A Note on the Decomposition Methods for Support Vector Regression

Shuo-Peng Liao, Hsuan-Tien Lin, and Chih-Jen Lin

Department of Computer Science and
Information Engineering
National Taiwan University
Taipei 106, Taiwan.
E-mail: cjlin@csie.ntu.edu.tw

Abstract

The dual formulation of support vector regression involves with two closely related sets of variables. When the decomposition method is used, many existing approaches use pairs of indices from these two sets as the working set. Basically they select a base set first and then expand it so that all indices are pairs. This makes the implementation different from that for support vector classification. In addition, a larger optimization sub-problem has to be solved in each iteration. In this paper from different aspects we demonstrate that there are no needs to do so. In particular we show that directly using this base set as the working set leads to similar convergence (number of iterations). Therefore, not only the program can be simpler, with a smaller working set and similar number of iterations, it can also be more efficient.

1 Introduction

Given a set of data points, $\{(x_1, z_1), \ldots, (x_l, z_l)\}$, such that $x_i \in R^n$ is an input and $z_i \in R^1$ is a target output. A major form for solving support vector regression (SVR) is the following optimization problem [17]:

$$\begin{aligned} \min \quad & \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + \epsilon \sum_{i=1}^{l}(\alpha_i + \alpha_i^*) \\ & + \sum_{i=1}^{l} z_i(\alpha_i - \alpha_i^*) \\ & \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, \\ & 0 \le \alpha_i, \alpha_i^* \le C, i = 1, \ldots, l, \end{aligned} \tag{1.1}$$

where C is the upper bound, $Q_{ij} \equiv \phi(x_i)^T \phi(x_j)$, α_i and α_i^* are Lagrange multipliers associated to ith data x_i, and ϵ is the parameter of the loss function. Note that training vectors x_i are mapped into a higher dimensional space by the function ϕ. An important property is that at the optimal solution, $\alpha_i \alpha_i^* = 0, i = 1, \ldots, l$.

Due to the density of Q, currently the decomposition method is the major method to solve (1.1) [16, 7, 9]. It is an iterative process where in each iteration the index set of variables are separated to two sets B and N, where B is the working set. Then in that iteration variables corresponding to N are fixed while a sub-problem on variables corresponding to B is minimized.

Following approaches for support vector classification, there are methods for selecting the working set. For many existing approaches for regression, after these methods are applied to find a base set, they expand it so that all elements are pairs. For example, if $\{\alpha_i, \alpha_j^*\}$ are chosen first, they include $\{\alpha_i^*, \alpha_j\}$ into the working set. Then the following sub-problem of four variables $(\alpha_i, \alpha_i^*, \alpha_j, \alpha_j^*)$ is solved:

$$\begin{aligned} \min \quad & \frac{1}{2}\begin{bmatrix}\alpha_i - \alpha_i^* \\ \alpha_j - \alpha_j^*\end{bmatrix}^T \begin{bmatrix}Q_{ii} & Q_{ij} \\ Q_{ji} & Q_{jj}\end{bmatrix} \begin{bmatrix}\alpha_i - \alpha_i^* \\ \alpha_j - \alpha_j^*\end{bmatrix} \\ & +(Q_{i,N}(\alpha_N - \alpha_N^*) + z_i)(\alpha_i - \alpha_i^*) \\ & +(Q_{j,N}(\alpha_N - \alpha_N^*) + z_j)(\alpha_j - \alpha_j^*) \\ & +\epsilon(\alpha_i + \alpha_i^* + \alpha_j + \alpha_j^*) \\ & (\alpha_i - \alpha_i^*) + (\alpha_j - \alpha_j^*) = -\sum_{s \ne i,j}(\alpha_s - \alpha_s^*) \\ & 0 \le \alpha_i, \alpha_j, \alpha_i^*, \alpha_j^* \le C. \end{aligned} \tag{1.2}$$

Note that α_N and α_N^* are variables corresponding to N. They are fixed here.

