these theoretical bounds to any superposition of basis functions regression model. Experimental evidence is given supporting these theoretical results.

**Keywords:** Minimax Probability Machine, Regression, Robust, Kernel Methods, Bounds, Distribution free

## 1. Introduction

We formulate the regression problem as one of maximizing the probability, denoted by  $\Omega$ , that the predicted output will be within some margin  $\epsilon$  of all future observations of the true regression function.<sup>1</sup> We show how to compute a direct estimate of this probability for any given margin  $\epsilon > 0$ . Following the idea of the minimax probability machine for classification (MPMC) due to Lanckriet et al. (2002a), we make no detailed distributional assumptions, but obtain a probability  $\Omega$  that is a lower bound for all possible distributions with a known mean and covariance matrix. In other words, we maximize the minimum probability of our regression model being within  $\pm\epsilon$  correct on test data for all possible distributions that have

---

1. Throughout this paper, the phenomenon that generated the training data is referred to as the true regression function (or surface), while the model that is constructed from the training data is referred to as the regression model.

the same mean and covariance matrix as the distribution that generated the training data. We term this type of regression model as minimax probability machine regression (MPMR) (see Strohmann and Grudic, 2003).

Current practice for estimating how good a regression model is dictates that one has to either estimate the underlying distribution of the data or make specific distributional assumptions (e.g. Gaussian), both of which have their problems. Estimating high dimensional distributions from small samples is a very difficult problem that has not been solved to satisfaction (see S. J. Raudys, 1991). Gaussian processes obtain regression models within a Bayesian framework (see Williams and Rasmussen, 1995, Rasmussen, 1996, Gibbs, 1997a, Mackay, 1997). As the name indicates, they put a prior on the function space that is a generalization of a Gaussian distribution for (a finite number of) random variables. Instead of mean and covariance, Gaussian processes use a mean function and a covariance function to express priors on the function space. By adopting these restrictions it is possible to obtain confidence intervals for a given regression estimate. In practice, however, the assumption that data is generated from underlying Gaussians does not always hold; and the Gaussian process confidence intervals can become unrealistic. Similar problems with unrealistic bounds occur in such algorithms as the relevance vector machine (Tipping, 2000) where approximations, which may be unrealistic, are made in obtaining these. In contrast—as long as one can obtain accurate estimates of mean and covariance matrix—the distribution-free algorithm proposed here will always yield a correct lower bound estimate, no matter what the underlying distribution is.

Because the accuracy of MPMR directly depends on the accuracy of mean and covariance estimates, we propose a theoretical framework for determining when these estimates are accurate. This in turn determines when the estimate of the bound  $\Omega$  is accurate. The general idea is to randomly discard a (small) subset of the training data and measure how sensitive the estimate of  $\Omega$  is when we use the remaining data to estimate mean and covariance matrix.

To solve the regression problem, we reduce it to a binary classification problem by generating two classes that are obtained by shifting the dependent variable  $\pm\epsilon$  (see Figure 1). The regression surface is then interpreted as being the boundary that separates the two classes. Although any classifier can be used to solve the regression problem in this way, we propose to use MPMC (Lanckriet et al., 2002a), which gives a bound  $\alpha$  for the classification accuracy that can in turn be used to compute the bound  $\Omega$  for regression. One result of this paper is a theorem stating that when MPMC is used to solve the regression problem as described above, the regression model is identical to the one obtained by standard least squares regression. The importance of this result is that it is now straightforward to calculate a generalization bound for the least squares method. This was previously only possible under strict, often unrealistic, distributional assumptions.

The paper is organized as follows: in section 2 we give a strict mathematical definition of our approach to the regression problem and describe our hypothesis space of functions for linear and nonlinear regression surfaces. Section 3 outlines the statistical foundation of the minimax probability machine, based on a theorem due to Marshall and Olkin (1960) that was later extended by Popescu and Bertsimas (2001). We then show how any regression problem can be formulated as a binary classification problem by creating two symmetric classes (see Figure 1). Following this scheme we conclude the section by deriving the linear

MPMR from the linear MPMC and showing that, due to the symmetry, it is computationally easier to find an MPM regression model than it is to find an MPM classifier. Section 3 also presents an analysis on the tightness of the MPMR bound. In section 4 we describe how to obtain nonlinear regression surfaces by using any set of nonlinear basis functions, and then focus on using nonlinear kernel maps as a distance metric on pairs of input vectors. The problem of overfitting the training data with highly sensitive nonlinear kernel maps is addressed in section 5. There we propose a sensitivity measure  $S$  that reflects how sensitive our estimate of  $\Omega$  is with respect to the estimates of mean and covariance (which are the only estimates MPMR relies upon). We give empirical evidence that the statistical assumptions  $S$  relies upon are valid and that  $S$  works well for detecting overfitting and doing model selection. Section 6 contains the experiments we conducted with our learning algorithm on toy problems and real world data sets. We start out with a simple toy example and show that for certain distributions the Gaussian process error estimates fail, while the ones obtained by MPMR still hold. For the real world data sets we focused on investigating the accuracy of  $\Omega$  as a lower bound probability on  $\pm\epsilon$ -accurate predictions and its relation to our sensitivity measure  $S$ . The last section summarizes the main contributions of this paper and points out some open research questions concerning the MPMR framework.

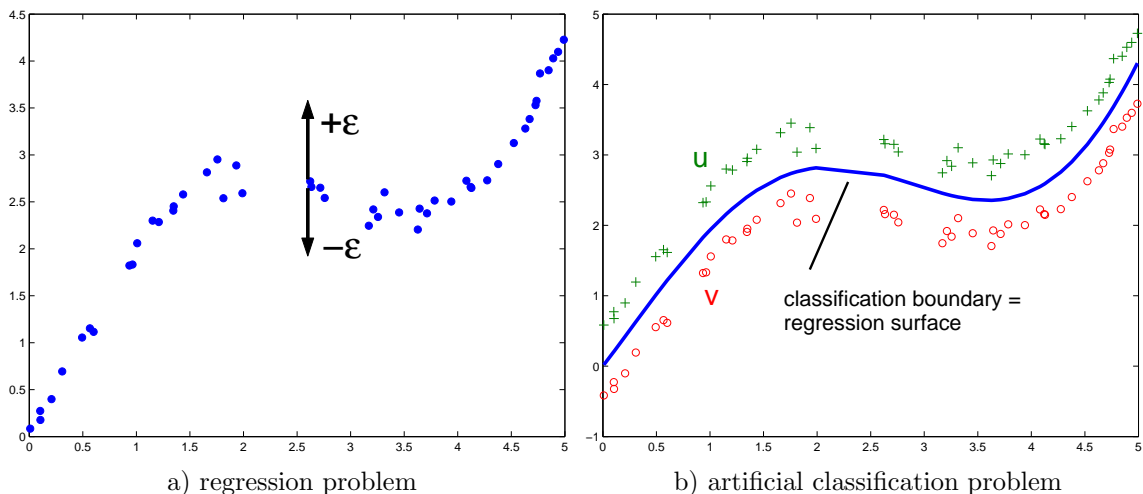


Figure 1: Formulating regression as binary classification. Each regression data point turns into two training examples — one positive and one negative — for classification. The positive examples have the output variable shifted by  $+\epsilon$  and the negative examples have it shifted by  $-\epsilon$ .

A Matlab implementation of the MPMR algorithm can be obtained from:  
<http://csel.cs.colorado.edu/~strohman/papers.html>

## 2. Regression Model

Formally, MPMR addresses the following problem: Let  $f^* : \mathbf{R}^d \rightarrow \mathbf{R}$  be some unknown regression function (i.e., the phenomenon that generated the learning data), let the random

vector  $\mathbf{x} \in \mathbf{R}^d$  be generated from some bounded distribution that has mean  $\bar{\mathbf{x}}$  and covariance  $\Sigma_{\mathbf{x}}$ —which we will write from now on as  $\mathbf{x} \sim (\bar{\mathbf{x}}, \Sigma_{\mathbf{x}})$ —and let our training data  $\Gamma = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$  be generated according to:

$$y = f^*(\mathbf{x}) + \rho$$

where  $\rho$  is a noise term with an expected value of  $E[\rho] = \bar{\rho} = 0$  and some finite variance  $Var[\rho] < \infty$ . Given a hypothesis space  $\mathcal{H}$  of functions from  $\mathbf{R}^d$  to  $\mathbf{R}$ , we want to find a model  $\hat{f} \in \mathcal{H}$  that maximizes the minimum probability of being  $\pm\epsilon$  accurate symbolized by  $\Omega_{\hat{f}}$ . We define:

$$\Omega_f = \inf_{(\mathbf{x}, y) \sim (\bar{\mathbf{x}}, \bar{y}, \Sigma)} Pr\{|f(\mathbf{x}) - y| < \epsilon\} \quad (1)$$

where

$$\Sigma = cov \left( y, x_1, x_2, \dots, x_d \right)$$

In other words,  $\Sigma \in \mathbf{R}^{(d+1) \times (d+1)}$  is the complete covariance matrix of the random vector  $\mathbf{x}$  and the random variable  $y = f^*(\mathbf{x}) + \rho$  (see equation (13) for details). The only assumptions we make about the first and second moments of the underlying distribution is that  $\bar{\mathbf{x}}, \bar{y}$  and  $\Sigma$  are finite. *No other distributional assumptions are made.*

For any function  $f \in \mathcal{H}$ , the model  $\hat{f}$  that we are looking for has to satisfy:

$$\Omega_{\hat{f}} \geq \Omega_f \quad \text{for all } f \in \mathcal{H}$$

For linear models,  $\mathcal{H}$  contains all the functions that are linear combinations of the input vector:

$$f(\mathbf{x}) = \sum_{j=1}^d \beta_j x_j + \beta_0 = \beta^T \mathbf{x} + \beta_0 \quad (\text{where } \beta \in \mathbf{R}^d, \beta_0 \in \mathbf{R}) \quad (2)$$

For nonlinear models we consider the general basis function formulation:

$$f(\mathbf{x}) = \sum_{i=1}^k \beta_i \Phi_i(\mathbf{x}) + \beta_0 \quad (3)$$

In this paper we focus on the commonly used kernel representation  $K : \mathbf{R}^d \times \mathbf{R}^d \rightarrow \mathbf{R}$  (see for example Smola and Schölkopf, 1998) where the hypothesis space consists of all linear combinations of kernel functions with the training inputs as their first arguments (i.e.  $\Phi_i(\mathbf{x}) = K(\mathbf{x}_i, \mathbf{x})$ ):

$$f(\mathbf{x}) = \sum_{i=1}^N \beta_i K(\mathbf{x}_i, \mathbf{x}) + \beta_0$$

where  $\mathbf{x}_i \in \mathbf{R}^d$  denotes the  $i$ th input vector of the training set  $\Gamma = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ . In the nonlinear formulation we consider the distribution of the random vector of basis functions  $\Phi = (\Phi_1, \Phi_2, \dots, \Phi_N)$  to formulate the MPMR bound  $\Omega$ :

$$\Omega_f^{\Phi} = \inf_{(\Phi, y) \sim (\bar{\Phi}, \bar{y}, \Sigma_{\Phi})} Pr\{|f(\mathbf{x}) - y| < \epsilon\}$$

where

$$\Sigma_{\Phi} = cov \left( y, \Phi_1, \Phi_2, \dots, \Phi_N \right)$$

See equation (13) for details on calculating  $\Sigma_{\Phi}$ .

### 3. Linear Framework

In this section we will develop the linear MPMR algorithm by reducing the regression problem to a single (specially structured) binary classification problem. We will also show that the regression estimator of the linear MPMR is equivalent to the one obtained by the method of least squares. In the following subsection we briefly review MPM classification.

