COMPUTATION OF THE REGRESSION KERNEL MATRIX USING SEMIDEFINITE PROGRAMMING*

T. B. Trafalis, A. M. Malyscheff
School of Industrial Engineering University of Oklahoma, Norman, USA
e-mail: ttrafalis@ou.edu, malyscheff@gmail.com

Оптимальное ядро матрицы регрессии рассчитано методом полуопределенного программирования с использованием трех базисных матриц. В работе представлены предварительные результаты, использующие стандартные реперные данные, для которых выявлены оптимальные параметры линейной комбинации трех базисных матриц ядра.

Introduction

The problem of identifying an optimal kernel for a specific class of data has recently found increasing attention in the machine learning community. Chapelle et al. [6] have investigated the selection of optimal parameters using a traditional steepest descent method. Their approach locates a local minimum in the space of parameters. Cristianini et al. [8] have suggested an approach which finds the kernel matrix that best describes the labels of the training set (kernel target alignment). Lanckriet et al. [9] employed ideas from semidefinite programming for computing the optimal kernel matrix for pattern classification problems. Bach, Lanckriet and Jordan [1] use sequential minimal optimization techniques to improve computational efficiency for solving support vector machine classification problems, which are based on a combination of kernel matrices. Trafalis and Malyscheff [16] computed the optimal kernel matrix for regression analysis problems using semidefinite programming excluding basis matrices. In this paper we extend these ideas computing the optimal parameters for a linear combination of three basis regression kernel matrices using semidefinite programming techniques [10, 11, 17]. We illustrate our findings with some examples and apply them to standard benchmark data.

The paper is organized as follows: in section 1 we will provide a brief introduction to support vector machine learning and describe the primal and dual versions for regression analysis problems. Based on the dual formulation we will then compute the dual of the dual support vector regression problem, since this formulation has some computational advantages for our purposes. Section 2 introduces the semidefinite programming framework and incorporates the results from the previous section resulting in two formulations, which were used for experimentation. In section 3 we will compute a few simple examples illustrating our formulation and then present results using standard benchmark data.

*This research has been supported partially by the National Science Foundation, NSF Grant ECS-0099378.

1. Support Vector Machine Learning for Regression Analysis

Support vector machine (SVM) learning and other kernel-based learning algorithms can be implemented either as a classification or regression analysis problem. This discussion will focus on the regression analysis environment, for further information on the subject, in particular on support vector machines in classification, we refer the reader to the texts [5, 7, 13, 18]. Regression analysis problems focus on the computation of a linear scalar based on one or more input values (attributes). Mathematically, let the training data consist of l vectors $\mathbf{x}_j \in \Re^d$ with an a priori known output value $y_j \in \Re$, where $j = 1, ..., l$. Hence, the training set can be written as $T = \{(\mathbf{x}_j, y_j)_{j=1}^{l}\} \subset \Re^{d+1}$. Vapnik [19] has shown that in the case of linear support vector machine regression the following primal optimization problem must be solved:

$$\begin{aligned} (P) \qquad & \min \frac{1}{2} \|\mathbf{w}\|^2, \\ & \text{subject to} \\ y_j - \mathbf{w}^T \mathbf{x}_j - b \leq \epsilon \quad & \forall j = 1, ..., l, \qquad (\lambda_j), \\ \mathbf{w}^T \mathbf{x}_j + b - y_j \leq \epsilon \quad & \forall j = 1, ..., l, \qquad (\lambda_j^*), \end{aligned} \tag{1}$$

where $\mathbf{w} \in \Re^d$ is the slope of the regression function and $b \in \Re$ the offset with respect to the origin. Note that for d-dimensional input data $\mathbf{w} \in \Re^d$. The parameter ϵ can be interpreted as the precision that is required from the regression function. Geometrically, it creates a tube of width 2ϵ around the regression function, within which all measured data samples $(\mathbf{x}_j, y_j)$ must be contained.

Let λ_j be the Lagrangian multiplier corresponding to the first set of constraints and λ_j^* be the Lagrangian multiplier corresponding to the second set ($j = 1, ..., l$). Define the vectors $\tilde{\mathbf{\Lambda}}^T = \mathbf{\Lambda}^T - \mathbf{\Lambda}^{*T} = (\lambda_1 - \lambda_1^*, \lambda_2 - \lambda_2^*, ..., \lambda_l - \lambda_l^*)$, $\mathbf{1}^T = (1, 1, ..., 1)$, and $\mathbf{y}^T = (y_1, y_2, ..., y_l)$ as well as the matrix $K_{ij} = x_i^T x_j$ in the linear case and $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ in the general case, where $k(\mathbf{x}_i, \mathbf{x}_j)$ is the kernel function. Popular kernel functions include a d-degree polynomial kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i^T \mathbf{x}_j + 1\right)^d$ or a radial-basis function kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-0.5 (\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j) / \sigma^2\right)$. The dual problem can then be written in closed form as:

$$\begin{aligned} (D) \qquad W(\mathbf{K}_{\text{tr}}) &= \max -\frac{1}{2} \tilde{\mathbf{\Lambda}}^T \mathbf{K}_{\text{tr}} \tilde{\mathbf{\Lambda}} - \epsilon \left(\mathbf{\Lambda}^T \mathbf{1} + \mathbf{\Lambda}^{*T} \mathbf{1}\right) + {\mathbf{y}_{\text{tr}}}^T \tilde{\mathbf{\Lambda}}, \\ & \text{subject to} \\ \tilde{\mathbf{\Lambda}}^T \mathbf{1} &= 0 \qquad (\pi), \\ (\mathbf{\Lambda})_j &\geq 0 \quad \forall j = 1, ..., l \qquad (\mathbf{\Gamma})_j, \\ (\mathbf{\Lambda}^*)_j &\geq 0 \quad \forall j = 1, ..., l \qquad (\mathbf{\Gamma}^*)_j. \end{aligned} \tag{2}$$