A reason of doing so is to maintain $\alpha_i \alpha_i^* = 0, i =$

$1,\ldots,l$ throughout all iterations. Hence the number of nonzero variables in the iterative process can be kept small.

However, it has been shown in [12, Theorem 4.1] that for some existing work (e.g. [7, 9]), if they do not expand the base set to pairs, the property $\alpha_i \alpha_i^* = 0, i = 1,\ldots,l$ still holds. In Section 2 we will elaborate on this in more detail.

Recently there have been implementation without using pairs of indices. For example, LIBSVM [2], SVMTorch [3], and mySVM [15]. A question immediately raised is on the performance of these two approaches. From one hand, we can think that an expanded working set leads to a larger sub-problem so the total number of iterations may be less. We also note that additional elements of those pairs are obtained for free. On the other hand, a larger sub-problem takes more time so the cost of each iteration is higher.

We discuss this issue in Section 3. First we consider approaches with the smallest working set size (i.e. two and four for both approaches) where the analytic solution of the sub-problem is handily available. This is from the Sequential Minimal Optimization (SMO) [14]. From mathematical explanation we show that while solving the four-variable sub-problems, in most cases, only those two variables obtained in the first stage of the working set selection are updated. Therefore, the number of iterations of both two-variable and four-variable approaches are nearly the same. Details of the proof are in an earlier technical report [11]. Hence there is no need to expand the working set using pairs of indices. About larger working sets, we also discussed that after some finite number of iterations, the sub-problem using only the base set are already optimal for the subproblem using pairs of variables. This gives us a theoretical justification that it is not necessary to use pairs of variables.

It is important to clarify the above properties. With them, the implementation can be simpler and easier.

In Section 4 we conduct experiments to demonstrate the validity of our analysis. We also discuss that without expanding the working set, the implementation of regression code can be nearly the same as that for classification.

There are other decomposition approaches for support vector regression (for example, [4]). They dealt with different situations which will not be discussed here.

2 Working Set Selection

Here we consider the working set selection from [6, 8] which were originally designed for classification cases. Remember that the dual formulations of support vector classification is:

$$\begin{aligned} \min \quad & \frac{1}{2}\alpha^T Q \alpha - e^T \alpha \\ & 0 \le \alpha_i \le C, \quad i = 1,\ldots,l, \\ & y^T \alpha = 0, \end{aligned} \tag{2.1}$$

where $y \in R^l$ with $y_i \in \{1, -1\}$ and e is the vector of all ones,

To make SVR similar to the classification formula, we define the following $2l$ by 1 vectors:

$$\alpha^{(*)} \equiv \begin{pmatrix} \alpha \\ \alpha^* \end{pmatrix}, \text{ and}$$

$$y_i = \begin{cases} +1 & i = 1,\ldots,l, \\ -1, & i = l+1,\ldots,2l. \end{cases} \tag{2.2}$$

Then the regression problem (1.1) can be reformulated as

$$\begin{aligned} \min \quad f(\alpha^{(*)}) = \; & \frac{1}{2}(\alpha^{(*)})^T \begin{bmatrix} Q & -Q \\ -Q & Q \end{bmatrix} \alpha^{(*)} \\ & + \left[\epsilon e^T + z^T, \epsilon e^T - z^T\right] \alpha^{(*)} \\ & 0 \le \alpha_i^{(*)} \le C, \quad i = 1,\ldots,2l, \\ & y^T \alpha^{(*)} = 0. \end{aligned} \tag{2.3}$$

Now f is the objective function of (2.3). We define

$$\begin{aligned} m(\alpha^{(*)}) \equiv \max(& \max_{\alpha_t^{(*)} < C, y_t = 1} -y_t \nabla f(\alpha^{(*)})_t, \\ & \max_{\alpha_t^{(*)} > 0, y_t = -1} -y_t \nabla f(\alpha^{(*)})_t), \end{aligned} \tag{2.4}$$

$$\begin{aligned} M(\alpha^{(*)}) \equiv \min(& \min_{\alpha_t^{(*)} > 0, y_t = 1} -y_t \nabla f(\alpha^{(*)})_t, \\ & \min_{\alpha_t^{(*)} < C, y_t = -1} -y_t \nabla f(\alpha^{(*)})_t). \end{aligned} \tag{2.5}$$