#### 3.1 Minimax Probability Machine Classification (MPMC)

The minimax binary classifier for linear decision boundaries can be formulated as a hyperplane that maximizes the minimum probability of correctly classifying data points generated from two different random vectors (Lanckriet et al., 2002a,b). It assumes that we have a random vector  $\mathbf{u}$  (the positive examples), a random vector  $\mathbf{v}$  (the negative examples) and that  $\mathbf{u}$  and  $\mathbf{v}$  are drawn from two probability distributions characterized by  $(\bar{\mathbf{u}}, \Sigma_{\mathbf{u}})$  and  $(\bar{\mathbf{v}}, \Sigma_{\mathbf{v}})$ . The linear MPMC algorithm then solves the following optimization problem:

$$\max_{\alpha, \mathbf{a} \neq \mathbf{0}, b} \alpha \quad s.t. \quad \inf_{\mathbf{u} \sim (\bar{\mathbf{u}}, \Sigma_{\mathbf{u}})} Pr\{\mathbf{a}^T \mathbf{u} \geq b\} \geq \alpha \wedge \inf_{\mathbf{v} \sim (\bar{\mathbf{v}}, \Sigma_{\mathbf{v}})} Pr\{\mathbf{a}^T \mathbf{v} \leq b\} \geq \alpha \quad (4)$$

A theorem due to Marshall and Olkin (1960) that was later extended by Popescu and Bertsimas (2001) gives a formula for the supremum probability that a random vector lies on one side of a hyperplane:

$$\sup_{\mathbf{v} \sim (\bar{\mathbf{v}}, \Sigma_{\mathbf{v}})} Pr\{\mathbf{a}^T \mathbf{v} \geq b\} = \frac{1}{1 + \delta^2}, \quad \text{with } \delta^2 = \inf_{\mathbf{a}^T \mathbf{w} \geq b} (\mathbf{w} - \bar{\mathbf{v}})^T \Sigma_{\mathbf{v}}^{-1} (\mathbf{w} - \bar{\mathbf{v}}) \quad (5)$$

Note that the infimum on the right side can be computed analytically from the hyperplane parameters  $\mathbf{a}$  and  $b$  and the estimates  $\bar{\mathbf{v}}, \Sigma_{\mathbf{v}}$  (Lanckriet et al., 2002b). This result can be used to simplify the optimization problem (4) to the following one:

$$m = \left( \min_{\mathbf{a}} \sqrt{\mathbf{a}^T \Sigma_{\mathbf{u}} \mathbf{a}} + \sqrt{\mathbf{a}^T \Sigma_{\mathbf{v}} \mathbf{a}} \right) \quad s.t. \quad \mathbf{a}^T (\bar{\mathbf{u}} - \bar{\mathbf{v}}) = 1 \quad (6)$$

The offset  $b$  of the hyperplane is uniquely determined for any  $\mathbf{a}$ :

$$b = \mathbf{a}^T \bar{\mathbf{u}} - \frac{\sqrt{\mathbf{a}^T \Sigma_{\mathbf{u}} \mathbf{a}}}{m} = \mathbf{a}^T \bar{\mathbf{v}} + \frac{\sqrt{\mathbf{a}^T \Sigma_{\mathbf{v}} \mathbf{a}}}{m} \quad (7)$$

Once a minimum value  $m$  is obtained for (6), the probability bound  $\alpha$  can be computed as:

$$\alpha = \frac{1}{1 + m^2}$$

#### 3.2 Formulating Regression as Classification

In a linear regression model we want to approximate  $f^* : \mathbf{R}^d \rightarrow \mathbf{R}$  with a linear estimator, i.e. a function of the form:

$$\hat{y} = \hat{f}(\mathbf{x}) = \beta^T \mathbf{x} + \beta_0$$

To obtain a minimax regression model—i.e. one which maximizes the minimum probability that future points are predicted within  $\pm\epsilon$  correctly—we use the MPMC framework in

the following way: each training data point  $(\mathbf{x}_i, y_i)$  for  $i = 1, \dots, N$  turns into two  $d + 1$  dimensional vectors, one is labelled as class  $\mathbf{u}$  (and has the  $y$  value shifted by  $+\epsilon$ ). The other one is labelled as class  $\mathbf{v}$  (and has the  $y$  value shifted by  $-\epsilon$ , see also Figure 1):

$$\begin{aligned} \mathbf{u}_i &= (y_i + \epsilon, x_{i1}, x_{i2}, \dots, x_{id}) & i = 1, \dots, N \\ \mathbf{v}_i &= (y_i - \epsilon, x_{i1}, x_{i2}, \dots, x_{id}) & i = 1, \dots, N \end{aligned} \tag{8}$$

It is interesting to note that we could use any binary classification algorithm to solve this artificial classification problem. The boundary obtained by the classifier turns directly into the regression surface one wants to estimate. For example one could even use decision trees as underlying classifier, determine where the decision tree changes its classification, and use this boundary as the regression output.

In this paper we focus on the consequences of using MPMC as the underlying classifier for the problem defined by (8). Once the hyperplane parameters  $\mathbf{a}$  and  $b$  have been determined by the MPMC, we use the classification boundary  $\mathbf{a}^T \mathbf{z} = b$  to predict the output  $\hat{y}$  for a new input  $\mathbf{x}$  (let  $\mathbf{z}$  be defined as  $\mathbf{z} = (\hat{y}, x_1, x_2, \dots, x_d)$ ):

$$\begin{aligned} & \mathbf{a}^T \mathbf{z} = b && \text{(classification boundary)} \\ \Leftrightarrow & a_1 \hat{y} + a_2 x_1 + a_3 x_2 + \dots + a_{d+1} x_d = b \\ \Leftrightarrow & \hat{y} = -\frac{a_2}{a_1} x_1 - \frac{a_3}{a_1} x_2 - \dots - \frac{a_{d+1}}{a_1} x_d + \frac{b}{a_1} && (a_1 \neq 0) \\ \Leftrightarrow & \hat{y} = \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_d x_d + \beta_0 = \beta^T \mathbf{x} + \beta_0 && \text{(regression function)} \end{aligned}$$

Therefore  $\beta_0 = \frac{b}{a_1}, \beta_j = \frac{a_{j+1}}{a_1} (j = 1, \dots, d)$  for the regression model defined in (2) (9)

The symmetric  $\pm\epsilon$  shift of the output variable has a number of useful properties. First of all, we can turn the probability bound  $\alpha$  of correct classification into the minimum probability  $\Omega$  that future predictions will be within  $\pm\epsilon$  accurate. Roughly speaking, if a hyperplane classifies a point of class  $\mathbf{u}$  correctly, then the corresponding  $y$  value can be at most  $\epsilon$  below the hyperplane. Similarly, for points of class  $\mathbf{v}$  that are classified correctly, the corresponding  $y$  value can be at most  $\epsilon$  above the hyperplane (see the next section for a rigorous proof). Second, the artificial classification problem has a much simpler structure than the general classification problem. In particular, we find that  $\Sigma_{\mathbf{u}} = \Sigma_{\mathbf{v}} =: \Sigma$  and  $\bar{\mathbf{u}} - \bar{\mathbf{v}} = (2\epsilon, 0, \dots, 0)^T$ . This allows us to simplify (6) even further; we can minimize  $2\sqrt{\mathbf{a}^T \Sigma \mathbf{a}}$  or equivalently  $\mathbf{a}^T \Sigma \mathbf{a}$  with respect to  $\mathbf{a}$ . The next section shows how to solve this minimization problem by solving just one linear system.

### 3.3 Linear MPMR

Given an estimate  $\Sigma = \Sigma_{\mathbf{u}} = \Sigma_{\mathbf{v}} \in \mathbf{R}^{(d+1) \times (d+1)}$  of the covariance matrix for the random vector  $(y, \mathbf{x})$  that generated the data, we can simplify (6):

$$\min_{\mathbf{a}} \sqrt{\mathbf{a}^T \Sigma_{\mathbf{u}} \mathbf{a}} + \sqrt{\mathbf{a}^T \Sigma_{\mathbf{v}} \mathbf{a}} \quad s.t. \quad \mathbf{a}^T (\bar{\mathbf{u}} - \bar{\mathbf{v}}) = 1 \Leftrightarrow \min_{\mathbf{a}} 2\sqrt{\mathbf{a}^T \Sigma \mathbf{a}} \quad s.t. \quad \mathbf{a}^T (2\epsilon, 0, \dots, 0)^T = 1$$

Since we are only interested in the minimum we can get rid of the square root and the factor of 2. The goal of the linear MPMR can then be formulated as the solution to the following constrained optimization problem for  $\mathbf{a} \in \mathbf{R}^{d+1}$ :

$$\min_{\mathbf{a}} \mathbf{a}^T \Sigma \mathbf{a} \quad s.t. \quad (2\epsilon, 0, \dots, 0) \mathbf{a} = 1 \tag{10}$$

We use Lagrangian multipliers to turn the problem into an unconstrained optimization problem:

$$L(\mathbf{a}, \lambda) = \mathbf{a}^T \Sigma \mathbf{a} + \lambda((2\epsilon, 0, \dots, 0)\mathbf{a} - 1)$$

By setting the derivatives  $\frac{\partial L(\mathbf{a}, \lambda)}{\partial \mathbf{a}} = \mathbf{0}$  and  $\frac{\partial L(\mathbf{a}, \lambda)}{\partial \lambda} = 0$  we obtain the following system of linear equations which describe a solution of (10):

$$2\Sigma \mathbf{a} + (\lambda 2\epsilon, 0, \dots, 0)^T = \mathbf{0} \quad (11)$$

and

$$a_1 = \frac{1}{2\epsilon} \quad (12)$$

where

$$\Sigma = \begin{pmatrix} \sigma_{yy} & \sigma_{yx_1} & \cdots & \sigma_{yx_d} \\ \sigma_{x_1y} & \sigma_{x_1x_1} & \cdots & \sigma_{x_1x_d} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{x_dy} & \sigma_{x_dx_1} & \cdots & \sigma_{x_dx_d} \end{pmatrix} \quad (13)$$

is the full sample covariance matrix for the training data  $\Gamma = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ . The entries of  $\Sigma$  have the following unbiased estimates (Wackerly et al., 2002):

$$\begin{aligned} \sigma_{yy} &= \frac{1}{N-1} \sum_{i=1}^N (y_i - \bar{y})^2 \\ \sigma_{x_jy} &= \frac{1}{N-1} \sum_{i=1}^N (x_{ij} - \bar{x}_j)(y_i - \bar{y}) = \sigma_{yx_j} \quad j = 1, \dots, d \\ \sigma_{x_jx_k} &= \frac{1}{N-1} \sum_{i=1}^N (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k) \quad j = 1, \dots, d; k = 1, \dots, d \end{aligned}$$

The system (11) has  $d+2$  equations and  $d+2$  unknowns:  $a_1, a_2, \dots, a_{d+1}$  and the Lagrange multiplier  $\lambda$  (also note that (12) guarantees the requirement  $a_1 \neq 0$  of (9) for an arbitrary  $\epsilon > 0$ ). For any values  $a_1, a_2, \dots, a_{d+1}$ , the first equation of (11) can always be solved by setting the free variable  $\lambda$  appropriately:

$$\lambda = -\frac{2(\Sigma \mathbf{a})_1}{2\epsilon} \quad \text{where } (\dots)_1 \text{ denotes the first component of a vector.} \quad (14)$$

We write out the other  $d$  equations of (11) and omit the factor of 2. We obtain:

$$\begin{pmatrix} \sigma_{x_1y} & \sigma_{x_1x_1} & \cdots & \sigma_{x_1x_d} \\ \sigma_{x_2y} & \sigma_{x_2x_1} & \cdots & \sigma_{x_2x_d} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{x_dy} & \sigma_{x_dx_1} & \cdots & \sigma_{x_dx_d} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \cdots \\ a_{d+1} \end{pmatrix} = \mathbf{0}$$