By writing $\mathbf{K}_{\text{tr}}$ we emphasize that for the solution of this problem we entirely rely on the training set excluding kernel products from the test set for the computation of $W(\mathbf{K}_{\text{tr}})$. We will elaborate on $\mathbf{K}_{\text{tr}}$ in section 2. We also denote the vector of the training labels by $\mathbf{y}_{\text{tr}}$, again, in order to show that this variable is solely based on training data. Note that the regression function can be expressed as

$$f(\mathbf{x}_j) = \sum_{i=1}^{l} \left(\tilde{\mathbf{\Lambda}}\right)_i k(\mathbf{x}_i, \mathbf{x}_j) + b \quad \forall j = 1, ..., l \tag{3}$$

for the training data. In this paper we identify during optimization also the $k(\mathbf{x}_i, \mathbf{x}_j)$ for the test data, thus, the prediction regression function for the test set can be found from

$$f(\mathbf{x}_j) = \sum_{i=1}^{l} \left(\tilde{\mathbf{\Lambda}}\right)_i k(\mathbf{x}_i, \mathbf{x}_j) + b \quad \forall j = l+1, ..., l+n_{\text{ts}}, \tag{4}$$

where n_{ts} indicates the number of samples in the test set. The bias b can be computed by solving the complementary slackness conditions [12, 13]. We will further discuss b in section 2.

In the next step we will compute the dual problem of the dual support vector regression formulation in (2), as this will facilitate the computational analysis. Using the variables $\mathbf{\Gamma} = (\gamma_1, ..., \gamma_l)$, $\mathbf{\Gamma}^* = (\gamma_1^*, ..., \gamma_l^*)$, $\mathbf{\Lambda} = (\lambda_1, ..., \lambda_l)$, $\mathbf{\Lambda}^* = (\lambda_1^*, ..., \lambda_l^*)$, and $\tilde{\mathbf{\Lambda}} = \mathbf{\Lambda} - \mathbf{\Lambda}^*$ the Lagrangian of the (dual) support vector machine regression problem can be written as:

$$\begin{aligned} L\left(\lambda_j, \lambda_j^*, \pi, \gamma_j, \gamma_j^*\right) &= \mathbf{y_{tr}}^T \mathbf{\Lambda} - \mathbf{y_{tr}}^T \mathbf{\Lambda}^* - \frac{1}{2}(\mathbf{\Lambda} - \mathbf{\Lambda}^*)^T \mathbf{K}_{\text{tr}} (\mathbf{\Lambda} - \mathbf{\Lambda}^*) - \epsilon (\mathbf{\Lambda} + \mathbf{\Lambda}^*)^T \mathbf{1} + \\ &\quad + \mathbf{\Gamma}^T \mathbf{\Lambda} + (\mathbf{\Gamma}^*)^T \mathbf{\Lambda}^* + \pi (\mathbf{\Lambda} - \mathbf{\Lambda}^*)^T \mathbf{1}. \end{aligned} \tag{5}$$

From duality theory [2, 4] we know:

$$\begin{aligned} W(\mathbf{K}_{\text{tr}}) &= \max_{\lambda_j \geq 0, \lambda_j^* \geq 0} \left\{ \min_{\gamma_j \geq 0, \gamma_j^* \geq 0, \pi} \left\{ L\left(\lambda_j, \lambda_j^*, \gamma_j, \gamma_j^*, \pi\right) \right\} \right\}, \\ W(\mathbf{K}_{\text{tr}}) &= \min_{\gamma_j \geq 0, \gamma_j^* \geq 0, \pi} \left\{ \max_{\lambda_j \geq 0, \lambda_j^* \geq 0} \left\{ L\left(\lambda_j, \lambda_j^*, \gamma_j, \gamma_j^*, \pi\right) \right\} \right\}. \end{aligned} \tag{6}$$

Computing the gradients $\nabla_{\mathbf{\Lambda}} L$ and $\nabla_{\mathbf{\Lambda}^*} L$ results in:

$$\nabla_{\mathbf{\Lambda}} L = \mathbf{y}_{\text{tr}} - \mathbf{K}_{\text{tr}} (\mathbf{\Lambda} - \mathbf{\Lambda}^*) - \epsilon \mathbf{1} + \mathbf{\Gamma} + \pi \mathbf{1} = 0; \tag{7}$$

$$\nabla_{\mathbf{\Lambda}^*} L = -\mathbf{y}_{\text{tr}} + \mathbf{K}_{\text{tr}} (\mathbf{\Lambda} - \mathbf{\Lambda}^*) - \epsilon \mathbf{1} + \mathbf{\Gamma}^* - \pi \mathbf{1} = 0. \tag{8}$$

Upon combining equations (7) and (8) one finds:

$$\mathbf{\Gamma} + \mathbf{\Gamma}^* = 2\epsilon \mathbf{1}. \tag{9}$$

Since $\mathbf{K}_{\text{tr}}$ is positive definite, expression (7) can be solved for $\mathbf{\Lambda}$:

$$\mathbf{\Lambda} = \mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) + \mathbf{\Lambda}^*. \tag{10}$$