For the convenience, we define the candidate of $m(\alpha^{(*)})$ as the set of all indices t which satisfy $\alpha_t^{(*),k} < C, y_t = 1$ or $\alpha_t^{(*),k} > 0, y_t = -1$ where $1 \le t \le 2l$. It is similar to candidate of $M(\alpha^{(*)})$.

The KKT condition states that a feasible $\alpha^{(*)}$ is an optimal solution if and only if

$$M(\alpha^{(*)}) - m(\alpha^{(*)}) \ge 0. \tag{2.6}$$

In each iteration, if $\alpha^{(*),k}$ stands for the current iteration, we define $m_k \equiv m(\alpha^{(*),k})$ and $M_k \equiv$

$M(\alpha^{(*),k})$. Also, let argm_k be the set of indices whose $-y_i\nabla f(\alpha^{(*),k})_i$ are the same as $m(\alpha^{(*),k})$. Similarly we define argM_k.

Thus during iterations of the decomposition method, $\alpha^{(*),k}$ is not an optimal solution yet so

$$m_k > M_k, \text{ for all } k.$$

If two elements are selected as the working set, intuitively we tend to choose indices i and j which satisfy

$$i \in \arg m_k \text{ and } j \in \arg M_k, \tag{2.7}$$

as they cause the maximal violation of the KKT condition.

A systematic way to select a larger working set in each iteration can be as follows. If q, an even number, is the size of the working set, $q/2$ indices are sequentially selected from the largest $-y_i\nabla f(\alpha^{(*)})_i$ values to smaller in candidates of m_k. That is,

$$-y_{i_1}\nabla f(\alpha^{(*),k})_{i_1} \geq \cdots \geq -y_{i_{\frac{q}{2}}}\nabla f(\alpha^{(*),k})_{i_{\frac{q}{2}}},$$

where $i_1 \in \arg m_k$. The other $q/2$ indices are sequentially selected from the smallest $-y_i\nabla f(\alpha^{(*)})_i$ values to larger in candidates of M_k. That is,

$$-y_{j_{\frac{q}{2}}}\nabla f(\alpha^{(*),k})_{j_{\frac{q}{2}}} \geq \cdots \geq -y_{j_1}\nabla f(\alpha^{(*),k})_{j_1},$$

where $j_1 \in \arg M_k$. Also, we have

$$-y_{j_{\frac{q}{2}}}\nabla f(\alpha^{(*),k})_{j_{\frac{q}{2}}} < -y_{i_{\frac{q}{2}}}\nabla f(\alpha^{(*),k})_{i_{\frac{q}{2}}}. \tag{2.8}$$

to ensure that the intersection of both selection groups is empty. Thus if q is not small, sometimes the actual number of selected indices may be less than q.

Note that this is the same as the working set selection in [14]. However, the original derivation in [14] was from the concept of feasible directions in constrained optimization but not from the violation of the KKT condition.

After the base set of q indices is selected, earlier approaches [7, 10] expand the set so that all elements in it are pairs.

However, if directly using elements in the base set, the following theorem has been proved in [12, Theorem 4.1]:

Theorem 2.1 *If the initial solution is zero, then* $\alpha_i^k(\alpha^*)_i^k = 0, i = 1, \ldots, l$ *for all* k.