Recall that we are interested in the coefficients  $\beta_j$  which can be written as  $\beta_j = -\frac{a_{j+1}}{a_1}$  for  $j = 1, \dots, d$  (see (9)). Therefore we rearrange the system above in terms of  $\beta_j$  (by multiplying out the  $\sigma_{x_jy}a_1$  terms, bringing them to the other side, and dividing by  $-a_1$ ):

$$\begin{pmatrix} a_1\sigma_{x_1y} \\ a_1\sigma_{x_2y} \\ \cdots \\ a_1\sigma_{x_dy} \end{pmatrix} + \begin{pmatrix} \sigma_{x_1x_1} & \sigma_{x_1x_2} & \cdots & \sigma_{x_1x_d} \\ \sigma_{x_2x_1} & \sigma_{x_2x_2} & \cdots & \sigma_{x_2x_d} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{x_dx_1} & \sigma_{x_dx_2} & \cdots & \sigma_{x_dx_d} \end{pmatrix} \begin{pmatrix} a_2 \\ a_3 \\ \cdots \\ a_{d+1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \cdots \\ 0 \end{pmatrix}$$

becomes

$$\begin{pmatrix} \sigma_{x_1x_1} & \sigma_{x_1x_2} & \cdots & \sigma_{x_1x_d} \\ \sigma_{x_2x_1} & \sigma_{x_2x_2} & \cdots & \sigma_{x_2x_d} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{x_dx_1} & \sigma_{x_dx_2} & \cdots & \sigma_{x_dx_d} \end{pmatrix} \begin{pmatrix} a_2 \\ a_3 \\ \cdots \\ a_{d+1} \end{pmatrix} = \begin{pmatrix} -a_1\sigma_{x_1y} \\ -a_1\sigma_{x_2y} \\ \cdots \\ -a_1\sigma_{x_dy} \end{pmatrix}$$

and finally (divide by  $-a_1$  and substitute  $\beta_j = -\frac{a_{j+1}}{a_1}$  for  $j = 1, \dots, d$ )

$$\begin{pmatrix} \sigma_{x_1x_1} & \sigma_{x_1x_2} & \cdots & \sigma_{x_1x_d} \\ \sigma_{x_2x_1} & \sigma_{x_2x_2} & \cdots & \sigma_{x_2x_d} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{x_dx_1} & \sigma_{x_dx_2} & \cdots & \sigma_{x_dx_d} \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \cdots \\ \beta_d \end{pmatrix} = \begin{pmatrix} \sigma_{x_1y} \\ \sigma_{x_2y} \\ \cdots \\ \sigma_{x_dy} \end{pmatrix} \quad (15)$$

One can show (see Appendix A) that this is exactly the same system which appears for solving linear least squares problems. Consequently, the offset  $\beta_0$  should also be equivalent to the offset  $\beta_0 = \bar{y} - \beta^T \bar{\mathbf{x}}$  of the least squares solution. To see why this is true, we start by looking at the offset  $b$  of the hyperplane (see 7):

$$\begin{aligned} b &= \mathbf{a}^T \bar{\mathbf{u}} - \frac{\sqrt{\mathbf{a}^T \Sigma \mathbf{a}}}{\sqrt{\mathbf{a}^T \Sigma \mathbf{a} + \sqrt{\mathbf{a}^T \Sigma \mathbf{a}}}} \\ &= \left( \frac{1}{2\epsilon}, -\frac{\beta_1}{2\epsilon}, \dots, -\frac{\beta_d}{2\epsilon} \right) (\bar{y} + \epsilon, \bar{x}_1, \dots, \bar{x}_d)^T - \frac{1}{2} \\ &= \frac{1}{2\epsilon} (\bar{y} + \epsilon - \sum_{i=1}^d \beta_i \bar{x}_i) - \frac{\epsilon}{2\epsilon} \\ &= \frac{1}{2\epsilon} (\bar{y} - \sum_{i=1}^d \beta_i \bar{x}_i) \end{aligned}$$

Therefore, we have:

$$\beta_0 = \frac{b}{a_1} = b \cdot 2\epsilon = \bar{y} - \sum_{j=1}^d \beta_j \bar{x}_j = \bar{y} - \beta^T \bar{\mathbf{x}} \quad (16)$$

From (15) and (16) we obtain our first theorem:

**Theorem 1** *The linear MPMR algorithm computes the same parameters  $\beta_0, \beta_1, \beta_2, \dots, \beta_d$  as the method of least squares.*

**Proof.** Derivation of (15) and (16). □

We have shown that for the linear case, the regression model we obtain from MPMR is the same as the one obtained from the method of least squares. The MPMR approach, however, allows us to compute a distribution-free lower bound  $\Omega$  on the probability that future data points are predicted within  $\pm\epsilon$  correctly. We know of no other formulation that yields such a bound for commonly used least squares regression. To compute the bound  $\Omega$ , we estimate the covariance matrix  $\Sigma$  from the training data and solve (15) for  $\beta$ . Then, for any given margin  $\epsilon > 0$ , we set  $a_1 = \frac{1}{2\epsilon}$ ,  $a_{j+1} = -\beta_j a_1$  ( $j = 1, \dots, d$ ) and calculate the MPM classification bound  $\alpha$  as:

$$\alpha = \frac{1}{4\mathbf{a}^T \Sigma \mathbf{a} + 1} \quad (17)$$

The following theorem establishes that the  $\alpha$  calculated in (17) can be used to put a lower bound  $\Omega$  on the probability that future predictions are  $\pm\epsilon$  accurate (provided that we have perfect estimates for  $\bar{\mathbf{x}}, \bar{y}$  and  $\Sigma$ ).

**Theorem 2** *For any distributions  $\Lambda_{\mathbf{x}}$  with  $E[\Lambda_{\mathbf{x}}] = \bar{\mathbf{x}}, V[\Lambda_{\mathbf{x}}] = \Sigma_{\mathbf{x}}$  and  $\Lambda_{\rho}$  with  $E[\Lambda_{\rho}] = 0, V[\Lambda_{\rho}] < \infty$  (where  $\bar{\mathbf{x}}, \Sigma_{\mathbf{x}}$  are finite) and  $\mathbf{x} = (x_1, \dots, x_d) \sim \Lambda_{\mathbf{x}}, \rho \sim \Lambda_{\rho}$  let  $y = f^*(\mathbf{x}) + \rho$  be the (noisy) output of the true regression function. Assume perfect knowledge of the statistics  $\bar{\mathbf{x}}, \bar{y}, \Sigma$  (where  $\Sigma$  is defined in (1)). Then the MPMR model  $\hat{f} : \mathbf{R}^d \rightarrow \mathbf{R}$  satisfies:*

$$\inf_{(\mathbf{x}, y) \sim (\bar{\mathbf{x}}, \bar{y}, \Sigma)} Pr\{|\hat{f}(\mathbf{x}) - y| < \epsilon\} \geq \max\{2\alpha - 1, 0\} =: \Omega \quad (18)$$

**Proof.** For any  $d + 1$  dimensional vector  $\mathbf{z} = (y, x_1, \dots, x_d)$  we have:

$$\begin{aligned} \mathbf{z} \text{ is classified as class } \mathbf{u} &\Leftrightarrow y > \hat{f}(x) \\ \mathbf{z} \text{ is classified as class } \mathbf{v} &\Leftrightarrow y < \hat{f}(x) \end{aligned}$$

(this follows directly from the construction of the  $d + 1$  dimensional hyperplane).

Now look at the random variables  $\mathbf{z}_{+\epsilon} = (y + \epsilon, \mathbf{x})$  and  $\mathbf{z}_{-\epsilon} = (y - \epsilon, \mathbf{x})$ . The distribution of  $\mathbf{z}_{+\epsilon}$  satisfies  $\mathbf{z}_{+\epsilon} \sim ((\bar{y} + \epsilon, \bar{\mathbf{x}}), \Sigma)$  which has the same mean and covariance matrix as our (artificial) class  $\mathbf{u} \sim (\bar{\mathbf{u}}, \Sigma_{\mathbf{u}})$ . Similarly, we find that  $\mathbf{z}_{-\epsilon} \sim (\bar{\mathbf{v}}, \Sigma_{\mathbf{v}})$ . The MPM classifier guarantees that  $\alpha$  is a lower bound for correct classification and consequently  $1 - \alpha$  is an upper bound for misclassification. We note that for regression a prediction is *not*  $\pm\epsilon$  accurate if at least one of the points  $\mathbf{z}_{+\epsilon}$  or  $\mathbf{z}_{-\epsilon}$  is misclassified:

$$\begin{aligned} Pr\{|\hat{f}(\mathbf{x}) - y| \geq \epsilon\} &= Pr\{\mathbf{z}_{+\epsilon} \text{ misclassified} \vee \mathbf{z}_{-\epsilon} \text{ misclassified}\} \\ &\leq Pr\{\mathbf{z}_{+\epsilon} \text{ misclassified}\} + Pr\{\mathbf{z}_{-\epsilon} \text{ misclassified}\} \\ &\leq (1 - \alpha) + (1 - \alpha) = 2 - 2\alpha \end{aligned}$$

Therefore:

$$Pr\{|\hat{f}(\mathbf{x}) - y| < \epsilon\} \leq 1 - (2 - 2\alpha) = 2\alpha - 1$$

Since all probabilities are  $\geq 0$ , we can write  $\max\{2\alpha - 1, 0\}$  as the minimum probability of making  $\pm\epsilon$  correct predictions. □

We note that the two-sided bound  $\Omega$  is *not* tight for all  $\epsilon$ . This is a direct consequence of the looseness of the bound (5) given by Marshall and Olkin (1960). We will discuss this fact in more detail in section 3.4.

One might also think that it might be better to consider  $Pr\{|\hat{f}(\mathbf{x}) - f^*(\mathbf{x})| < \epsilon\}$  instead of  $Pr\{|\hat{f}(\mathbf{x}) - y| < \epsilon\}$ . The disadvantage of  $Pr\{|\hat{f}(\mathbf{x}) - f^*(\mathbf{x})| < \epsilon\}$  is that this formulation only considers noise in the training data, but does not account for the noise in the test data. The term  $Pr\{|\hat{f}(\mathbf{x}) - y| < \epsilon\}$  does account for this because the random variable  $y$  is the noisy observation of the true regression function  $f^*$ .

In the following paragraphs we derive the interesting relation that exists between the distribution-free probability bound  $\Omega$ , the  $\pm\epsilon$  interval, and the training error of MPMR (or, equivalently, least squares). We start by filtering the  $\epsilon$  out of the expression  $4\mathbf{a}^T\Sigma\mathbf{a}$ . By plugging in  $a_1 = \frac{1}{2\epsilon}$  and  $a_{j+1} = -\beta_j a_1$  for  $j = 1, \dots, d$  it follows that:

$$4\mathbf{a}^T\Sigma\mathbf{a} = \frac{1}{\epsilon^2} \begin{pmatrix} 1 \\ -\beta \end{pmatrix}^T \Sigma \begin{pmatrix} 1 \\ -\beta \end{pmatrix} =: \frac{\nu}{\epsilon^2} \quad (19)$$

As indicated above we define the part that is independent of  $\epsilon$  as  $\nu := \begin{pmatrix} 1 \\ -\beta \end{pmatrix}^T \Sigma \begin{pmatrix} 1 \\ -\beta \end{pmatrix}$ . In the following theorem we show that if the offset  $b$  is set to  $b = \bar{y} - \beta^T\mathbf{x}$ , then  $\nu$  is essentially equivalent to the mean squared training error.<sup>2</sup>

**Lemma (MSE)** *Let  $y = \sum_{j=1}^d \beta_j x_j + b$  be a linear model with arbitrary coefficients  $\beta_j$  and an offset  $b = \bar{y} - \beta^T\mathbf{x}$ . Let  $MSE' = \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{N-1}$  denote the “scaled” mean squared error on the training data. Then  $\nu = MSE'$ .*

**Proof.**

We first write out  $\nu$ :

$$\nu = \begin{pmatrix} 1 \\ -\beta \end{pmatrix}^T \Sigma \begin{pmatrix} 1 \\ -\beta \end{pmatrix} = \sigma_{yy} - 2 \sum_{j=1}^d \beta_j \sigma_{yx_j} + \sum_{j=1}^d \sum_{k=1}^d \beta_j \beta_k \sigma_{x_j x_k}$$

The “scaled” mean squared error  $MSE'$  is equivalent to:

$$\begin{aligned} \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{N-1} &= \frac{\sum_{i=1}^N (y_i - \sum_{j=1}^d \beta_j x_{ij} - \bar{y} + \sum_{j=1}^d \beta_j \bar{x}_j)^2}{N-1} \\ &= \frac{\sum_{i=1}^N (y_i - \bar{y})^2 + 2(y_i - \bar{y}) \sum_{j=1}^d \beta_j (\bar{x}_j - x_{ij}) + (\sum_{j=1}^d \beta_j (\bar{x}_j - x_{ij}))^2}{N-1} \\ &= \frac{\sum_{i=1}^N (y_i - \bar{y})^2}{N-1} - \frac{2 \sum_{j=1}^d \beta_j \sum_{i=1}^N (y_i - \bar{y})(x_{ij} - \bar{x}_j)}{N-1} + \frac{\sum_{j=1}^d \sum_{k=1}^d \beta_j \beta_k \sum_{i=1}^N (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)}{N-1} \\ &= \sigma_{yy} - 2 \sum_{j=1}^d \beta_j \sigma_{yx_j} + \sum_{j=1}^d \sum_{k=1}^d \beta_j \beta_k \sigma_{x_j x_k} \quad \square \end{aligned}$$

Next we show that the “scaled” mean squared error  $\nu$  establishes the relation between  $\Omega$  and  $\epsilon$ .