We can recompute the Lagrangian by using the results from (9) and (10):

$$\begin{aligned} &L(\lambda_j, \lambda_j^*, \pi, \gamma_j, \gamma_j^*) = \\ &= \mathbf{y_{tr}}^T \mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) - \epsilon \mathbf{1}^T \left[\mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) + 2\mathbf{\Lambda}^* \right] + \\ &+ \pi \mathbf{1}^T \mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) + \mathbf{\Gamma}^{\mathbf{T}} \left[\mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) + \mathbf{\Lambda}^* \right] + \\ &+ (\mathbf{\Gamma}^*)^T \mathbf{\Lambda}^* - \frac{1}{2} \left[\mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) \right]^T \mathbf{K}_{\text{tr}} \mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}) \end{aligned}$$

and after simplifying we obtain:

$$L\left(\lambda_{j,\text{opt}}, \lambda_{j,\text{opt}}^*, \pi, \gamma_j, \gamma_j^*\right) = \frac{1}{2} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma})^T \mathbf{K}_{\text{tr}}^{-1} (\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}). \tag{11}$$

Therefore, the "dual of the dual" for support vector machine regression reduces to:

$$\begin{aligned} W\left(\mathbf{K}_{\text{tr}}, \pi_{\text{opt}}, \mathbf{\Gamma}_{\text{opt}}\right) &= \min \frac{1}{2}\left(\mathbf{y}_{\text{tr}} - \epsilon\mathbf{1} + \pi\mathbf{1} + \mathbf{\Gamma}\right)^T \mathbf{K}_{\text{tr}}^{-1}\left(\mathbf{y}_{\text{tr}} - \epsilon\mathbf{1} + \pi\mathbf{1} + \mathbf{\Gamma}\right) \\ &\quad \text{subject to} \\ &\quad 2\epsilon\mathbf{1} - \mathbf{\Gamma} \geq 0, \\ &\quad \mathbf{\Gamma} \geq 0, \\ &\quad \pi \text{ unrestricted.} \end{aligned} \tag{12}$$

Note that the regression parameters can be obtained from $\tilde{\mathbf{\Lambda}} = \mathbf{\Lambda} - \mathbf{\Lambda}^*$. Taking into account equation (10) one finds for the regression parameters:

$$\tilde{\mathbf{\Lambda}} = \mathbf{\Lambda} - \mathbf{\Lambda}^* = \mathbf{K}_{\text{tr}}^{-1}\left(\mathbf{y}_{\text{tr}} - \epsilon\mathbf{1} + \pi\mathbf{1} + \mathbf{\Gamma}\right). \tag{13}$$

2. Formulation using Semidefinite Programming

In the previous section we briefly discussed support vector machine learning in regression analysis and presented the dual of the dual formulation. We will now introduce the semidefinite programming (SDP) framework in which problem (12) will be embedded.

Let us begin by first decomposing the kernel matrix $\mathbf{K}$. This matrix contains mappings of scalar products of the input data for both the training and the test set. Since a part of the analysis extracts information solely contained in the training set, while other parts will require information of test set input data, we will write the kernel matrix $\mathbf{K}$ as:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{\text{tr}} & \mathbf{K}_{\text{tr},t} \\ \mathbf{K}_{\text{tr},t}^T & \mathbf{K}_t \end{bmatrix}. \tag{14}$$

The matrix $\mathbf{K}_{\text{tr}}$ reflects information of the training data, $\mathbf{K}_{\text{tr},t}$ describes mappings of scalar products between training and test set, while $\mathbf{K}_t$ represents mappings solely of test set input vectors. The expression in equation (2) for example operates only on scalar product mappings $\mathbf{K}_{\text{tr}}$ from the training inputs:

$$\begin{aligned} W\left(\mathbf{K}_{\text{tr}}\right) = \max -\frac{1}{2}\left(\mathbf{\Lambda} - \mathbf{\Lambda}^*\right)^T \mathbf{K}_{\text{tr}}\left(\mathbf{\Lambda} - \mathbf{\Lambda}^*\right) - \epsilon\left(\mathbf{\Lambda}^T\mathbf{1} + \mathbf{\Lambda}^{*T}\mathbf{1}\right) + \mathbf{y}_{\text{tr}}{}^T\left(\mathbf{\Lambda} - \mathbf{\Lambda}^*\right) : \\ \left(\mathbf{\Lambda} - \mathbf{\Lambda}^*\right)^T\mathbf{1} = 0, \mathbf{\Lambda} \geq 0, \mathbf{\Lambda}^* \geq 0. \end{aligned}$$

It can be observed that $W\left(\mathbf{K}_{\text{tr}}\right)$ is convex in $\mathbf{K}_{\text{tr}}$. Moreover, since regression analysis problems can be interpreted as a classification problem [3], we can use the concept of the margin and apply it in this context. Thus, since $W\left(\mathbf{K}_{\text{tr}}\right)$ is the inverse of the margin for classification problems, we can follow the same reasoning minimizing $W\left(\mathbf{K}_{\text{tr}}\right)$ under the assumption that $\mathbf{K} \succeq 0$ and $\text{trace}\left(\mathbf{K}\right) = \text{const}$ [8]. Thus, we require semidefiniteness for all data samples, while optimality is enforced on the training data:

$$\begin{aligned} &\min W\left(\mathbf{K}_{\text{tr}}\right) \\ &\quad \text{subject to} \\ &\quad \mathbf{K} \succeq 0, \\ &\quad \text{trace}\left(\mathbf{K}\right) = c. \end{aligned} \tag{15}$$

By introducing the variable t the above problem can be formulated in terms of $\mathbf{K}_{\text{tr}}$ and $\mathbf{K}$ as follows:

$$\begin{aligned} & \min t \\ & \text{subject to} \\ \mathbf{K} & \succeq 0, \\ t & \geq W\left(\mathbf{K}_{\text{tr}}\right), \\ \text{trace}\left(\mathbf{K}\right) & = c. \end{aligned} \tag{16}$$