Another important issue for the decomposition method is the stopping criteria. From (2.6), a natural choice of the stopping criteria is

$$M_k - m_k \geq -\delta, \tag{2.9}$$

where δ, the stopping tolerance, is a small positive number. The stopping criteria (2.9) for the $q = 2$ case using indices i and j selected from (2.7) is

$$-y_j\nabla f(\alpha^{(*),k})_j - (-y_i\nabla f(\alpha^{(*),k})_i) \geq -\delta. \tag{2.10}$$

The convergence of the decomposition method under some conditions of the kernel Q is shown in [12] for the base method. Some theoretical justification on the use the stopping criteria (2.9) for the decomposition method is in [13]. For the method of using pairs, however, no particular convergence proof has been made, but we will assume it for our analyses.

3 Number of Iterations

In this section we will show that in final iterations using only the base set is the same as using pairs of indices as the working set.

First we state an important property on the difference between the ith and $(i+l)$th gradient elements. Consider α_i and α_i^*. We have

$$\begin{aligned}\nabla f(\alpha^{(*)})_{i+l} &= -(Q(\alpha - \alpha^*))_i + \epsilon - z_i \\ &= -\nabla f(\alpha^{(*)})_i + 2\epsilon. \end{aligned} \tag{3.1}$$

We will use this frequently in later analyses.

In [11] we discussed the case of $q = 2$. Using (3.1) [11] shows that i and i^* are never selected at the same iteration so the approach using pairs always has four elements in the working set. It then proves the following result:

Theorem 3.2 *For all iterations with the violation on the stopping criterion (2.10) no more than* 2ϵ*, an optimal solution of the two-variable sub-problem is already an optimal solution of the corresponding four-variable sub-problem.*

Therefore, for the four-variable sub-problem, in final iterations only two indices from the base set are still modified. If ϵ is not small, in most iterations the stopping tolerance is smaller than 2ϵ. In addition, as most decomposition iterations are spent in the final stage (due to slow convergence), this theorem has shown a conclusive result that no matter using two-variable or four-variable approaches, the difference on the number of iterations should not be much.

For general cases ($q > 2$), we may not be able to get results as elegant as Theorem 3.2. When $q = 2$, we exactly know the relation on the changes of $\alpha_i^{(*)}$ and $\alpha_j^{(*)}$ as $y_i(\alpha_i^{(*),k} - \alpha_i^{(*),k}) = -y_j(\alpha_j^{(*),k} - \alpha_j^{(*),k})$. However, when $q > 2$, the change on each variable can be different. Anyway in the following we will show a similar but weaker result.

If not stopped in finite iterations, the decomposition method generates an infinite sequence which converges to an optimal solution. In the following we will show that in final iterations, i.e. after k is large enough, solving the sub-problem with q variables is the same as solving the larger sub-problem which contains pairs of variables. Next we describe some properties which will be used for the proof.

Assume that the sequence $\{\alpha^{(*),k}\}$ of the method using only q elements from the base set converges to an optimal solution $\bar{\alpha}^{(*)}$. Then we can define

$$\bar{M} \equiv M(\bar{\alpha}^{(*)}) \text{ and } \bar{m} \equiv m(\bar{\alpha}^{(*)}). \tag{3.2}$$

We also note that (2.8) implies that for any index i in the working set of the kth iteration,

$$M_k \leq -y_i \nabla f(\alpha^{(*),k})_i \leq m_k. \tag{3.3}$$

We then describe two theorems from [13] which are needed for the main proof. [13] deals with a general framework of decomposition methods for different SVM formulations. We can easily check that the current working selection satisfies required conditions in [13] so these two theorems can be applied:

Theorem 3.3

$$\lim_{k \to \infty} m_k - M_k = 0. \tag{3.4}$$

Theorem 3.4 *For any $\bar{\alpha}_i^{(*)}$ whose corresponding $-y_i \nabla f(\bar{\alpha}^{(*)})_i$ is neither $\bar{m}$ nor $\bar{M}$, after k is large enough, $\alpha_i^{(*),k}$ is at a bound and is equal to $\bar{\alpha}_i^{(*)}$.*

Immediately we have a corollary of Theorem 3.3 which is specific to the regression problems:

Corollary 3.5 *After k is large enough, α_i^k and $(\alpha^*)_i^k$ would not be both selected in the working set.*