**Theorem 3** *For any linear regression model  $f(\mathbf{x}) = \sum_{j=1}^d \beta_j x_j + b$  with offset  $b = \bar{y} - \beta^T\mathbf{x}$ , we have the following relationships between the distribution-free lower bound probability  $\Omega$ , the margin size  $\epsilon$ , and the “scaled” mean squared error  $\nu$ :*

$$(i) \quad \Omega = \max\left(\frac{1-\nu/\epsilon^2}{1+\nu/\epsilon^2}, 0\right) \quad (20)$$

$$(ii) \quad \epsilon = \sqrt{\nu \frac{1+\Omega}{1-\Omega}} \quad (21)$$

**Proof.** We know from Theorem 2 that  $\Omega = 2\alpha - 1$  (provided this term is nonnegative). The MSE Lemma lets us then rewrite:

$$\Omega = 2\alpha - 1 = \frac{2}{1+4\mathbf{a}^T\Sigma\mathbf{a}} - 1 = \frac{2}{1+\nu/\epsilon^2} - 1 = \frac{2-(1+\nu/\epsilon^2)}{1+\nu/\epsilon^2} = \frac{1-\nu/\epsilon^2}{1+\nu/\epsilon^2}$$

2. With the slight difference that here the sum of the squared errors is divided by  $N - 1$  and not  $N$ .

If we solve the equation above for  $\epsilon$  we obtain part (ii) of Theorem 3.

□

### 3.4 Relation to Chebyshev Inequality and Tightness of MPMR Bound

We will analyse the tightness of the MPMR bound for a “family” of worst case distributions. Assume that the regression variable  $y$  is independent from the input  $\mathbf{x}$  and has a distribution with mean  $\bar{y} = 0$  and variance  $\sigma^2 = 1$ . It is easy to show that MPMR will yield a model with  $f(\mathbf{x}) = 0$  for any  $\mathbf{x}$ . Applying the two-sided Chebyshev inequality to the random variable  $y$ , we obtain:

$$\begin{aligned} Pr\{|y - E[y]| \geq k\sqrt{Var[y]}\} &\leq 1/k^2 & (\epsilon := k\sqrt{Var[y]} = k\sqrt{1} = k) \\ Pr\{|y - 0| \geq \epsilon\} &\leq 1/\epsilon^2 & (f(\mathbf{x}) = 0) \\ Pr\{|f(\mathbf{x}) - y| < \epsilon\} &\geq 1 - 1/\epsilon^2 = (\epsilon^2 - 1)/\epsilon^2 \end{aligned}$$

The two-sided MPMR bound  $\Omega$  relies on two *one-sided* Chebyshev bounds which come from the fact that the underlying MPM classifier places one-sided bounds on false positives and false negatives. For this particular distribution we have:

$$\alpha = \frac{1}{1/\epsilon^2 + 1} \Rightarrow \Omega = 2\alpha - 1 = \frac{2}{1/\epsilon^2 + 1} - 1 = (\epsilon^2 - 1)/(\epsilon^2 + 1)$$

One family of distributions with mean 0, variance 1 that achieves equality for the two-sided Chebyshev inequality is the following:

$$Pr\{y = 0\} = p, Pr\{y = -\sqrt{1/(1-p)}\} = Pr\{y = \sqrt{1/(1-p)}\} = (1-p)/2 \quad (0 \leq p < 1)$$

It is easy to see that the Chebyshev inequality is tight for  $\epsilon = \sqrt{1/(1-p)}$ . The worst case distribution has  $Pr\{|f(\mathbf{x}) - y| < \epsilon\} = p$  since only  $y = 0$  has a distance *less* than  $\epsilon$ . The Chebyshev formula yields:

$$Pr\{|f(\mathbf{x}) - y| < \epsilon\} \geq 1 - 1/\epsilon^2 = 1 - \frac{1}{1/(1-p)} = p$$

Figure 2 shows how tight the MPMR bound  $\Omega$  is for various values of  $\epsilon$ . If  $\epsilon$  is less than 1, no test point will be within  $\pm\epsilon$  for that particular distribution. For values  $\epsilon > 1$ , the Chebyshev bound is tighter  $((\epsilon^2 - 1)/\epsilon^2)$  than the MPMR bound  $\Omega = (\epsilon^2 - 1)/(\epsilon^2 + 1)$ .

## 4. Nonlinear Framework

We now extend the concepts of MPMR to nonlinear regression using the basis function framework:

$$f(\mathbf{x}) = \sum_{i=1}^k \beta_i \Phi_i(\mathbf{x}) + \beta_0$$

where the basis functions  $\Phi_i : \mathbf{R}^d \rightarrow \mathbf{R}$  can be arbitrary nonlinear mappings.

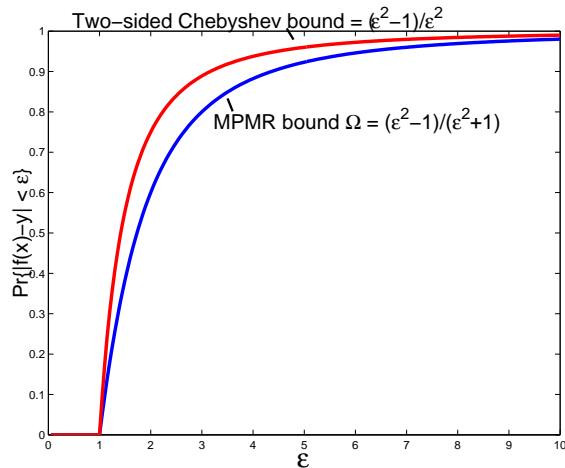


Figure 2: For values of  $\epsilon$  less than the standard deviation of  $y$  neither Chebyshev nor MPMR can give a nontrivial bound because all values for  $y$  could be exactly one standard deviation away from the mean. As  $\epsilon$  increases, the gap between the Chebyshev bound and the MPMR bound  $\Omega$  first increases but then decreases as both bounds approach 1.

#### 4.1 Kernel Maps as Features

In this paper we focus our analysis and experiments on Mercer kernels and use the kernel maps  $K(\mathbf{x}_i, \cdot)$  as basis functions. Instead of  $d$  features each input vector is now represented by  $N$  features: the kernel map evaluated at all of the other training inputs (including itself).

$$f(\mathbf{x}) = \sum_{i=1}^N \beta_i K(\mathbf{x}_i, \mathbf{x}) + \beta_0$$

We then solve the regression problem above with the linear MPMR, i.e. we minimize the following least squares expression:

$$\min_{\beta} \sum_{i=1}^N (y_i - (\sum_{j=1}^N \beta_j z_{ij} + \beta_0))^2$$

where

$$z_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) \quad i = 1, \dots, N; \quad j = 1, \dots, N$$

A new input  $\mathbf{x}$  is then evaluated as:

$$\hat{y} = \hat{f}(\mathbf{x}) = \sum_{i=1}^N \beta_i K(\mathbf{x}_i, \mathbf{x}) + \beta_0$$

Commonly used kernel functions include the linear  $K(\mathbf{s}, \mathbf{t}) = \mathbf{s}^T \mathbf{t}$ , the polynomial  $K(\mathbf{s}, \mathbf{t}) = (\mathbf{s}^T \mathbf{t} + 1)^p$  and the Gaussian kernel  $K(\mathbf{s}, \mathbf{t}) = \exp(-\kappa(\mathbf{s} - \mathbf{t})^T(\mathbf{s} - \mathbf{t}))$  (see for example

Schölkopf and Smola, 2002). Note that the learning algorithm now has to solve an  $N$ -by- $N$  system instead of a  $d$ -by- $d$  system:

$$\begin{pmatrix} \sigma_{z_1 z_1} & \sigma_{z_1 z_2} & \cdots & \sigma_{z_1 z_N} \\ \sigma_{z_2 z_1} & \sigma_{z_2 z_2} & \cdots & \sigma_{z_2 z_N} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{z_N z_1} & \sigma_{z_N z_2} & \cdots & \sigma_{z_N z_N} \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \cdots \\ \beta_N \end{pmatrix} = \begin{pmatrix} \sigma_{z_1 y} \\ \sigma_{z_2 y} \\ \cdots \\ \sigma_{z_N y} \end{pmatrix} \quad (22)$$

and setting

$$\beta_0 = \bar{y} - \sum_{i=1}^N \beta_j \bar{z}_j \quad (23)$$

where

$$\begin{aligned} \bar{y} &= \frac{1}{N} \sum_{i=1}^N y_i \\ \bar{z}_j &= \frac{1}{N} \sum_{i=1}^N z_{ij} & j = 1, \dots, N \\ \sigma_{z_j y} &= \frac{1}{N-1} \sum_{i=1}^N (z_{ij} - \bar{z}_j)(y_i - \bar{y}) = \sigma_{y z_i} & j = 1, \dots, N \\ \sigma_{z_j z_k} &= \frac{1}{N-1} \sum_{i=1}^N (z_{ij} - \bar{z}_j)(z_{ik} - \bar{z}_k) & j = 1, \dots, N; k = 1, \dots, N \end{aligned}$$

The proof of Theorem 3 generalizes in a straightforward way to the basis function framework:

**Corollary to Theorem 3** *For any basis function regression model  $f(\mathbf{x}) = \sum_{i=1}^N \beta_i \Phi_i(\mathbf{x}) + \beta_0$  we have the relationships (20) and (21) between the distribution-free lower bound probability  $\Omega$ , the margin size  $\epsilon$ , and the “scaled” mean squared error  $\nu$ , which is in the nonlinear case defined by:*

$$\nu = \begin{pmatrix} 1 \\ -\beta \end{pmatrix}^T \Sigma_{\Phi} \begin{pmatrix} 1 \\ -\beta \end{pmatrix}$$

**Proof.** Analogous to proof of Theorem 3. □

## 4.2 Regularization

A common issue of kernel estimation is that it can lead to overfitting the training data. One way around this problem is to use a regularization parameter to bound the magnitude of the vector  $\beta$ . Instead of solving the standard (i.e. unregularized) least squares optimization one solves the following problem:

$$\min_{\beta, \beta_0} \sum_{i=1}^n (z_{\cdot i}^T \beta + \beta_0 - y_i)^2 + \lambda \beta^T \beta \quad (24)$$

(where  $z_{\cdot i} = (z_{1i}, z_{2i}, \dots, z_{ni})^T$ ). Setting the derivative of (24) with respect to  $\beta$  and  $\beta_0$  to 0, leads to the following system for solving regularized least squares:

$$\begin{pmatrix} \sum_{i=1}^n z_{\cdot i} z_{\cdot i}^T + \lambda I & \sum_{i=1}^n z_{\cdot i} \\ \sum_{i=1}^n z_{\cdot i}^T & n \end{pmatrix} \begin{pmatrix} \beta_1 \\ \cdots \\ \beta_n \\ \beta_0 \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n y_i z_{\cdot i} \\ \sum_{i=1}^n y_i \end{pmatrix} \quad (25)$$

For  $\lambda = 0$  the system reduces to the unregularized case and it is equivalent to (22) and (23).