Moreover, considering both equations (12) and (16) one can write:

$$\begin{aligned} & \min t \\ & \text{subject to} \\ \mathbf{K} & \succeq 0, \\ t & \geq \min\left\{\frac{1}{2}\left(\mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\right)^{T}\mathbf{K}_{\text{tr}}^{-1}\left(\mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\right) : 2\epsilon\mathbf{1}-\mathbf{\Gamma}\geq 0, \mathbf{\Gamma}\geq\mathbf{0}\right\}, \\ \text{trace}\left(\mathbf{K}\right) & = c. \end{aligned} \tag{17}$$

The constraint imposed on $\mathbf{\Gamma}$ can be shifted from the subproblem to the global problem:

$$\begin{aligned} & \min t \\ & \text{subject to} \\ \mathbf{K} & \succeq 0, \\ t & \geq \min\left\{\frac{1}{2}\left(\mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\right)^{T}\mathbf{K}_{\text{tr}}^{-1}\left(\mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\right)\right\}, \\ 2\epsilon\mathbf{1}-\mathbf{\Gamma} & \geq 0, \\ \mathbf{\Gamma} & \geq 0, \\ \text{trace}\left(\mathbf{K}\right) & = c. \end{aligned} \tag{18}$$

From Schur's complement we know that for the symmetric matrix

$$\mathbf{X}=\mathbf{X}^{T}=\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{T} & \mathbf{C}\end{bmatrix} \tag{19}$$

holds that if $\mathbf{A}\succ 0$, then $\mathbf{X}\succeq 0$, if and only if $\mathbf{S}=\mathbf{C}-\mathbf{B}^{T}\mathbf{A}^{-1}\mathbf{B}\succeq 0$.

Using this additional information equation (18) can be written as:

$$\begin{aligned} & \min_{\mathbf{K},t,\pi,\mathbf{\Gamma}} t \\ & \text{subject to} \\ \text{trace}\left(\mathbf{K}\right) & = c, \\ \mathbf{K} & \succeq 0, \\ \begin{bmatrix}\mathbf{K}_{\text{tr}} & \mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\\ \left(\mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\right)^{T} & 2t\end{bmatrix} & \succeq 0, \\ 2\epsilon\mathbf{1}-\mathbf{\Gamma} & \geq 0, \\ \mathbf{\Gamma} & \geq 0. \end{aligned} \tag{20}$$

The matrices $\mathbf{K}_{\text{tr}}$ and $\mathbf{K}$ contain as elements input vector scalar products or mappings thereof. Various different mappings exist (polynomial, radial-basis function) and one aspect of research in support vector machine learning addresses the issue of selecting an efficient parameter for these mappings. Here, these parameters will be preselected for three basis kernel matrices, which are subsequently optimally combined using a set of multipliers $\mu_i \in \Re$. Mathematically, consider

$$\mathbf{K} = \mu_1 \mathbf{K}_1 + \mu_2 \mathbf{K}_2 + \mu_3 \mathbf{K}_3. \tag{21}$$

Here, $\mathbf{K}_1$ describes a polynomial kernel with entries

$$k_1 (\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i^T \mathbf{x}_j + 1\right)^d. \tag{22}$$

For experiments in this paper a value of $d = 2$ was selected. Next, $\mathbf{K}_2$ implements a radial-basis function kernel with entries

$$k_2 (\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-0.5 \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \left(\mathbf{x}_i - \mathbf{x}_j\right) / \sigma^2\right). \tag{23}$$

For this analysis we chose $\sigma = 0.5$. Finally, $\mathbf{K}_3$ realizes a linear kernel with entries

$$k_3 (\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j. \tag{24}$$

Taking into account the decompostion as described in equations (21)–(24) we can rewrite problem (20):

$$\begin{aligned}
&\min_{\mu_1, \mu_2, \mu_3, t, \pi, \mathbf{\Gamma}} t \\
&\text{subject to} \\
&\text{trace}\left(\mu_1 \mathbf{K}_1 + \mu_2 \mathbf{K}_2 + \mu_3 \mathbf{K}_3\right) = c, \\
&\mu_1 \mathbf{K}_1 + \mu_2 \mathbf{K}_2 + \mu_3 \mathbf{K}_3 \succeq 0, \\
&\begin{bmatrix} \mu_1 \mathbf{K}_{1,\text{tr}} + \mu_2 \mathbf{K}_{2,\text{tr}} + \mu_3 \mathbf{K}_{3,\text{tr}} & \mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma} \\ \left(\mathbf{y}_{\text{tr}} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma}\right)^T & 2t \end{bmatrix} \succeq 0, \\
&2\epsilon \mathbf{1} - \mathbf{\Gamma} \geq 0, \\
&\mathbf{\Gamma} \geq 0, \\
&\mu_1, \mu_2, \mu_3 \quad \text{free}.
\end{aligned} \tag{25}$$

The semidefinite programming problem in (25) leaves the parameters μ_i unrestricted and one set of experiments on standard benchmark data was conducted using this formulation. In addition we also formulated a kernel-based learning algorithm with the additional requirement of the μ_i to be nonnegative. This optimization problem is spelled out in (26) and benchmark

tests were also performed using this formulation.

