Journal of Machine Learning Research 1 (2006) Submitted 02/06; Revised 10/06; Published

General Polynomial Time Decomposition Algorithms

Nikolas List nlist@lmi.rub.de
Hans Ulrich Simon simon@lmi.rub.de
Fakultät für Mathematik,
Ruhr-Universität Bochum,
44780 Bochum,
Germany

Editor:

Abstract

We present a general decomposition algorithm that is uniformly applicable to every (suitably normalized) instance of Convex Quadratic Optimization and efficiently approaches an optimal solution. The number of iterations required to be within ε of optimality grows linearly with $1/\varepsilon$ and quadratically with the number m of variables. The working set selection can be performed in polynomial time. If we restrict our considerations to instances of Convex Quadratic Optimization with at most k_0 equality constraints for some fixed constant k_0 plus some so-called box-constraints (conditions that hold for most variants of SVM-optimization), the working set is found in linear time. Our analysis builds on a generalization of the concept of rate certifying pairs that was introduced by Hush and Scovel. In order to extend their results to arbitrary instances of Convex Quadratic Optimization, we introduce the general notion of a rate certifying q-set. We improve on the results by Hush and Scovel (2003) in several ways. First our result holds for Convex Quadratic Optimization whereas the results by Hush and Scovel are specialized to SVM-optimization. Second, we achieve a higher rate of convergence even for the special case of SVM-optimization (despite the generality of our approach). Third, our analysis is technically simpler.
We prove furthermore that the strategy for working set selection which is based on rate certifying sets coincides with a strategy which is based on a so-called "sparse witness of sub-optimality". Viewed from this perspective, our main result improves on convergence results by List and Simon (2004) and Simon (2004) by providing convergence rates (and by holding under more general conditions).

Keywords: Convex Quadratic Optimization, Decomposition Algorithms, Support Vector Machines

1. Introduction

Support vector machines (SVMs) introduced by Vapnik and co-workers (Boser et al., 1992; Vapnik, 1998) are a promising technique for classification, function approximation, and other key problems in statistical learning theory. In this paper, we consider the optimization problems that are induced by SVMs. These SVM-optimization problems (SVO) are special cases of Convex Quadratic Optimization (CQO).

The difficulty of solving problems of this kind is the density of the matrix that represents the "quadratic part" of the cost function. Thus, a prohibitive amount of memory is required to store the matrix and traditional optimization algorithms (such as Newton, for example) cannot be directly applied. Several authors have proposed (different variants of) a decomposition method to overcome this difficulty (Osuna et al., 1997; Joachims, 1998; Platt, 1998; Saunders et al., 1998; Mangasarian and Musicant, 1999, 2000, 2001; Chang et al., 2000; Keerthi et al., 2000, 2001; Keerthi and Gilbert, 2002; Lin, 2001b, 2002a,b; Laskov, 2002; Chang and Lin, 2001; Hsu and Lin, 2002; Liao et al., 2002; Hush and Scovel, 2003; List and Simon, 2004; List, 2004; Chen et al., 2005). Given an instance of CQO, this method keeps track of a current feasible solution which is iteratively improved. In each iteration the variable indices are split into a "working set" $I \subseteq \{1, \ldots, m\}$ and its complement $J = \{1, \ldots, m\} \setminus I$. Then, the simplified instance with the variables x_i, $i \in I$, is solved, thereby leaving the values for the remaining variables x_j, $j \in J$, unchanged. The success of the method depends in a quite sensitive manner on the policy for the selection of the working set I (whose size is typically much smaller than m). Ideally, the selection procedure should be computationally efficient and, at the same time, effective in the sense that the resulting feasible solutions become cost-optimal in the limit (with high speed of convergence).

Our results and their relation to previous work: Hush and Scovel (2003) were concerned with SVO. They introduced the notion of an "α-rate certifying pair" and showed that every decomposition algorithm for SVO that always inserts an α-rate certifying pair in its current working set comes within ε of optimality after $O(1/(\varepsilon\alpha^2))$ iterations. Building on a result by Chang et al. (2000), they presented furthermore an algorithm that constructs an $1/m^2$-rate certifying pair in $O(m \log m)$ steps.[1] Combining these results, we see that the decomposition algorithm of Hush and Scovel for problem SVO is within ε of optimality after $O(m^4/\varepsilon)$ iterations.