### 4.3 A Different Approach

In previous work (Strohmamm and Grudic, 2003) we proposed a different nonlinear MPMR formulation which is based on the nonlinear MPM for classification by Lanckriet et al. (2002a). Instead of using kernel maps as features, this formulation finds a minimax hyperplane in an higher dimensional space. Like kernelized support vector machines, one considers the (implicit) mapping  $\varphi : \mathbf{R}^d \rightarrow \mathbf{R}^h$  and its dot products  $\varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j) =: K(\mathbf{x}_i, \mathbf{x}_j)$ . This MPMR algorithm (we call it kernel-MPMR since it requires Mercer kernels) was also obtained by using the MPM classifier on an artificial data set where the dependent variable, shifted by  $\pm\epsilon$ , is added as first component to the input vector (yielding the  $d+1$ -dimensional vector  $(y, x_1, \dots, x_d)$ ). The kernel-MPMR model looks like:

$$\sum_{i=1}^{2N} \gamma_i K(\mathbf{z}_i, \mathbf{z}) + \gamma_0 = 0 \quad (26)$$

where  $\mathbf{z}_i = (y_i + \epsilon, x_{i1}, \dots, x_{id})$  for  $i = 1, \dots, N$  and  $\mathbf{z}_i = (y_{i-N} - \epsilon, x_{i-N,1}, \dots, x_{i-N,d})$  for  $i = N + 1, \dots, 2N$ . Evaluating a new point  $\mathbf{x}$  amounts to plugging in  $\mathbf{z} = (\hat{y}, \mathbf{x})$  into (26) and solving the equation for  $\hat{y}$ . For an arbitrary kernel this would imply that we have to deal with a nonlinear equation which is at best computationally expensive and at worst not solvable at all. Therefore, the kernel is restricted to be *output-linear*, i.e. we require:

$$K((s_1, s_2, \dots, s_{d+1}), (t_1, t_2, \dots, t_{d+1})) = s_1 t_1 + K'((s_2, \dots, s_{d+1}), (t_2, \dots, t_{d+1})) \quad (27)$$

The theory of positive definite kernels (see for example Schölkopf and Smola, 2002) states that positive definite kernels are closed under summation, which means that we could use an arbitrary positive kernel for  $K'$  and obtain a valid kernel  $K$  (of course the linear output kernel is also positive definite). By using a kernel  $K$  of the form (27), we can analytically solve (26) for  $y$  and obtain the regression model:

$$y = (-2\epsilon) \left[ \left( \sum_{i=1}^N (\gamma_i + \gamma_{i+N}) K'(\mathbf{x}_i, \mathbf{x}) \right) + \gamma_0 \right]$$

If we compare the two different MPM regression algorithms we note that the computational cost of learning an kernel-MPMR model is higher, because instead of an  $N \times N$  system we have to solve an  $2N \times 2N$  system to obtain the coefficients  $\gamma$ . For MPMR we are not restricted to Mercer kernels, but can use any set  $\Phi_i$  of nonlinear basis functions. It is an open question as to whether MPMR and kernel-MPMR are equivalent if we use Mercer kernels for the MPMR, or whether one has a theoretical or practical advantage over the other.

## 5. Robust Estimates

The MPMR learning algorithm estimates the covariance matrix from the training data. Of course these statistical estimates will be different from the true values and thus the regression surface itself, and the lower bound  $\Omega$ , may be inaccurate. Especially for the nonlinear case the problem of overfitting the training data with highly sensitive kernel

functions arises, so our goal is to find kernels that produce robust estimates. One way of dealing with the problem of selecting the right kernel is to do cross validation on the training data and pick the kernel which yields the best result. Cross validation can also be used to achieve robustness via regularization, i.e. finding a penalty factor  $\lambda$  for big coefficients  $\beta_i$  (Schölkopf and Smola, 2002). In this paper we explore a different approach which is tied to the probability bound  $\Omega$ , and is furthermore computationally more efficient than doing cross validation.

The idea is to select a fraction  $\gamma$  of rows that will be left out when calculating the covariance matrix (note that we measure the covariance columnwise, i.e. deleting rows will not affect the dimension of the covariance matrix). If the covariance matrix of the remaining training data is "reasonably close" to the covariance matrix of all the training data, we consider our model to be robust.

We now define in mathematical terms what "reasonably close" means. The objective of the MPMR algorithm is to maximize the lower bound probability  $\Omega$ , which is determined by the term  $\mathbf{a}^T \Sigma \mathbf{a}$ . Therefore, we measure the sensitivity of an MPMR model by looking at the sensitivity in  $\mathbf{a}^T \Sigma \mathbf{a}$ . We call a model *MPMR-robust* if the relative error for this expression is  $\leq \delta$  when leaving a fraction of  $\gamma$  points out for computing the covariance matrix:

### Definition

$$\begin{array}{ll} \Sigma_* & \text{- true covariance matrix} & \mathbf{a}_* & \text{- minimizes } \mathbf{a}_*^T \Sigma_* \mathbf{a}_* \text{ s.t. } a_{*1} = \frac{1}{2\epsilon} \\ \Sigma & \text{- cov. matrix of all } N \text{ examples} & \mathbf{a} & \text{- minimizes } \mathbf{a}^T \Sigma \mathbf{a} \text{ s.t. } a_1 = \frac{1}{2\epsilon} \\ \Sigma_\gamma & \text{- cov. matrix of } (1 - \gamma)N \text{ examples} & \mathbf{a}_\gamma & \text{- minimizes } \mathbf{a}_\gamma^T \Sigma_\gamma \mathbf{a}_\gamma \text{ s.t. } a_{\gamma 1} = \frac{1}{2\epsilon} \end{array}$$

$$\text{A model is } \textit{MPMR-robust} \Leftrightarrow E\left[\frac{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma - \mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma}\right] \leq \delta$$

We cannot apply this definition directly since we don't know the true covariance matrix  $\Sigma_*$ . However, we will prove that under the following two assumptions we can obtain a formula that does not involve  $\Sigma_*$ :

$$\begin{array}{l} 1) E[\mathbf{a}^T \Sigma_* \mathbf{a}] \leq E[\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma] \\ 2) E\left[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}\right] \geq E\left[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}^T \Sigma_* \mathbf{a}}\right] \end{array} \quad (28)$$

The first assumption states the whole model is more accurate than the model obtained when we leave out a fraction  $\gamma$  of data points. The second assumption says that if we leave out a *small* fraction of points, we expect that this model ( $\mathbf{a}_\gamma$ ) is closer to the whole model ( $\mathbf{a}$ ), than the whole model is to the optimal model ( $\mathbf{a}_*$ ). In Appendix B we give experimental evidence that these two assumptions do hold in practice.

Given the above assumptions, the following theorem holds:

**Theorem 4** *If a model is MPMR-robust then  $\frac{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma - \mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma} \leq \delta$ . In formulas:*

$$E\left[\frac{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma - \mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma}\right] \leq \delta \Rightarrow E\left[\frac{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma - \mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}\right] \leq \delta \quad (29)$$

### Proof.

$$\text{from 2) } E\left[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}\right] \geq E\left[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}^T \Sigma_* \mathbf{a}}\right]$$

from 1)  $E[\mathbf{a}^T \Sigma_* \mathbf{a}] \leq E[\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma] \Rightarrow E[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a} \Sigma_*^T \mathbf{a}}] \geq E[\frac{\mathbf{a}_* \Sigma_* \mathbf{a}_*}{\mathbf{a}_\gamma \Sigma_* \mathbf{a}_\gamma}]$   
 from premise)  $E[\frac{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma - \mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma}] \leq \delta \Leftrightarrow E[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma}] \geq 1 - \delta$

combine inequalities:

$$E[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}] \geq E[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}^T \Sigma_* \mathbf{a}}] \geq E[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma}] \geq 1 - \delta$$

$$E[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}] \geq 1 - \delta \Leftrightarrow 1 - E[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}] \leq \delta \Leftrightarrow E[\frac{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma - \mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}] \leq \delta$$

□

We refer to the number

$$S := E \left[ \frac{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma - \mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma} \right] \quad (30)$$

as the *sensitivity measure*. In our experiments we apply the theorem to show that a model is *not MPMR-robust*. A model where we find that  $S > \delta$  will be rejected because of too much sensitivity in  $\Omega$ .

A different approach for constructing a robust minimax probability machine has been investigated by Lanckriet et al. (2002b) for the task of classification. There the main idea is to put some error bounds on the plug-in estimates of mean and covariance. Instead of a single mean, the robust classifier considers an ellipsoid of means which is centered around the estimated mean and bound in size by an error bound parameter  $\nu$ . Similarly, instead of a single covariance matrix one looks at a matrix ball, where each matrix is at most  $\rho$  away (measured in Frobenius norm) from the estimated covariance matrix. When a robust model is calculated, the lower bound probability  $\alpha$  (for correct classification) decreases when either  $\nu$  or  $\rho$  increase. The model itself is independent of  $\nu$ , but it can depend on  $\rho$  and a small toy example shows (see Lanckriet et al., 2002b, p. 578) that a robust model ( $\rho > 0$ ) can yield a better performance than a simple one (where  $\rho = 0$ ).

From a theoretical point of view the two approaches differ in that our method is *output* oriented (it considers the sensitivity of  $\Omega$  which is the objective that an MPM maximizes), whereas the other method is more *input* oriented (it considers the sensitivity of the plug-in estimates derived from the training data). Both approaches rely on sampling methods to make robust estimates: we sample a fraction  $(1 - \gamma)$  percent of examples to determine the sensitivity of  $\Omega$ ; Lanckriet et al. (2002b) use sampling methods to determine reasonable values for the parameters  $\nu$  and  $\rho$ . Our method, however, has the advantage that it provides a straightforward way to do model selection: choose the model that has a maximal  $\Omega$  while satisfying  $S < \delta$ . It is not immediately clear how a strategy for model selection can be implemented with the approach of Lanckriet et al. (2002b). It is an open question as for which practical applications either approach is preferable over the other.

## 5.1 Efficiently Estimating $S$

In order to be a viable approach for finding robust models, the quantity  $S$  must be efficiently computable. In particular, we don't want to have to build a whole new model when leaving out a fraction of  $\gamma$  training points. A recent result of Strohmamm et al. (2003) that is based on the linearity of covariance, states that we can incrementally update an MPM model if

just one training example is added or removed. The cost for this update is (for nonlinear kernel models)  $O(N)$ . Here we use this result to compute “leave one out” estimates for the sensitivity measure  $S$ . We set  $\gamma = \frac{1}{N}$  and estimate  $S$  by averaging over all  $N$  models that have one example left out:

$$\widehat{S} = \left( \sum_{i=1}^N \frac{\mathbf{a}_{-i}^T \Sigma \mathbf{a}_{-i} - \mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_{-i}^T \Sigma \mathbf{a}_{-i}} \right) / N \quad (31)$$

Where  $\mathbf{a}_{-i}$  is the solution of the MPMR when data point  $i$  is left out.

This brings the computational cost for estimating  $S$  to  $O(n^2)$  – which is significantly less than the  $O(n^3)$  needed to build the initial model.