$$\begin{array}{rcl} \min t & & \\ \mu_1,\mu_2,\mu_3,t,\pi,\mathbf{\Gamma} & & \\ \text{subject to} & & \\ \text{trace}\left(\mu_1\mathbf{K}_1+\mu_2\mathbf{K}_2+\mu_3\mathbf{K}_3\right) & = & c, \\ \mu_1\mathbf{K}_1+\mu_2\mathbf{K}_2+\mu_3\mathbf{K}_3 & \succeq & 0, \\ \left[\begin{array}{cc} \mu_1\mathbf{K}_{1,\text{tr}}+\mu_2\mathbf{K}_{2,\text{tr}}+\mu_3\mathbf{K}_{3,\text{tr}} & \mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma} \\ \left(\mathbf{y}_{\text{tr}}-\epsilon\mathbf{1}+\pi\mathbf{1}+\mathbf{\Gamma}\right)^T & 2t \end{array}\right] & \succeq & 0, \\ 2\epsilon\mathbf{1}-\mathbf{\Gamma} & \geq & 0, \\ \mathbf{\Gamma} & \geq & 0, \\ \mu_1,\mu_2,\mu_3 & \geq & 0. \end{array} \tag{26}$$

In order to compute the predicted output $f(\mathbf{x}_j)$ for both training and test set we require the $\left(\tilde{\mathbf{\Lambda}}\right)_i$, which can be computed from equations (13), (14), and (21). For the computation of the bias b in equation (3) complementary slackness imposes:

$$\gamma_j\lambda_j = 0 \quad \forall j=1,...,l, \tag{27}$$

and

$$\gamma_j^*\lambda_j^* = 0 \quad \forall j=1,...,l. \tag{28}$$

In addition, we know from (9) that $\mathbf{\Gamma}+\mathbf{\Gamma}^*=2\epsilon\mathbf{1}$. Therefore, if $\gamma_j^*=0$, we conclude that $\gamma_j\neq 0$ requiring $\lambda_j=0$. Finally, going back to the original problem in (1) we can postulate that $b=y_j-\mathbf{w}^T\mathbf{x}_j-\epsilon$, if $\lambda_j\neq 0$. A similar analysis can be conducted for $\gamma_j=0$, which yields overall:

$$\gamma_j=0 \quad \rightarrow \quad \lambda_j\neq 0 \quad \rightarrow \quad b=y_j-\mathbf{w}^T\mathbf{x}_j-\epsilon \tag{29}$$

and

$$\gamma_j^*=0 \quad \rightarrow \quad \lambda_j^*\neq 0 \quad \rightarrow \quad b=y_j-\mathbf{w}^T\mathbf{x}_j+\epsilon. \tag{30}$$

The scalar product of $\mathbf{w}^T\mathbf{x}_j$ can be replaced by $\sum_{i=1}^{l}\left(\tilde{\mathbf{\Lambda}}\right)_i k(\mathbf{x}_i,\mathbf{x}_j)$. Note that it is possible to have both, $\gamma_j\neq 0$ and $\gamma_j^*\neq 0$, resulting in data points which are located entirely inside the regression tube.

3. Computational Results

3.1. Examples

In this section we will employ the semidefinite programming problem from (25) to compute the optimal kernel matrix for several simple regression problems. For our computations we used SeDuMi 1.05 [14] and Yalmip [15].

In the first experiment consider the quadratic function $f(x)=x^2$. The training set spanned $\mathbf{x}_{\text{tr}}^T=(-2,-1,0,1,2)$ with the test set consisting of $\mathbf{x}_{\text{ts}}^T=(1.5)$. Thus, the training output

assumed the values of $\mathbf{y}_{\text{tr}}{}^T = (4, 1, 0, 1, 4)$, while the predicted output $\mathbf{y}_{\text{ts,pr}}^T$ was compared to the value $\mathbf{y}_{\text{ts}}^T = (2.25)$. For this experiment we selected $\epsilon = 0.01$ and $c = 1$ requiring thus trace $(\mathbf{K}) = 1$. Solving problem (25) for these values yields the following result for the parameters μ:

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{pmatrix} = \begin{pmatrix} 0.0222 \\ 0.0000 \\ -0.0444 \end{pmatrix}. \tag{31}$$

Keeping in mind that $\mathbf{K}_1$ corresponds to a polynomial kernel of degree 2, $\mathbf{K}_2$ to a radial-basis function kernel of degree 0.5, and $\mathbf{K}_3$ to a linear kernel the overall training matrix becomes:

$$\mathbf{K}_{\text{tr}} = \mu_1 \mathbf{K}_{1,\text{tr}} + \mu_2 \mathbf{K}_{2,\text{tr}} + \mu_3 \mathbf{K}_{3,\text{tr}} = \begin{bmatrix} 0.3773 & 0.1110 & 0.0222 & 0.1110 & 0.3773 \\ 0.1110 & 0.0444 & 0.0222 & 0.0444 & 0.1110 \\ 0.0222 & 0.0222 & 0.0222 & 0.0222 & 0.0222 \\ 0.1110 & 0.0444 & 0.0222 & 0.0444 & 0.1110 \\ 0.3773 & 0.1110 & 0.0222 & 0.1110 & 0.3773 \end{bmatrix}. \tag{32}$$

The overall matrix includes also the test set, for this simple example the test set contained only one pattern, therefore $\mathbf{K} \in \Re^{6\times 6}$ carrying the values:

$$\mathbf{K} = \begin{bmatrix} 0.3773 & 0.1110 & 0.0222 & 0.1110 & 0.3773 & 0.2219 \\ 0.1110 & 0.0444 & 0.0222 & 0.0444 & 0.1110 & 0.0721 \\ 0.0222 & 0.0222 & 0.0222 & 0.0222 & 0.0222 & 0.0222 \\ 0.1110 & 0.0444 & 0.0222 & 0.0444 & 0.1110 & 0.0721 \\ 0.3773 & 0.1110 & 0.0222 & 0.1110 & 0.3773 & 0.2219 \\ 0.2219 & 0.0721 & 0.0222 & 0.0721 & 0.2219 & 0.1345 \end{bmatrix}. \tag{33}$$