In this paper we present an extension of (and an improvement on) this result. We first define the general notion of an α-rate certifying q-set and show (with a simplified analysis) that it basically fits the same purpose for CQO as the α-rate certifying pair for SVO, where the number of iterations needed to be within ε of optimality is proportional to $q/(\varepsilon\alpha^2)$. We present a general decomposition algorithm that is uniformly applicable to every (suitably normalized) instance of CQO. Given an instance with k equality constraints and m variables, it finds an $1/m$-rate certifying $(k+1)$-set in polynomial time.[2] Combining these results, we are within ε of optimality after $O(km^2/\varepsilon)$ iterations of our decomposition algorithm. The problem SVO (considered by Hush and Scovel) has only one equality constraint. Plugging in $k = 1$ in our general result, we arrive at an upper bound on the number of iterations that improves on the bound obtained by Hush and Scovel by factor m^2. The analysis of Hush and Scovel (2003) builds on an earlier analysis of conditional gradient algorithms given by Dunn (1979). For this part of the analysis, we will present simpler arguments.

List and Simon (2004) suggested a strategy for working set selection that is based on a so-called "sparse witnesses of sub-optimality". They were able to prove that this strategy leads to feasible solutions of minimum cost in the limit. To this end, they had to impose a technical restriction on the cost function. Their result states the mere convergence without

1. Time bound $O(m \log m)$ can be improved to $O(m)$ by using the method of Simon (2004).
2. Moreover, the algorithm can be implemented such as to find this set even in linear time when we restrict its application to instances of CQO with at most k_0 equality constraints for some fixed constant k_0. If we restrict its application to SVO, we may use the highly efficient method of Simon (2004).

providing any convergence rates. In this paper, we prove (by means of Linear Programming duality) that the strategy for working set selection which is based on rate certifying sets coincides with the strategy of List and Simon (2004). Viewed from this perspective, our main result improves on the convergence result by List and Simon (2004) (and on a related result by Simon (2004)) by providing convergence rates (and by holding under more general conditions).

There are some alternatives to the approach we follow here. A paper by Lin (2001a) seems to imply the following quite strong result for SVO (although not stating it explicitly): decomposition algorithms following the approach of maximally KKT-violating pairs are within ε of optimality after only $O(\log 1/\varepsilon)$ iterations.[3] However, the analysis is specialized to SVO. Furthermore it has to assume strict convexity of the objective function and some non-degeneracy conditions. The convergence rate is only given in terms of ε whereas the dependence on problem parameters (like, for example, m) is not clarified.

Another algorithm (related to but different from decomposition algorithms) is SimpleSVM (Vishwanthan et al., 2003) which tries to iteratively include the support vectors in the working set. Assuming strict convexity of the objective function, Vishwanthan et al. (2003) claim a linear convergence of the method (but do neither give a complete proof nor exhibit the dependence on the various parameters explicitly). The main difference between SimpleSVM and decomposition algorithms is the size of the working set which can grow-up to the number of support vectors in the former case and is kept constant in the latter. Note that the number of support vectors is particularly large on noisy data.

2. Preliminaries

After fixing some notation (Section 2.1), we briefly recall the main result by Hush and Scovel (2003) about SVO and rate certifying pairs (Section 2.2) and the main result by List and Simon (2004) about CQO and sparse witnesses of sub-optimality (Section 2.3).

2.1 Notations and Basic Facts

We are mainly concerned with the problem "Convex Quadratic Optimization with box-constraints". It is denoted simply as CQO in this paper and is formally given as follows:

Definition 1 (CQO) *An instance $\mathcal{P}$ of CQO is given by*

$$\min_x f(x) = \frac{1}{2}x^\top Qx + w^\top x \text{ s.t. } Ax = b, l \le x \le r \ ,$$

where

- $Q \in \mathbb{R}^{m\times m}$ *is a symmetric positive semi-definite matrix over the reals and* $w \in \mathbb{R}^m$*. That means* $f(x)$ *is a convex quadratic cost function in* m *scalar variables,*
- $A \in \mathbb{R}^{k\times m}$ *and* $b \in \mathbb{R}^k$ *such that* $Ax = b$ *represents* k *linear equality constraints,*

3. See the work of Chen et al. (2005, 2006) for a generalization of this result to similar but more general policies for working set selection.