## 6. Experimental Results

In section 6.1 we compare the MPMR algorithm to two well known learning methods – Gaussian processes and support vector machines. In sections 6.2 - 6.4 we evaluate the accuracy of the MPMR and its probability bound  $\Omega$  on real world data sets. The Gaussian kernel  $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\kappa \|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$  is used to obtain a nonlinear regression surface. We plot the behavior of  $\Omega$  and  $S$  for a range of kernel parameters  $\kappa$ . Regularization is done by using singular value decomposition with a threshold of  $10^{-8}$  for the singular values. For the sensitivity measure  $S$  we use the heuristic  $\delta = \gamma = \frac{1}{N}$  for all our experiments. To quantify how accurate our lower bound estimate  $\Omega$  is, we compute the fraction of test points that are within  $\pm\epsilon$  correct for each test run (we call this  $\Omega^T$ ). The common behavior in all data sets is that small values for  $\kappa$  (which yield robust models) have a small value for  $S$ , i.e. the corresponding model is accepted and we find that  $\Omega^T > \Omega$  which means that the lower bound holds. As  $\kappa$  increases, we obtain less robust estimates of the covariance matrix and therefore less robust MPMR models (with  $S > \delta$ ) that will be rejected. The point of interest is where the sensitivity measure  $S$  becomes bigger than  $\delta$  because, this will be the place where we lose confidence in our model and our  $\Omega$  bound.

### 6.1 Relation to other Algorithms

In this section we relate the proposed MPMR algorithm to two well known learning algorithms: Gaussian processes and support vector machines. We show that the error estimates obtained by gaussian processes can be violated for noise distributions that are non Gaussian. As we will see, the worst case distribution bounds of the MPMR will still hold for such distributions. To demonstrate the implication of Theorem 3 and its corollary, we build a model from a support vector machine learning algorithm, calculate its  $\Omega$  bound, and give empirical evidence that the bound will be useful and accurate in practice.

#### 6.1.1 COMPARISON TO GAUSSIAN PROCESSES

In this section we use the *sinc* function as toy data and add a noise variable  $\rho$  that is *not* drawn from a (single) Gaussian distribution. Instead, we generate a distribution from two separate Gaussians: with probability  $p_1 = 0.5$ ,  $\rho$  is drawn from  $\mathcal{N}(\mu, 1)$  and with probability  $p_2 = 0.5$ ,  $\rho$  is drawn from  $\mathcal{N}(-\mu, 1)$  (see Figure 3b). We tested 30 values for  $\mu$  in the range

$\mu = 0.1, 0.2, \dots, 3.0$ . Both the training and test set had  $N_{train} = N_{test} = 100$  examples. The

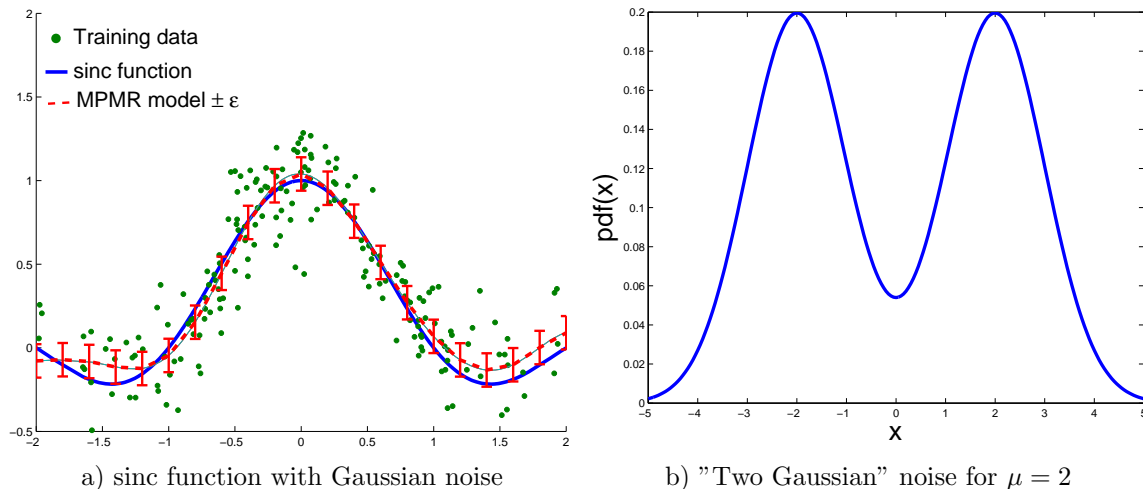


Figure 3:

Gaussian process implementation we used was provided by Gibbs (1997b). We calculated the error estimates of the Gaussian process for  $\sigma =$  one standard deviation. In the MPMR framework this corresponds to a value of  $\Omega = 0.6829$  (i.e. the integral of the Gaussian pdf over the mean  $\pm$  one standard deviation). By using (21) we computed the MPMR error bounds for each distribution. As expected, the Gaussian process error estimates  $\epsilon_{GP}$  were tighter (i.e. more optimistic) than the worst case MPMR error bounds  $\epsilon_{MPMR}$  (see Figure 4a).

For small values of  $\mu$ , where the noise distribution is still close to a (single) Gaussian, the Gaussian process error estimates are still accurate, i.e. they match the measured percentage of  $\pm\epsilon$  accurate predictions. As  $\mu$  increases and the noise distribution becomes less Gaussian, the Gaussian process error estimates no longer match the prediction accuracy. The MPMR error bounds, however, are valid for all possible values of  $\mu$  (see Figure 3):

### 6.1.2 BOUNDS FOR SVMs

As mentioned earlier (section 4.1), we can take the coefficients of any linear model or basis function model and use them to compute a lower bound on the probability of making  $\pm\epsilon$  correct predictions. A widely used state of the art method is the support vector machine regression (Smola and Schölkopf, 1998). For our experiments we used the  $\nu$ -SVM implementation of Chang and Lin (2003) and computed the probability bound according to (20). The results were averaged over 50 runs for the Boston Housing Data and are shown in Figure 5. Notice that the bound holds for all values of  $\epsilon$ .

## 6.2 Boston Housing

The first real world data set we experimented with is the Boston Housing data, which consists of  $N = 506$  examples, each with  $d = 13$  real valued inputs and one real valued output. We divided the data randomly into a training set of size  $N_{train} = 481$  and a test

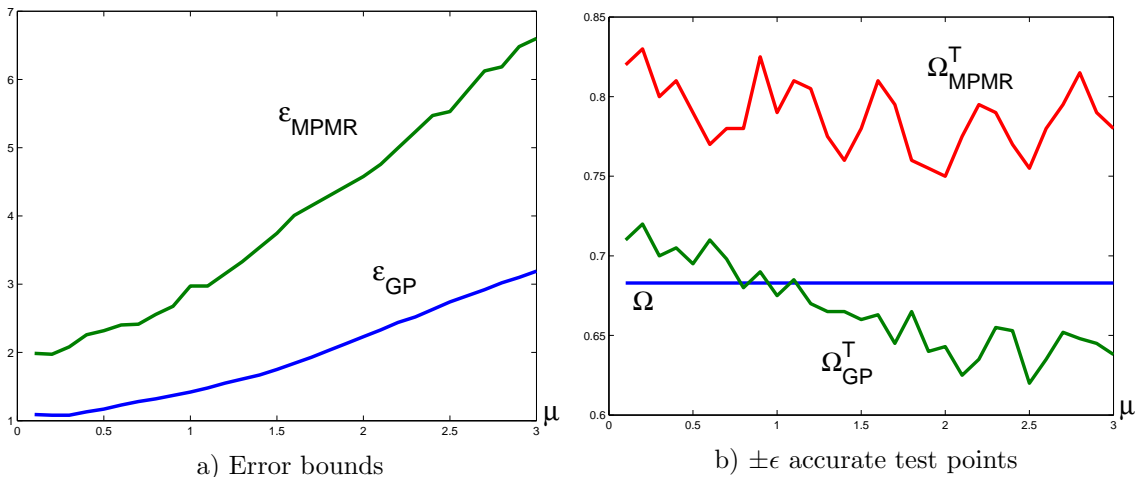


Figure 4: Comparison MPMR with Gaussian processes (GP) on the sinc data with a “two Gaussian” noise. Plot a) shows the error bounds for  $\Omega = 0.6829$  which corresponds to 1 standard deviation of a Gaussian distribution. The worst case bound of MPMR ( $\epsilon_{MPMR}$ ) is looser than the Gaussian process estimate ( $\epsilon_{GP}$ ). Plot b) shows the actual percentage  $\Omega^T$  of test points within the error bounds given by the MPMR and the GP model. While the MPMR bound is always valid, the estimate obtained by Gaussian processes does not hold for the “two Gaussian” noise with a large  $\mu$ .

set of size  $N_{test} = 25$ . We tried 50 different values for  $\kappa$ , and for each of these we averaged our results over 100 runs. The results in Figure 6 indicate that the sensitivity measure  $S$  is suitable for figuring out when a model is robust, both in terms of giving a valid lower bound  $\Omega$  and producing a low mean squared error on the test data. The models that are rejected because of a high sensitivity measure  $S$  have inaccurate bounds and a high (normalized) mean squared error (see Figure 6).

### 6.3 Abalone

The Abalone data set contains  $N = 4177$  examples, each with eight inputs and one real valued output. One of the input features is not a real number but can take on one of three distinct values. We split up the discrete valued feature into 3 real valued features, setting one component to 1 and the other two to 0, which increases the input dimension to  $d = 10$ . The data was then divided randomly into  $N_{train} = 1000$  training examples and  $N_{test} = 3177$  test examples. As in the Boston experiment we went over a range of kernel parameters (here 30) and averaged the results over a number of test runs (here 50) (see Figure 6).

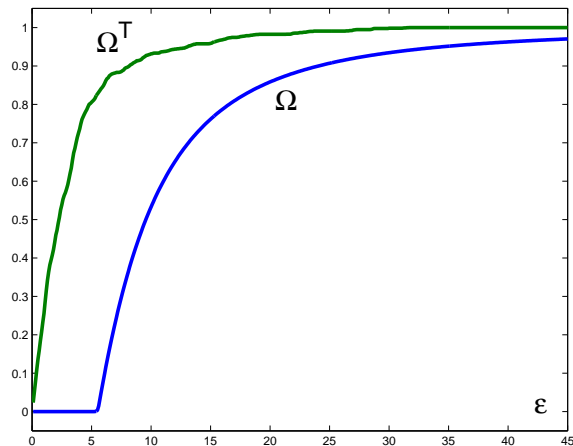


Figure 5: Minimax probability bound ( $\Omega$ ) and percentage of test points within  $\pm\epsilon$  ( $\Omega^T$ ) for the  $\nu$ -SVM on the Boston Housing data set. Observe that the lower probability bound  $\Omega$  holds for all possible  $\epsilon$ . Values for the SVM parameters  $C$  and  $\nu$  and for the kernel function were obtained by cross-validation.

#### 6.4 Kinematic Data

The KIN8NM data set consists of  $N = 8192$  examples, each with eight real valued inputs and one real valued output. We used the same parameters as in the ABALONE data set:  $N_{train} = 1000$  training examples, a range of 30 kernel parameters and 50 test runs to average the results for each kernel parameter (see Figure 7). The sensitivity measure  $S$  was again useful for distinguishing models where the bound holds from models where it does not hold. In terms of mean squared error we found that there were also some good models that would have been rejected by the sensitivity measure. One reason for this result might be that the heuristic  $\delta = \gamma$  that we used is not the optimal choice for all data sets.