Indeed, the trace of this matrix adds up to one. Moreover, for the value of the objective function one finds $t = 22.31$. The unrestricted Lagrangian multiplier assumes a value of $\pi = -0.01$. For $\mathbf{\Gamma}$ and $\mathbf{\Gamma}^*$ the following vectors are calculated respectively:

$$\mathbf{\Gamma} = \begin{pmatrix} 0.0000 \\ 0.0150 \\ 0.0200 \\ 0.0150 \\ 0.0000 \end{pmatrix} \text{ and } \mathbf{\Gamma}^* = \begin{pmatrix} 0.0200 \\ 0.0050 \\ 0.0000 \\ 0.0050 \\ 0.0200 \end{pmatrix}. \tag{34}$$

The vector of $\tilde{\mathbf{\Lambda}}$'s is obtained using the following identity:

$$\tilde{\mathbf{\Lambda}} = \mathbf{K}_{\text{tr}}^{-1} \left(\mathbf{y} - \epsilon \mathbf{1} + \pi \mathbf{1} + \mathbf{\Gamma} \right) = \begin{pmatrix} 5.0953 \\ 2.0405 \\ -14.2701 \\ 2.0405 \\ 5.0953 \end{pmatrix} \tag{35}$$

and the predicted training outputs $\mathbf{y}_{\text{tr,pr}}$ can be computed from

$$\mathbf{y}_{\text{tr,pr}} = \mathbf{K}_{\text{tr}} \tilde{\mathbf{\Lambda}} + b\mathbf{1} = \begin{pmatrix} 3.9967 \\ 1.0117 \\ 0.0167 \\ 1.0117 \\ 3.9967 \end{pmatrix}, \tag{36}$$

where $b = 0.0167$ was derived using the complementary slackness conditions in (29) and (30). In order to identify the predicted test output introduce the matrix $\mathbf{K}_{\rm ts} = [\mathbf{K}_{\rm tr}, \mathbf{K}_{{\rm tr},t}]^T$ with

$$\mathbf{K}_{\rm ts} = \left[\begin{array}{c} \mathbf{K}_{\rm tr} \\ \mathbf{K}_{{\rm tr},t} \end{array} \right] = \left[\begin{array}{ccccc} 0.3773 & 0.1110 & 0.0222 & 0.1110 & 0.3773 \\ 0.1110 & 0.0444 & 0.0222 & 0.0444 & 0.1110 \\ 0.0222 & 0.0222 & 0.0222 & 0.0222 & 0.0222 \\ 0.1110 & 0.0444 & 0.0222 & 0.0444 & 0.1110 \\ 0.3773 & 0.1110 & 0.0222 & 0.1110 & 0.3773 \\ 0.2219 & 0.0721 & 0.0222 & 0.0721 & 0.2219 \end{array} \right], \tag{37}$$

where the last row corresponds to the matrix $\mathbf{K}_{{\rm tr},t}$ in equation (14). Indeed, the expression $\mathbf{K}_{\rm ts}\tilde{\mathbf{\Lambda}} + b$ becomes now

$$\mathbf{K}_{\rm ts}\tilde{\mathbf{\Lambda}} + b\mathbf{1} = \left[\begin{array}{c} \mathbf{y}_{\rm tr,pr} \\ \mathbf{y}_{\rm ts,pr} \end{array} \right] = \left(\begin{array}{c} 3.9967 \\ 1.0117 \\ 0.0167 \\ 1.0117 \\ 3.9967 \\ 2.2554 \end{array} \right), \tag{38}$$

where the last value is the predicted value for $\mathbf{y}_{\rm ts}^T = (2.25)$. Equation (38) can also be interpreted as the closed form version of equations (3) and (4).

Next, let us discuss an approximation of the function $f(x) = \exp(x)$. We computed $f(x)$ for values from -5 to $+5$ at increments of 1.0. We chose $\epsilon = 0.5$ and $c = 1$. The semidefinite programming approach identified a feasible optimal solution with an objective function value of $t = 76100$. For the Lagrangian multiplier we found $\pi = -5.5912$. For the bias we calculated $b = 5.5903$, while the parameters for the basis matrices attained the values:

$$\mu = \left(\begin{array}{c} \mu_1 \\ \mu_2 \\ \mu_3 \end{array} \right) = \left(\begin{array}{c} 0.0001 \\ 0.0558 \\ 0.0016 \end{array} \right). \tag{39}$$

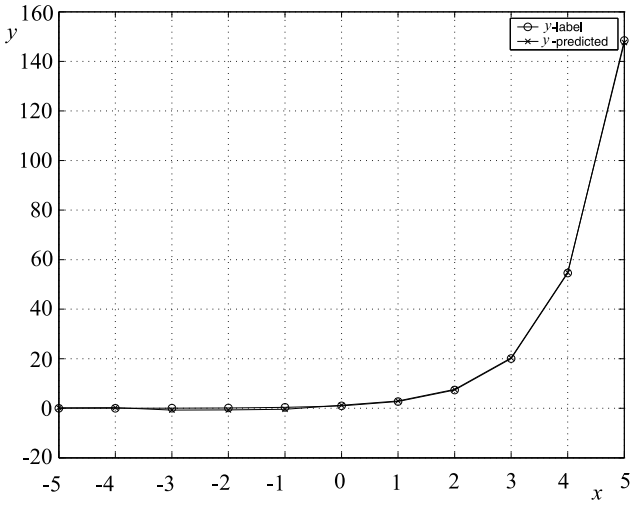

Fig. 1. Comparison $f(x) = \exp(x)$.