- *$l, r \in \mathbb{R}^m$ and $l \le x \le r$ is the short-notation for the "box-constraints"*

$$\forall i = 1, \ldots, m : l_i \le x_i \le r_i \ .$$

In this paper, we will sometimes express $f(x')$ by means of the Taylor-expansion around x:

$$f(x') = f(x) + \nabla f(x)^\top (x' - x) + \frac{1}{2}(x' - x)^\top Q(x' - x) \ ,$$

where $\nabla f(x) = Qx + w$. This can be rewritten as follows:

$$f(x) - f(x') = \nabla f(x)^\top (x - x') - \frac{1}{2}(x - x')^\top Q(x - x') \ . \tag{1}$$

Note that $(x - x')^\top Q(x - x') \ge 0$ because Q is positive semi-definite.

In the sequel,

$$R(\mathcal{P}) = \{x \in \mathbb{R}^m | \ Ax = b, l \le x \le r\}$$

denotes the compact and convex set of feasible points for $\mathcal{P}$. The well-known first-order optimality condition for convex optimization (see for example Boyd and Vandenberghe, 2004) (valid for an arbitrary convex cost function) states that $x \in R(\mathcal{P})$ is an optimal feasible solution iff

$$\forall x' \in R(\mathcal{P}) : \nabla f(x)^\top (x' - x) \ge 0 \ .$$

We briefly note that any instance of the *general* convex quadratic optimization problem with cost function $f(x)$, linear equality constraints, linear inequality constraints (not necessarily in the form of box-constraints) and a compact region of feasible points can be transformed into an equivalent instance of CQO because we may convert the linear inequalities into linear equations by introducing non-negative slack variables. By the compactness of the region of feasible points, we may also put a suitable upper bound on each slack variable such that finally all linear inequalities take the form of box-constraints.

We now define a subproblem of CQO, denoted as SVO in this paper, that is actually one of the most well studied SVM-optimization problems:

Definition 2 (SVO) *An instance $\mathcal{P}_0$ of SVO is given by*

$$\min_x f(x) \text{ s.t. } y^\top x = 0 \ , \ l \le x \le r \ ,$$

where $f(x), l, r$ are understood as in Definition 1 and $y \in \{-1, 1\}^m$ is a vector whose components represent binary classification labels.

The main difference between SVO and the general problem CQO is that SVO has only a single equality constraint. Furthermore, this equality constraint is of a special form.

We are now prepared to introduce (informally) the notion of "decomposition algorithms". A *decomposition algorithm for CQO* with working sets of size at most q (where we allow that q depends on k) proceeds iteratively as follows: given an instance $\mathcal{P}$ of CQO and a feasible solution $x \in R(\mathcal{P})$ (chosen arbitrarily in the beginning), a so-called working set $I \subseteq \{1, \ldots, m\}$ of size at most q is selected. Then x is updated by an optimal solution for the simplified instance with variables x_i, $i \in I$ (leaving the values x_j with $j \notin I$ unchanged). *Decomposition algorithms for SVO* are defined analogously. The policy for working set selection is a critical issue that we discuss in the next sections.

Notational Conventions:

- The parameters m, k, f, A, y, b, l, r are consistently used in this paper as the components of an instance of CQO or SVO. Similarly, parameter q (possibly dependent on k) always represents the (maximal) size of the working set.
- L_{max} and S_{max} are two more parameters that we associate with an instance of CQO or SVO (where L_{max} depends also on q). L_{max} denotes the largest among the eigenvalues of all the principal $(q \times q)$-sub-matrices of Q. $S_{max} := \max_{1 \leq i \leq m}(r_i - l_i)$ denotes the maximum side length of the box spanned by l and r.
- For a decomposition algorithm $\mathcal{A}$, we denote the current feasible solution obtained after n iterations by x^n (such that x^0 is the feasible solution $\mathcal{A}$ starts with[4]). x^* always denotes an optimal feasible solution such that

$$\Delta_n := f(x^n) - f(x^*) \tag{2}$$

denotes the difference between the value of the current solution and the optimal value.