Proof: By the convergence of $m_k - M_k$ to 0, after k is large enough, $m_k - M_k < \epsilon$. If α_i^k and $(\alpha^*)_i^k$ are both selected in the working set, from (3.3), $M_k \leq -\nabla f(\alpha^{(*),k})_i \leq m_k$ and $M_k \leq \nabla f(\alpha^{(*),k})_{i+l} \leq m_k$. However, (3.1) shows $\nabla f(\alpha^{(*),k})_{i+l} = -\nabla f(\alpha^{(*),k})_i + 2\epsilon$ so $m_k - M_k \geq 2\epsilon$ and there is a contradiction. ■

The main result of this section is:

Theorem 3.6 *We assume that $\bar{M} \neq \bar{m} + 2\epsilon$. After k is large enough, any optimization sub-problem by using only q elements is already optimal for the larger sub-problem which contains pairs of variables.*

Proof: If the result is wrong, there is an index i and an infinite set $\mathcal{K}$ such that for all $k \in \mathcal{K}$, α_i^k (or $(\alpha^*)_i^k$) is selected in the working set but then $(\alpha^*)_i^k$ (or α_i^k) is also modified. Without loss of generality, we assume that α_i^k is selected in the working set but $(\alpha^*)_i^k$ is modified infinite times.

By Theorem 3.4, since $(\alpha^*)_i^k$ is modified infinite times, $\nabla f(\bar{\alpha}^{(*)})_{i+l} = \bar{m}$ or $\nabla f(\bar{\alpha}^{(*)})_{i+l} = \bar{M}$. For the first case, (3.1) implies that $-\nabla f(\bar{\alpha}^{(*)})_i < \bar{m}$ while the second case implies that $-\nabla f(\bar{\alpha}^{(*)})_i = \bar{M} - 2\epsilon < \bar{M}$. For the second case, by the assumption that $\bar{m} \neq \bar{M} - 2\epsilon$, $\bar{m} < -\nabla f(\bar{\alpha}^{(*)})_i$ or $\bar{m} > -\nabla f(\bar{\alpha}^{(*)})_i$. If $\bar{m} < -\nabla f(\bar{\alpha}^{(*)})_i$, we have $\bar{m} < -\nabla f(\bar{\alpha}^{(*)})_i < \bar{M}$ which is impossible for an optimal solution. Hence

$$-y_i \nabla f(\bar{\alpha}^{(*)})_i < \bar{m} \tag{3.5}$$

holds for both cases.

Therefore, we can define

$$\Delta \equiv \min(\epsilon/2, (\bar{m} - (-y_i \nabla f(\bar{\alpha}^{(*)})_i))/3) > 0.$$

By the convergence of the sequence $\{-y_j \nabla f(\alpha^{(*),k})_j\}$ to $-y_j \nabla f(\bar{\alpha}^{(*)})_j$, for all $j = 1, \ldots, 2l$, after k is large enough,

$$|y_j \nabla f(\alpha^{(*),k})_j - y_j \nabla f(\alpha^{(*),k+1})_j| \leq \Delta \text{ and} \tag{3.6}$$

$$|y_j \nabla f(\alpha^{(*),k})_j - y_j \nabla f(\bar{\alpha}^{(*)})_j| \leq \Delta. \tag{3.7}$$

Suppose that at the kth iteration $j \in \arg M_k$ is selected in the working set and

$$-y_j \nabla f(\alpha^{(*),k})_j = M_k.$$

By (3.3), (3.6), (3.7), and (3.5),

$$\begin{aligned}
&-y_j\nabla f(\bar{\alpha}^{(*)})_j\\
&\le -y_j\nabla f(\alpha^{(*),k})_j+\Delta\\
&=M_k+\Delta\\
&\le -y_i\nabla f(\alpha^{(*),k})_i+\Delta\\
&\le -y_i\nabla f(\bar{\alpha}^{(*)})_i+2\Delta\\
&\le -y_i\nabla f(\bar{\alpha}^{(*)})_i+2(\bar{m}-(-y_i\nabla f(\bar{\alpha}^{(*)})_i))/3\\
&<\bar{m}\le\bar{M}. \qquad (3.8)
\end{aligned}$$