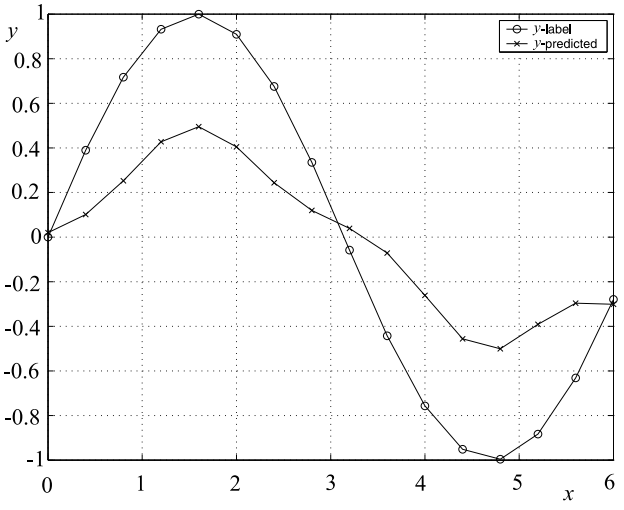

Fig. 2. Comparison $f(x) = \sin(x)$.

Figure 1 shows the graphs for the labels $\mathbf{y}_{\text{tr}}$ and the predicted values $\mathbf{y}_{\text{pr}}$ computed by SeDuMi.

Note that in this experiment the predicted training labels are almost identical to the true training labels, as the two curves are very close to each other.

For the third experiment the function $f(x) = \sin(x)$ was examined. Here, we computed labels from 0 to 6.28 at increments of 0.4. Once more, we chose $\epsilon = 0.5$ and $c = 1$. The solution of the semidefinite programming problem yielded an objective function value of $t = 3.5102$ and a Lagrangian multiplier of $\pi = -0.0132$. For the sinusoidal function the bias assumes a significantly smaller value of $b = 0.0086$ than was the case for the exponential function. For the parameters for the basis matrices we retrieve the values:

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{pmatrix} = \begin{pmatrix} 0.0002 \\ 0.0634 \\ -0.0041 \end{pmatrix}. \tag{40}$$

Figure 2 shows the two graphs for labels $\mathbf{y}_{\text{tr}}$ and the predicted values $\mathbf{y}_{\text{pr}}$.

Notice that for both the exponential and the sinusoidal function most of the weight of the μ_i is placed on the radial-basis function kernel, while for the quadratic function in the first experiment the polynomial kernel is disproportionately favored.

3.2. Benchmark tests

Hereafter, generalization performance for the semidefinite programming approach was computed using problems (25) and (26). Subsets with 100 samples of the publicly available datasets `abalone`[1], `add10`[2], and `boston`[3] were selected as benchmark datasets. All datasets were randomly split into a training and a test set with a training to test set ratio of 80 % : 20 %. For each dataset 30 different scenarios were created and results are presented as an average over these 30 scenarios. As a reference the best radial-basis function kernel support vector machine is also displayed. The kernel parameter was tuned using cross-validation over 30 training set scenarios. The `abalone` dataset was first preprocessed converting nonnumeric information and subsequently normalizing inputs and target. The goal is to predict the age of abalone from physical measurements. The value ϵ was set to $\epsilon = 0.05$. Performance was also evaluated on the first of the three synthetic Friedman-functions (`add10`), which are all popular benchmark datasets in regression analysis [20]. The first Friedman model has 10 attributes, however, the output value is computed by using only the first five inputs and by including normally distributed noise ξ. More specifically, the function $y = 10 \sin(\pi x_1 x_2) + 20 (x_3 - 0.5)^2 + 10x_4 + 5x_5 + \xi$ is to be examined. The 10 input variables are uniformly distributed in $[0, 1]$. A regression tube of $\epsilon = 5$ was selected. The dataset `boston` with 13 attributes describing various input characteristics (e. g. crime rate, pupil-teacher ratio, highway accessibility) predicts as output the value of a home. The samples are normalized and $\epsilon = 0.1$ was selected.

Table displays the mean square generalization error and the standard deviation for the semidefinite programming formulations (25) and (26). The table also lists the mean square error for the best radial-basis function support vector regression tuned using cross-validation. The second column shows the ϵ-values for which the experiments were conducted. Also, the corresponding values for the μ_i are displayed. Performance of the two SDP formulations is

[1] ftp://ftp.ics.uci.edu/pub/machine-learning-databases/abalone
[2] http://www.cs.toronto.edu/ delve/data/add10/desc.html
[3] http://lib.stat.cmu.edu/datasets/boston