### 7. Conclusion & Future Work

We have presented the MPMR framework as a new approach to address regression problems where models are represented as nonlinear (or linear) basis function expansions (see (2) and (3)). The objective of MPMR is to find a model which maximizes the worst case probability that future data is predicted within  $\pm\epsilon$  accurate. Also, we introduced a general approach to reduce any regression problem to a binary classification problem by shifting the dependent variable both by  $+\epsilon$  and by  $-\epsilon$ . In this paper, we use MPMC as the underlying classifier which gives a distribution-free probability bound, and therefore a regression algorithm that can estimate the probability of making  $\pm\epsilon$  correct predictions. We also analyse the tightness of the MPMR bound and relate it to the well known two-sided Chebyshev inequality. Remarkably, the resulting MPMR is equivalent to the method of least squares, which is a standard technique in statistical analysis. One consequence of our work is that it is now possible to calculate distribution-free confidence intervals for least squares solu-

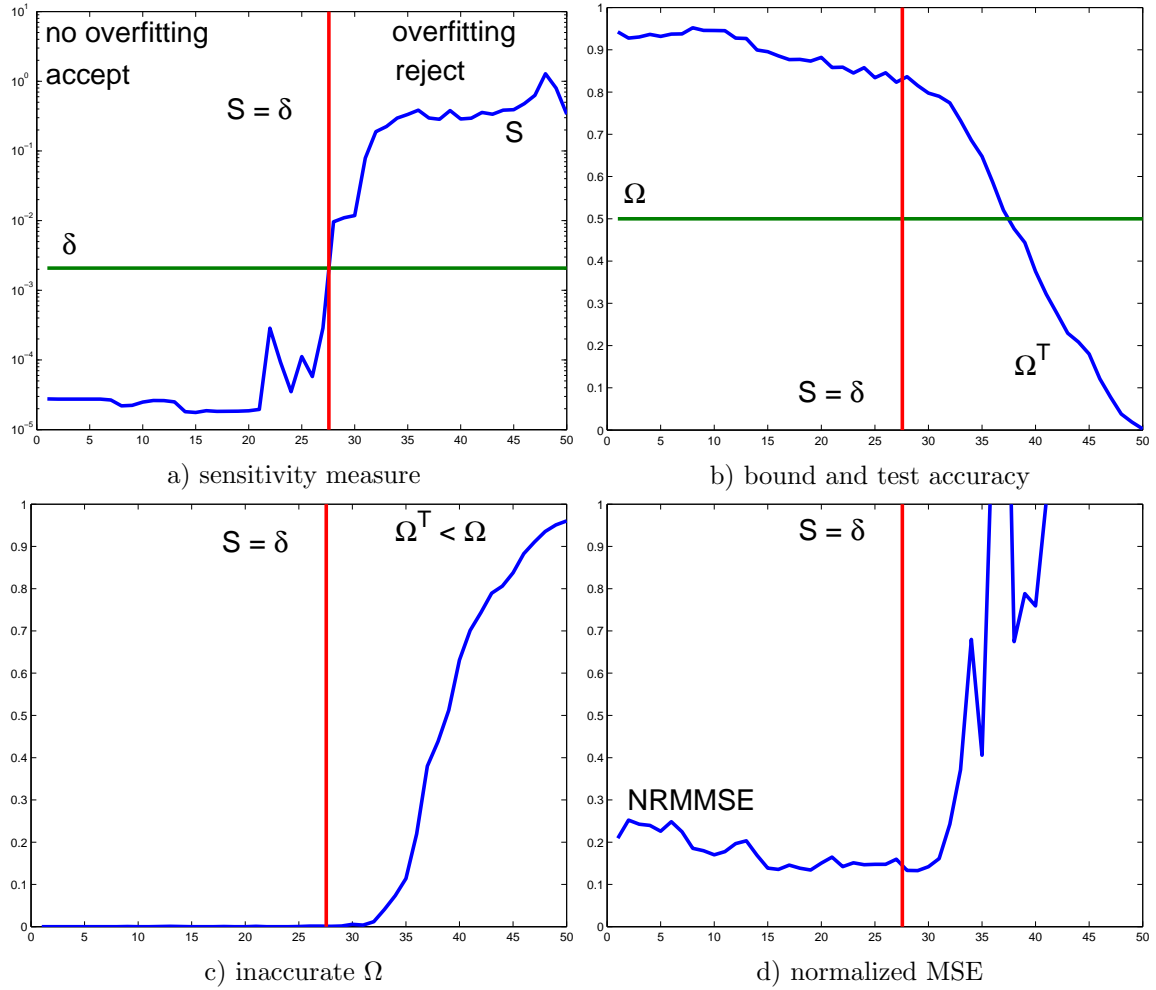


Figure 6: Boston Housing Results. Plot a) shows where our sensitivity measure  $S$  crosses the threshold  $\delta$  (which we set to  $\delta := \gamma = \frac{1}{N_{train}}$  in all our experiments). Models on the left with  $S < \delta$  are considered robust and models on the right are considered to be too sensitive and to overfit the training data. Plot b) shows that for robust models  $\Omega^T > \Omega$  holds on average. Plot c) contains the percentage of models where the bound did not hold and plot d) shows the normalized mean squared error on the test data. All results were averaged over 100 runs and the kernels being used were  $k(u, v) = \exp(-1.25^t 10^{-5} \|u - v\|_2^2)$  for  $t = 1, 2, \dots, 50$ .

tions (formerly this was only feasible under Gaussian or other specific, often unrealistic, distributional assumptions).

For nonlinear problems, we use Mercer kernels as basis functions for our regression model. However, our formulation is not limited to Mercer kernels, but works with arbitrary basis functions. Typically, there are kernel parameters (e.g. the width parameter  $\kappa$  of the

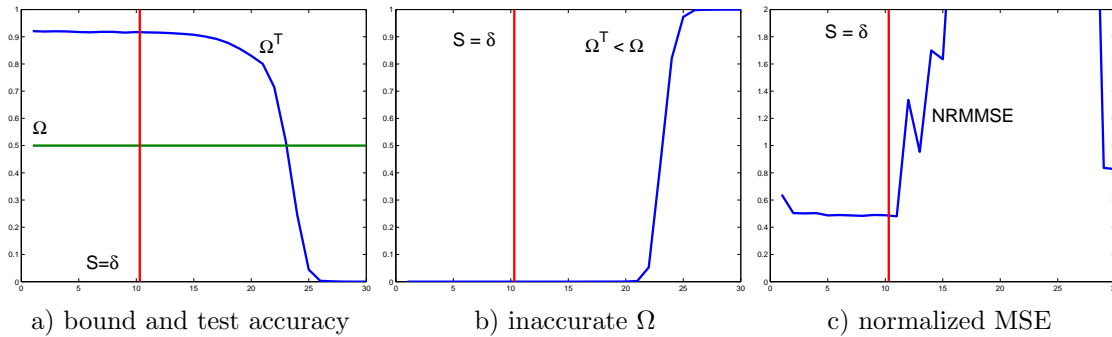


Figure 7: Abalone Results. The plots are similar to the ones for the Boston housing and Abalone data: The bound  $\Omega$  is accurate for all the models where our sensitivity measure was  $S < \delta$  and becomes inaccurate only when  $S > \delta$ . All results were averaged over 50 runs and the kernels being used were  $k(u, v) = \exp(-2.25^t 10^{-9} \|u - v\|_2^2)$  for  $t = 1, 2, \dots, 30$ .

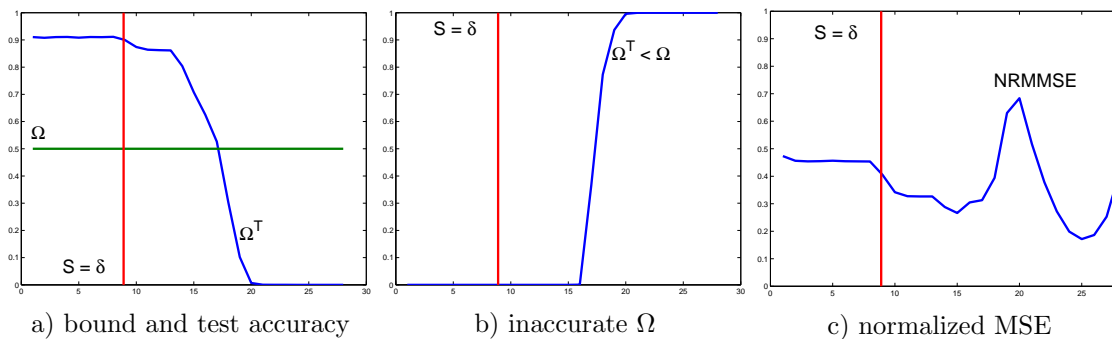


Figure 8: Kinematic Data Results. The plots are similar to the ones for the Boston housing data: The bound  $\Omega$  is accurate for all the models where our sensitivity measure was  $S < \delta$ . In plot c), we observed that the normalized mean squared error is low for some (but not all) of the models that would have been rejected by our sensitivity measure  $S$ . All results were averaged over 50 runs and the kernels being used were  $k(u, v) = \exp(-1.5^t 10^{-5} \|u - v\|_2^2)$  for  $t = 1, 2, \dots, 30$ .

Gaussian kernel) that need to be adjusted properly to obtain good regression models. A common technique to find these parameters is to do  $N$ -fold cross validation, which can be computationally expensive. Our approach, however, does not involve cross validation but is based on a sensitivity measure  $S$ .  $S$  is calculated by sampling parts of the training data and looking at the effect on  $\Omega$ —the objective of MPMR. This is computationally more efficient for  $N > 2$ , since  $N$ -fold cross validation builds  $N$  models whereas the  $S$  approach only builds two models. Experiments with real world data sets have shown that the assumptions underlying the sensitivity measure  $S$  are valid (see Appendix B) and that  $S$  works well for filtering out models that overfit the data, and consequently violate the property of  $\Omega$  as a

true lower bound. Further evidence was given showing that  $S$  finds optimal parameters for nonlinear kernel models, and therefore is suitable for model selection.

The current implementation of MPMR requires solving an  $N \times N$  linear system of equations (complexity  $\mathcal{O}(N^3)$ ) and therefore is prohibitive for large  $N$  (where  $N$  is the number of training examples). As a result we are currently working on sparse versions of MPMR. The approach we are taken is similar to the one for MPM classification that was recently published (Strohmann et al., 2004).

Also, it is well known that in real world domains data is often highly structured (e.g. in forms of clusters). Rather than giving one global bound on the overall probability of being within  $\epsilon$  correct we are exploring local MPMR models that have region specific probability bounds.

Finally, we note that for  $N \rightarrow \infty$  MPMR converges to the true  $\Omega$  (and the optimal model) since the estimates of the mean and covariance matrix become more and more accurate as the number of training examples ( $N$ ) increases. Therefore, one interesting open question is the rate of convergence for  $\Omega$  as  $N$  increases, which should be directly related to the rate of convergence for the mean and covariance estimates.

A Matlab implementation of the MPMR algorithm can be obtained from:  
<http://cse1.cs.colorado.edu/~strohman/papers.html>

## 8. Acknowledgements

This work has been partially supported by NFS grant CMS-0430593. We are grateful to Mark Gibbs for providing an implementation to build regression models with Gaussian processes. We also thank Chih-Chung Chang and Chih-Jen Lin for support vector machines regression code.