From Theorem 3.4 and (3.8), after k is large enough, $\alpha_j^{(*),k}$ is bounded and is equal to $\bar{\alpha}_j^{(*)}$. That is, $\alpha_j^{(*),k} = \alpha_j^{(*),k+1} = \bar{\alpha}_j^{(*)}$. Since $\alpha_j^{(*),k+1} = \alpha_j^{(*),k}$ and $\alpha_j^{(*),k}$ is a candidate of M_k, by (3.3), (3.6), and (3.7)

$$\begin{aligned}
M_{k+1} &\le -y_j\nabla f(\alpha^{(*),k+1})_j\\
&\le -y_j\nabla f(\alpha^{(*),k})_j+\Delta\\
&= M_k+\Delta\\
&\le -y_i\nabla f(\alpha^{(*),k})_i+\Delta\\
&\le -y_i\nabla f(\alpha^{(*),k+1})_i+2\Delta.
\end{aligned}$$

Hence we get

$$M_{k+1}\le -y_i\nabla f(\alpha^{(*),k+1})_i+\epsilon. \qquad (3.9)$$

On the other hand, $(\alpha^*)_i^k$ is modified to $(\alpha^*)_i^{k+1}$. At least one of them is strictly positive. By the definition of m_k,

$$\begin{aligned}
&\nabla f(\alpha^{(*),k})_{i+l}\le m_k, \text{ or}\\
&\nabla f(\alpha^{(*),k+1})_{i+l}\le m_{k+1}.
\end{aligned} \qquad (3.10)$$

From (3.1), (3.3), (3.9), and (3.10), for all large enough $k\in\mathcal{K}$,

$$\begin{aligned}
&M_k\le m_k-2\epsilon, \text{ or}\\
&M_{k+1}\le m_{k+1}-\epsilon.
\end{aligned}$$

Therefore,

$$\lim_{k\to\infty} m_k - M_k \neq 0$$

which contradicts Theorem 3.3. ■

4 Experiments

In this section we experiment with some practical problems to confirm our analysis.

Table 4.1: Problem abalone (first 200 data, $q = 10$)

Parameters	Iter.(base)	Iter.(pairs)	Jumps
$C = 10, \epsilon = 0.1$	132	173	77
$C = 10, \epsilon = 1$	49	46	3
$C = 10, \epsilon = 10$	1	1	0
$C = 100, \epsilon = 0.1$	1641	1983	184
$C = 100, \epsilon = 1$	807	552	29
$C = 100, \epsilon = 10$	1	1	0

Note: "Jumps" means the number of changes in totally about Iter.$\times q$ components.

Table 4.2: Problem abalone (first 200 data, $q = 20$)

Parameters	Iter.(base)	Iter.(pairs)	Jumps
$C = 10, \epsilon = 0.1$	100	114	111
$C = 10, \epsilon = 1$	42	40	10
$C = 10, \epsilon = 10$	1	1	0
$C = 100, \epsilon = 0.1$	1168	1086	250
$C = 100, \epsilon = 1$	258	310	46
$C = 100, \epsilon = 10$	1	1	0

We consider two regression problems abalone (4177 data) and add10 (9792 data) from [1] and [5], respectively. The RBF kernel is used:

$$Q_{ij}\equiv e^{-\|x_i-x_j\|^2/n},$$

where n is the number of attributes in each data. For these two problems, n is eight and ten, respectively.