2.2 Rate Certifying Pairs

Let us consider problem SVO from Definition 2 along with a problem instance $\mathcal{P}_0$. Let $x \in R(\mathcal{P}_0)$ be a feasible solution and $x^* \in R(\mathcal{P}_0)$ an optimal feasible solution. In the sequel, we will often use the following first-order approximation of the maximal distance (with regard to the value of the objective function) between a given point x and any other feasible solution x'

$$\sigma(x) \quad := \quad \sup_{x' \in R(\mathcal{P}_0)} \nabla f(x)^\top (x - x') \ .$$

As already noted by Hush and Scovel (2003), the following holds:[5]

$$f(x) - f(x^*) \leq \nabla f(x)^\top (x - x^*) \leq \sigma(x) \ . \tag{3}$$

In other words, $f(x)$ is always within $\sigma(x)$ of optimality. Note that $\sigma(x^*) = 0$ (which immediately follows from the first-order optimality condition). Thus, $\sigma(x) > 0$ if and only if x is sub-optimal.

Since we are dealing with working sets whose size is bounded by a (small) parameter q, it is natural to restrict the range of x' to feasible solutions that differ from x in at most q coordinates. For $q = 2$, this leads to the following definition:

$$\sigma(x|i_1, i_2) \quad := \quad \sup_{x' \in R(\mathcal{P}_0): x'_i = x_i \text{ for } i \neq i_1, i_2} \nabla f(x)^\top (x - x') \ .$$

The following notion is crucial: (i_1, i_2) is called an *α-rate certifying pair* for x if

$$\sigma(x|i_1, i_2) \geq \alpha(f(x) - f(x^*)) \ .$$

4. The problem of finding an initial feasible solution can be casted as an instance of Linear Programming. For an instance of SVO, an initial guess usually is the zero vector.
5. The first inequality follows from (1) and the positive semi-definiteness of Q; the second-one is trivial.

Let $\vec{\alpha}$ be a function in m with strictly positive values. An *$\vec{\alpha}$-rate certifying algorithm* is a decomposition algorithm for SVO that, for every m and every input instance $\mathcal{P}_0$ with m variables, always includes an $\vec{\alpha}(m)$-rate certifying pair in the current working set. As mentioned already in the introduction, the main results by Hush and Scovel (2003) are as follows:

Theorem 3 (Hush and Scovel (2003)) *1. Let $\mathcal{A}$ be an $\vec{\alpha}$-rate certifying algorithm. Consider any instance $\mathcal{P}_0$ of SVO with, say, m variables. Let L_{max} and S_{max} be the quantities associated with $\mathcal{P}_0$.[6] For sake of brevity, let $\alpha = \vec{\alpha}(m)$. Then, $\mathcal{A}$ is within ε of optimality after*

$$1 + \frac{2\max\{1, 2S_{max}^2\}}{\alpha}\left(\frac{\max\{1, \alpha\Delta_0/L_{max}\}L_{max}}{\alpha\varepsilon} - 1\right) = O\left(\frac{L_{max}(1+S_{max})^2}{\alpha^2\varepsilon} + \frac{\Delta_0(1+S_{max})^2}{\alpha\varepsilon}\right)$$

iterations.

2. For function $\vec{\alpha}$ given by $\vec{\alpha}(m) = 1/m^2$, there exists an $\vec{\alpha}$-rate certifying algorithm. It constructs a working set (given $\mathcal{P}_0$ and a sub-optimal feasible solution x) in $O(m\log m)$ steps (or in $O(m)$ steps when the method of Simon (2004) is applied). Furthermore, it is within ε of optimality after

$$O\left(\frac{L_{max}(1+S_{max})^2m^4}{\varepsilon} + \frac{\Delta_0(1+S_{max})^2m^2}{\varepsilon}\right)$$

iterations.