## 9. Appendix A

**Proof of Theorem 1.** The least squares problem can be formulated as the following linear system (see for example Wackerly et al., 2002, chap. 11):

$$\mathbf{X}^T \mathbf{X} \beta = \mathbf{X}^T \mathbf{Y} \tag{32}$$

where

$$\mathbf{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \cdots & \cdots & \cdots & \cdots \\ 1 & x_{N1} & \cdots & x_{Nd} \end{pmatrix} \quad \mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_N \end{pmatrix}$$

If we write out (32) we obtain (from now on all sums are from  $i = 1$  to  $N$ ):

$$\begin{pmatrix} N & \sum x_{i1} & \cdots & \sum x_{id} \\ \sum x_{i1} & \sum x_{i1}^2 & \cdots & \sum x_{i1}x_{id} \\ \cdots & \cdots & \cdots & \cdots \\ \sum x_{ij} & \cdots & \sum x_{ij}x_{ik} & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \sum x_{id} & \sum x_{id}x_{i1} & \cdots & \sum x_{id}^2 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \cdots \\ \beta_j \\ \cdots \\ \beta_d \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum x_{i1}y_i \\ \cdots \\ \sum x_{ij}y_i \\ \cdots \\ \sum x_{id}y_i \end{pmatrix}$$

Now we write all the sums in terms of means and covariances:

$$\begin{aligned}
 \sum y_i &= N\bar{y} \\
 \sum x_{ij} &= N\bar{x}_j & j = 1, \dots, d \\
 \sum x_{ij}y_i &= (N-1)\sigma_{x_j y} + N\bar{x}_j\bar{y} & j = 1, \dots, d \\
 \sum x_{ij}x_{ik} &= (N-1)\sigma_{x_j x_k} + N\bar{x}_j\bar{x}_k & j = 1, \dots, d; k = 1, \dots, d
 \end{aligned}$$

where the last two lines are obtained by noting that:

$$\begin{aligned}
 \sigma_{x_j x_k} &= \frac{1}{N-1} \sum (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k) \\
 \Leftrightarrow \sigma_{x_j x_k} &= \frac{1}{N-1} (\sum x_{ij}x_{ik} - \sum x_{ij}\bar{x}_k - \sum x_{ik}\bar{x}_j + \sum \bar{x}_j\bar{x}_k) \\
 \Leftrightarrow \sigma_{x_j x_k} &= \frac{1}{N-1} (\sum x_{ij}x_{ik} - N\bar{x}_j\bar{x}_k - N\bar{x}_j\bar{x}_k + N\bar{x}_j\bar{x}_k) \\
 \Leftrightarrow (N-1)\sigma_{x_j x_k} &= \sum x_{ij}x_{ik} - N\bar{x}_j\bar{x}_k \\
 \Leftrightarrow \sum x_{ij}x_{ik} &= (N-1)\sigma_{x_j x_k} + N\bar{x}_j\bar{x}_k
 \end{aligned}$$

If we plug these expressions into the linear system above we have:

$$\begin{aligned}
 &\begin{pmatrix} N & N\bar{x}_1 & \dots & N\bar{x}_d \\ N\bar{x}_1 & (N-1)\sigma_{x_1 x_1} + N\bar{x}_1\bar{x}_1 & \dots & (N-1)\sigma_{x_1 x_d} + N\bar{x}_1\bar{x}_d \\ \dots & \dots & (N-1)\sigma_{x_j x_k} + N\bar{x}_j\bar{x}_k & \dots \\ N\bar{x}_d & (N-1)\sigma_{x_d x_1} + N\bar{x}_d\bar{x}_1 & \dots & (N-1)\sigma_{x_d x_d} + N\bar{x}_d\bar{x}_d \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \dots \\ \beta_d \end{pmatrix} \\
 &= \begin{pmatrix} N\bar{y} \\ (N-1)\sigma_{x_1 y} + N\bar{x}_1\bar{y} \\ \dots \\ (N-1)\sigma_{x_d y} + N\bar{x}_d\bar{y} \end{pmatrix}
 \end{aligned}$$

If we look at the first equation we retrieve the familiar formula for  $\beta_0$ :

$$\begin{aligned}
 N\beta_0 + \sum_{j=1}^d N\bar{x}_j\beta_j &= N\bar{y} \\
 \Leftrightarrow \beta_0 &= \bar{y} - \sum_{j=1}^d \beta_j\bar{x}_j
 \end{aligned}$$

Now look at the  $j+1$ th ( $j = 1, \dots, d$ ) equation which has  $x_j$  as the common variable:

$$\begin{aligned}
 &N\bar{x}_j\beta_0 + \sum_{k=1}^d \beta_k[(N-1)\sigma_{x_j x_k} + N\bar{x}_j\bar{x}_k] = (N-1)\sigma_{x_j y} + N\bar{x}_j\bar{y} \\
 \Leftrightarrow &N\bar{x}_j(\bar{y} - \sum_{k=1}^d \beta_k\bar{x}_k) + N\sum_{k=1}^d \beta_k\bar{x}_j\bar{x}_k + (N-1)\sum_{k=1}^d \beta_k\sigma_{x_j x_k} = (N-1)\sigma_{x_j y} + N\bar{x}_j\bar{y} \\
 \Leftrightarrow &N\bar{x}_j\bar{y} + (N-1)\sum_{k=1}^d \beta_k\sigma_{x_j x_k} = (N-1)\sigma_{x_j y} + N\bar{x}_j\bar{y} \\
 \Leftrightarrow &(N-1)\sum_{k=1}^d \beta_k\sigma_{x_j x_k} = (N-1)\sigma_{x_j y} \\
 \Leftrightarrow &\sum_{k=1}^d \beta_k\sigma_{x_j x_k} = \sigma_{x_j y}
 \end{aligned}$$

The last expression matches exactly the  $j$ th equation of (15). Since this is true for all  $j = 1, \dots, d$  we have shown that (15) / (16) describe the same system that appears for least squares problems.

□

## 10. Appendix B

In this section we will give experimental evidence that the assumptions stated in (28) do hold in practice. To test the assumptions we need to “know” the true covariance matrix  $\Sigma^*$ . We will simulate this knowledge by using *all* data points of the data set to compute  $\Sigma^*$  and using only half of the data to compute the covariance matrix  $\Sigma$ . The following plots illustrate the validity of the assumptions for different kernels and different values for  $\gamma$ :

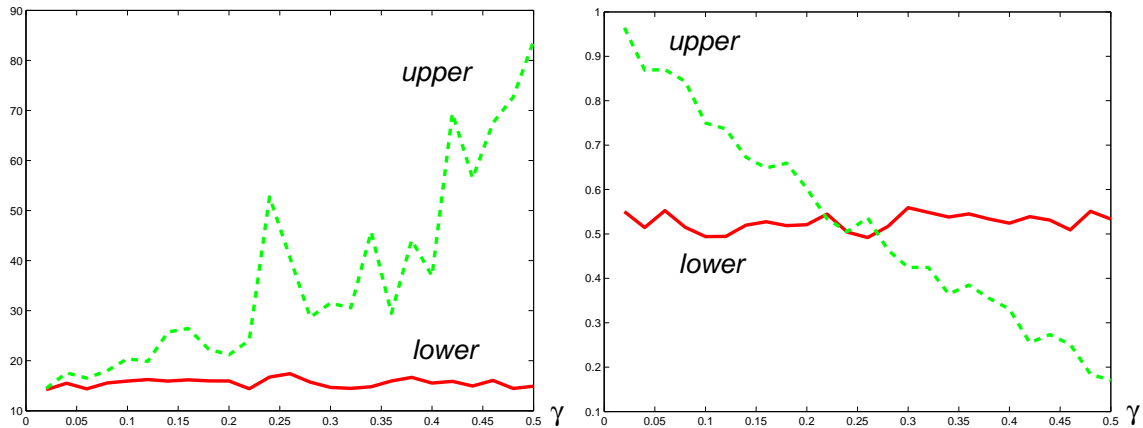


Figure 9: Boston Housing data with Gaussian Kernel ( $s=0.0001$  - no overfitting). The left plot shows  $lower = E[\mathbf{a}^T \Sigma_* \mathbf{a}]$  and  $upper = E[\mathbf{a}_\gamma^T \Sigma_* \mathbf{a}_\gamma]$  for different fractions  $\gamma$  of points left out. The right plot shows  $lower = E[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}^T \Sigma_* \mathbf{a}}]$  and  $upper = E[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}]$ . All results were averaged over 50 runs. In both plots the assumptions are valid (as indicated by the right plot we can only expect that the second assumption  $E[\frac{\mathbf{a}_*^T \Sigma_* \mathbf{a}_*}{\mathbf{a}^T \Sigma_* \mathbf{a}}] \leq E[\frac{\mathbf{a}^T \Sigma \mathbf{a}}{\mathbf{a}_\gamma^T \Sigma \mathbf{a}_\gamma}]$  holds for  $\gamma$  small).

## References

- C. C. Chang and C. J. Lin. Libsvm – a library for support vector machines, 2003. URL <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.
- M. N. Gibbs. *Bayesian Gaussian Processes for Regression and Classification*. PhD thesis, University of Cambridge, 1997a.
- M. N. Gibbs. Tpros software: Regression using gaussian processes, 1997b. URL <http://www.inference.phy.cam.ac.uk/mng10/GP/>.
- G. R. G. Lanckriet, L. E. Ghaoui, C. Bhattacharyya, and M. I. Jordan. Minimax probability machine. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002a. MIT Press.
- G. R. G. Lanckriet, L. E. Ghaoui, C. Bhattacharyya, and M. I. Jordan. A robust minimax approach to classification. *Journal of Machine Learning Research*, 3:555–582, 2002b.

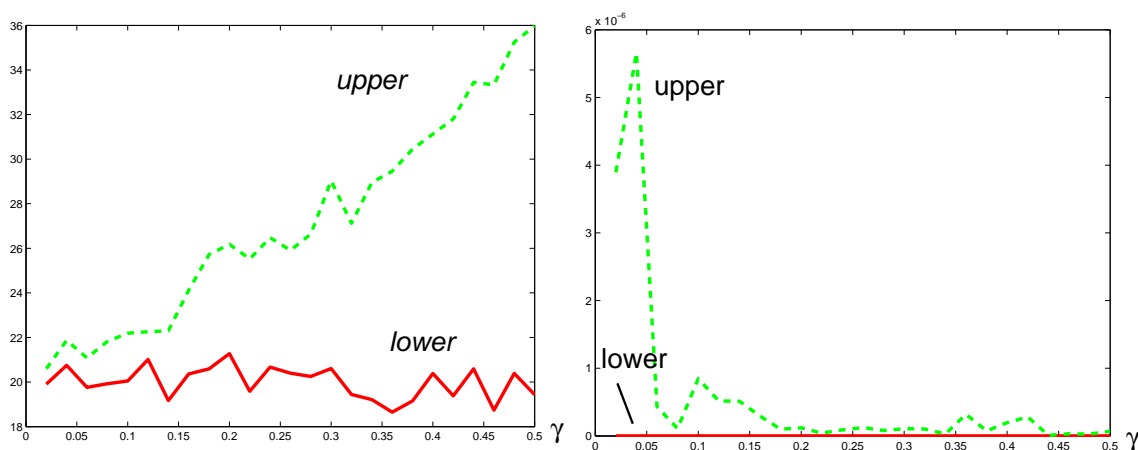


Figure 10: The above experiment repeated with a different Gaussian Kernel ( $s=1$  - overfitting). Due to overfitting the ratios of the right plot are close to zero. But again, both the assumptions made in (28) hold.

- D. Mackay. Gaussian processes – a replacement for supervised neural networks?, 1997. Lecture notes for a tutorial at NIPS.
- A. W. Marshall and I. Olkin. Multivariate chebyshev inequalities. *Annals of Mathematical Statistics*, 31(4):1001–1014, 1960.
- I. Popescu and D. Bertsimas. Optimal inequalities in probability theory: A convex optimization approach. Technical Report TM62, INSEAD, Dept. Math. O.R., Cambridge, Mass, 2001.
- C. E. Rasmussen. *Evaluation of Gaussian Processes and other Methods for Non-Linear Regression*. PhD thesis, University of Toronto, 1996.
- A. K. Jain S. J. Raudys. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(3):252–264, 1991.
- B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- A. Smola and B. Schölkopf. A tutorial on support vector regression. Technical Report NC2-TR-1998-030, Royal Holloway College, 1998.
- T. R. Strohmman, A. Belitski, G. Z. Grudic, and D. DeCoste. Sparse greedy minimax probability machine classification. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.
- T. R. Strohmman, Andrei Belitski, G.Z. Grudic, and Dennis DeCoste. Sparse greedy minimax probability machine classification. Technical Report CU-CS-956-03, University of Colorado, Boulder, 2003.

- T. R. Strohmann and G. Z. Grudic. A formulation for minimax probability machine regression. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.
- M. Tipping. The relevance vector machine. In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, Cambridge, MA, 2000. MIT Press.
- D. Wackerly, W. Mendenhall, and R. Scheaffer. *Mathematical Statistics with Applications, Sixth Edition*. Duxbury Press, Pacific Grove, CA, 2002.
- Christopher K. I. Williams and Carl Edward Rasmussen. Gaussian processes for regression. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, Cambridge, MA, 1995. MIT Press.