We use a simple implementation written in MATLAB so only small problems are tested. We consider the first 200 data points of abalone and add10. Results are in Tables 4.1 to 4.4. For each problem we test two different sizes of the working set: $q = 10$ and $q = 20$. Then in each table we try different ϵ and C. We did not do any model selections as our purpose is not on the solution quality. We consider ϵ below to 0.1 because the number of support vectors has approached the number of training data. On the other hand, the largest ϵ used is 10 as the number of support vectors has been close to zero. For each parameter set, we present the number of iterations by both approaches using base set and pairs of variables, number of support vectors (bounded support vectors), and the number of jumps. If pairs of indices are used, the working set is the union

Table 4.3: Problem add10 (first 200 data, $q = 10$)

Parameters	Iter.(base)	Iter.(pairs)	Jumps
$C = 10, \epsilon = 0.1$	59	50	13
$C = 10, \epsilon = 1$	55	60	0
$C = 10, \epsilon = 10$	2	2	0
$C = 100, \epsilon = 0.1$	2519	2094	112
$C = 100, \epsilon = 1$	944	956	12
$C = 100, \epsilon = 10$	2	2	0

Table 4.4: Problem add10 (first 200 data, $q = 20$)

Parameters	Iter.(base)	Iter.(pairs)	Jumps
$C = 10, \epsilon = 0.1$	27	26	33
$C = 10, \epsilon = 1$	18	18	0
$C = 10, \epsilon = 10$	2	2	0
$C = 100, \epsilon = 0.1$	1317	1216	135
$C = 100, \epsilon = 1$	236	278	28
$C = 100, \epsilon = 10$	2	2	0

of two sets: B from (2.7) and its complement B^*. We check that before and after solving the sub-problem, how many components of α_{B^*} are changed. Then the total number of such changes are shown in the "Jumps" column.

It can be clearly seen that both approaches have similar number of iterations. In addition, the number of jumps is very small, especially when ϵ is larger. In addition, it can be clearly seen that comparing to the totally about Iter.$\times q$ components of B^* (the size of B^* in each iteration may be less than q if B contains pairs; this seldom happens, from Corollary 3.5), the number of components changed is very small.

Acknowledgments

This work was supported in part by the National Science Council of Taiwan via the grant NSC 89-2213-E-002-106.

References

[1] C. L. Blake and C. J. Merz. UCI repository of machine learning databases. Technical report, University of California, Department of Information and Computer Science, Irvine, CA, 1998. Available at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[2] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[3] R. Collobert and S. Bengio. SVMTorch: A support vector machine for large-scale regression and classification problems. *Journal of Machine Learning Research*, pages 143–160, 2001. Available at `http://www.idiap.ch/learning/SVMTorch.html`.

[4] G. W. Flake and S. Lawrence. Efficient SVM regression training with SMO. *Machine Learning*, 2001. To appear.

[5] J. Friedman. Multivariate adaptive regression splines. Technical Report No. 102, Laboratory for Computational Statistics, Department of Statistics, Stanford University, 1988.

[6] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA, 1998. MIT Press.

[7] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to SMO algorithm for SVM regression. Technical Report CD-99-16, Department of Mechanical and Production Engineering, National University of Singapore, 1999. To appear in IEEE Transactions on Neural Networks.

[8] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13:637–649, 2001.

[9] P. Laskov. An improved decomposition algorithm for regression support vector machines. In *Workshop on Support Vector Machines, NIPS99*, 1999.

[10] P. Laskov. An improved decomposition algorithm for regression support vector machines. *Machine Learning*, 2001. To appear.

[11] S.-P. Liao, H.-T. Lin, and C.-J. Lin. A note on the decomposition methods for support vector regression. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, 2000.

[12] C.-J. Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 2001. To appear.

[13] C.-J. Lin. Stopping criteria of decomposition methods for support vector machines: a theoretical justification. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001.

[14] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA, 1998. MIT Press.

[15] S. Rüping. mySVM - another one of those support vector machines, 2000. Software available at `http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/`.

[16] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. Neuro COLT Technical Report TR-1998-030, Royal Holloway College, 1998.

[17] V. Vapnik. *Statistical Learning Theory.* Wiley, New York, NY, 1998.