2.3 Sparse Witnesses of Sub-optimality

Consider the situation where we apply the decomposition method to a problem instance $\mathcal{P}$ of the general problem CQO from Definition 1. As suggested by List and Simon (2004), the selection of a working set can be guided by a function family $(C_I(x))_{I\subseteq\{1,\ldots,m\}}$ as follows:

Strict Rule: Given the current feasible solution x, choose the next working set $I \subseteq \{1,\ldots,m\}$ of size at most q such as to maximize $C_I(x)$.

Relaxed Rule: Given the current feasible solution x, choose the next working set $I \subseteq \{1,\ldots,m\}$ of size at most q such as to maximize $C_I(x)$ up to a factor of $0 < \rho < 1$.

List and Simon (2004) say that $(C_I(x))$ is a *q-sparse witness of sub-optimality* if it satisfies the following conditions:

(C1) For each $I \subseteq \{1,\ldots,m\}$ such that $|I| \le q$, $C_I(x)$ is continuous on $R(\mathcal{P})$.

(C2) If $|I| \le q$ and x' is an optimal solution for the subproblem induced by the current feasible solution x and working set I, then $C_I(x') = 0$.

6. See our notational conventions in Section 2 for a definition (where $q = 2$).

(C3) If x is not an optimal solution for $\mathcal{P}$, then there exists an $I \subseteq \{1,\dots,m\}$ such that $|I| \le q$ and $C_I(x) > 0$.

The following is shown by List and Simon (2004) and Simon (2004):

1. The function family

$$C_I(x) := \inf_{\lambda \in \mathbb{R}^k} \left(\sum_{i \in I} (x_i - l_i) \max\{0, \nabla f(x)_i - A_i^\top \lambda\} + (r_i - x_i) \max\{0, A_i^\top \lambda - \nabla f(x)_i\} \right) \tag{4}$$

   is a q-sparse witness of sub-optimality provided that $q \ge k+1$. Here, A_i denotes the i'th column of A and $A_i^\top$ its transpose.

2. The working set selection problem induced by the function family $C_I(x)$ from (4) is NP-complete if we insist on the strict rule. But it is solvable in polynomial time if we follow the relaxed rule with a constant $0 < \rho < 1$.

3. Let $q \ge k+1$. Assume that $Q \in \mathbb{R}^{m \times m}$ has positive definite sub-matrices $Q_{I,I}$ for all subsets $I \subseteq \{1,\dots,m\}$ of size at most q. If $(C_I(x))$ is a q-sparse witness of sub-optimality and the working set is always selected according to the relaxed rule, then the resulting feasible solutions become cost-optimal in the limit.

Note that the latter convergence result holds especially for the family $(C_I(x))$ from (4). However, it merely states convergence without guaranteeing any convergence rate. Furthermore, it requires the above assumption of positive definite sub-matrices $Q_{I,I}$. The results in the next section improve on this as follows:

- They do provide a convergence rate.
- They do not require the technical assumption about positive definite sub-matrices of Q.

3. Rate Certifying Sets and the Main Theorem

We now turn our attention to the main results of this paper. In Section 3.1, we present the new notion of rate certifying sets and state the main convergence theorem whose proof is given in Sections 3.2 and 3.3. In Section 3.4, we reveal the relation between rate certifying sets and sparse witnesses of sub-optimality.

3.1 Rate Certifying Sets

The notion of rate certifying sets generalizes the notion of rate certifying pairs from Section 2.2. The definition of $\sigma(x)$ is easily extended to any instance $\mathcal{P}$ of CQO:

$$\sigma(x) := \sup_{x' \in R(\mathcal{P})} \nabla f(x)^\top (x - x') \ . \tag{5}$$

Clearly, inequality (3) and the subsequent comments are still valid without any change. However, since CQO deals with several equality constraints, one can in general not expect to find rate certifying *pairs*. Instead, the following general definition for $I \subseteq \{1, \dots, m\}$ will prove useful:

$$\sigma(x|I) := \sup_{x' \in R(\mathcal{P}) : x'_i = x_i \text{ for } i \notin I} \nabla f(x)^\top (x - x') \ . \tag{6}$$