Mean Square Error on Standard Benchmark Data

Dataset	ϵ	MSE(SDP)	MSE(SDP $\mu_i \geq 0$)	MSE(RBF-SVM)
		μ_1 / μ_2 / μ_3	μ_1 / μ_2 / μ_3	
		-0.0014 / 0.0111 / 0.0043	0 / 0.01 / 0	
`abalone`	0.05	$(10.400 \pm 1.923) \cdot 10^{-4}$	$(10.387 \pm 1.921) \cdot 10^{-4}$	$(10.456 \pm 1.923) \cdot 10^{-4}$
		μ_1 / μ_2 / μ_3	μ_1 / μ_2 / μ_3	
		0.0009 / 0.0057 / -0.0040	0.0003 / 0.0046 / 0	
`add10`	5	15.111 ± 3.594	15.453 ± 4.906	11.170 ± 2.314
		μ_1 / μ_2 / μ_3	μ_1 / μ_2 / μ_3	
		0.0041 / 0.0121 / -0.0186	0 / 0.01 / 0	
`boston`	0.1	$(44.013 \pm 6.483) \cdot 10^{-4}$	$(44.061 \pm 6.441) \cdot 10^{-4}$	$(40.897 \pm 9.601) \cdot 10^{-4}$

comparable, the difference between MSE(SDP) and MSE(SDP $\mu_i \geq 0$) is rather marginal for all three datasets. For the dataset `abalone` nonnegative μ_i's lead to a slight improvement of the mean square error. Furthermore, for the datasets `abalone` and `boston` nonnegative μ_i's result in a pure radial-basis function solution with $\mu_1 = \mu_3 = 0$ for both datasets. For unrestricted multipliers the dataset `abalone` shows a negative coefficient for the polynomial kernel, while the datasets `add10` and `boston` display a negative coefficient for the linear kernel. Compared to the best support vector machine the SDP approach is competitive for the datasets `abalone` and `boston`. Nonetheless one needs to keep in mind that the computational effort for evaluating problems (25) and (26) is significantly smaller when compared to support vector machine learning tuned using cross-validation. The solution for the dataset `abalone` for SDP ($\mu_i \geq 0$) is governed by a pure radial-basis function solution (μ_1 and μ_3 are zero). The mean square error for the corresponding support vector machine solution is very similar ($10.387 \cdot 10^{-4}$ versus $10.456 \cdot 10^{-4}$), since tuning for the dataset `abalone` resulted in an optimal radial-basis function parameter of $\sigma = 0.5$, which is identical to the σ used for $\mathbf{K}_2$.

Conclusion and Outlook

In this paper we have presented a new method for calculating the regression kernel matrix as a linear combination of three basis kernel matrices using semidefinite programming techniques. The coefficients for the three matrices were first left unconstrained and subsequently constrained to be nonnegative. The parameter selection problem is equivalent to a convex optimization problem guaranteeing that this algorithm identifies the global optimum. Preliminary experimentation on standard benchmark data show promising results for these techniques when compared to the best radial-basis function support vector machine.

References

[1] Bach F.R., Lanckriet G.R.G., Jordan M.I. Multiple Kernel Learning, Conic Duality, and the SMO Algorithm // Proc. of the 21st Intern. Conf. on Machine Learning, Banff, Canada, 2004.

[2] Bazaraa M.Z., Sherali H.D., Shetty C.M. Nonlinear Programming: Theory and Algorithms. N.Y.: John Wiley & Sons, 1993.

[3] Bi J., Bennett K.P. A geometric approach to support vector regression // Neurocomputing. 2003. Vol. 55. P. 79–108.

[4] Bertsekas D.P. Nonlinear Programming. Athena Scientific, Belmont, Massachusetts, 1999.

[5] Burges C.J.C. A tutorial on support vector machines for pattern classification // Data Mining and Knowledge Discovery. 1998. Vol. 2(2). P. 121–167.

[6] Chapelle O., Vapnik V., Bousquet O., Mukherjee S. Choosing multiple parameters for support vector machines // Machine Learning. 2002. Vol. 46(1/3). P. 131–159.

[7] Cristianini N., Shawe-Taylor J. An Introduction to Support Vector Machines. Cambridge: Cambridge Univ. Press, 2000.

[8] Cristianini N., Shawe-Taylor J., Kandola J., Elisseef A. On kernel target alignment // Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 2001.

[9] Lanckriet G., Cristianini N., El-Ghaoui L. et al. Learning the kernel matrix with semidefinite programming // J. of Machine Learning Research. 2004. Vol. 5. P. 27–72.

[10] Pardalos P.M., Wolkowicz H. (Eds) Topics in semidefinite and interior-point methods // Fields Institute Communications Series. 1998. Vol. 18. Amer. Math. Society.

[11] Ramana M., Pardalos P.M. Semidefinite programming // Interior point methods of mathematical programming / T. Terlaky (Ed.). N.Y.: Kluwer Acad. Publ., 1996. P. 369–398.

[12] Schölkopf B., Smola A.J. Learning with Kernels. Cambridge, Massachusetts: MIT Press, 2002.

[13] Smola A., Schölkopf B. A tutorial on support vector regression // Statistics and Computing. 2004. Vol. 14(3). P. 199–222.

[14] Sturm J.F. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones // Optimization Methods and Software. 1999. Vol. 11–12. P. 625–653.

[15] Löfberg J. YALMIP: a Toolbox for Modeling and Optimization in MATLAB // Proc. of the CACSD Conf., 2004. Taipei, Taiwan. `http://control.ee.ethz.ch/~joloef/yalmip.php`.

[16] Trafalis T.B., Malyscheff A.M. Optimal selection of the regression kernel matrix with semidefinite programming // Frontiers In Global Optimization. Ser. on Nonconvex Optimization and Its Applications / C.A. Floudas, P.M. Pardalos (eds). N.Y.: Kluwer Acad. Publ., 2004. P. 575–584.

[17] Vandenberghe L., Boyd S. Semidefinite programming // SIAM Review. 1996. Vol. 38(1).

[18] Vapnik V. Estimation of Dependencies Based on Empirical Data. Berlin: Springer Verlag, 1982.

[19] Vapnik V. The Nature of Statistical Learning Theory. Berlin: Springer Verlag, 1995.

[20] Vapnik V. Statistical Learning Theory. N.Y.: John Wiley & Sons, 1998.

Received for publication September 18, 2005