I is called an *α-rate certifying q-set* if $|I| \le q$ and

$$\sigma(x|I) \ge \alpha(f(x) - f(x^*)) \ . \tag{7}$$

I is called a *strong α-rate certifying q-set* (implying that it is an α-rate certifying q-set) if $|I| \le q$ and

$$\sigma(x|I) \ge \alpha\sigma(x) \ .$$

Let $\vec{\alpha}$ be a function in m with strictly positive values and let $\vec{q}$ be a function in k whose values are strictly positive integers. A (strong) $(\vec{\alpha}, \vec{q})$*-rate certifying algorithm* is a decomposition algorithm for CQO that, for every m, k and any problem instance $\mathcal{P}$ with m variables and k equality constraints, includes a (strong) $\vec{\alpha}(m)$-rate certifying $\vec{q}(k)$-set in the current working set. With these notations, the following holds:

Theorem 4

1. *Let $\mathcal{A}$ be an $(\vec{\alpha}, \vec{q})$-rate certifying algorithm. Consider an instance $\mathcal{P}$ of CQO with, say, m variables and k equality constraints. For sake of brevity, let $\alpha = \vec{\alpha}(m)$, $q = \vec{q}(k)$, and let L_{max}, S_{max} be the quantities associated with $\mathcal{P}$ and q. Then, $\mathcal{A}$ is within ε of optimality after*

$$\left\lceil \frac{2qL_{max}S_{max}^2}{\alpha^2\varepsilon} \right\rceil + \max\left\{0, \left\lceil \frac{2}{\alpha} \ln\left(\frac{\Delta_0}{\varepsilon}\right) \right\rceil \right\} \tag{8}$$

   iterations. Moreover, if $qL_{max}S_{max}^2 \le \varepsilon\alpha$, then $\max\{0, \lceil 2\ln(\Delta_0/\varepsilon)/\alpha \rceil\}$ iterations (the second term in (8)) are enough.

2. *For functions $\vec{\alpha}, \vec{q}$ given by $\vec{\alpha}(m) = 1/m$ and $\vec{q}(k) = k+1$, there exists a strong $(\vec{\alpha}, \vec{q})$-rate certifying algorithm $\mathcal{A}$.*[7] *It constructs a working set (given $\mathcal{P}$ and a sub-optimal feasible solution x) in polynomial time. Moreover, if we restrict its application to instances of CQO with at most k_0 equality constraints for some fixed constant k_0, there is a linear time bound for the construction of the working set.*[8]

3. *The algorithm $\mathcal{A}$ from the preceding statement is within ε of optimality after*

$$\left\lceil \frac{2(k+1)m^2L_{max}S_{max}^2}{\varepsilon} \right\rceil + \max\left\{0, \left\lceil 2m\ln\left(\frac{\Delta_0}{\varepsilon}\right) \right\rceil \right\}$$

   iterations. Moreover, if $(k+1)L_{max}S_{max}^2 \le \varepsilon/m$, then $\max\{0, \lceil 2m\ln(\Delta_0/\varepsilon) \rceil\}$ iterations are enough.

---

7. As for the subproblem SVO and the special case $q = 2$, $\vec{\alpha}(m) = 1/m$ can be slightly improved to $\vec{\alpha}(m) = 1/(m-1)$ and this is optimal (Hush et al., 2006).
8. If we restrict its application to instances of SVO, we may use the highly efficient method of Simon (2004).

Clearly, the third statement in Theorem 4 follows directly from the first two statements (which will be proven in Subsections 3.2 and 3.3, respectively).

A few comments on Theorem 4 are in place here. One might be tempted to think that an $(\vec{\alpha}, \vec{q})$-rate certifying algorithm decreases (an upper bound on) the distance between the current feasible solution and the best feasible solution (with regard to the objective value) roughly by factor $1 - \alpha$ (for $\alpha := \vec{\alpha}(m)$). If such a "contraction" took place, we would be within ε of optimality after only $O(\log(1/\varepsilon)/\alpha)$ iterations. This is however spurious thinking because the σ-function is *not* concerned with this distance itself but rather with a first-order approximation of it. We will see in the proof of Theorem 4 that a run of an $(\vec{\alpha}, \vec{q})$-rate certifying algorithm can be decomposed into two phases. As long as the distance from the optimal value is large in comparison to (an upper bound on) the second order terms (phase 1), a contraction by factor $1 - \alpha/2$ takes place. However when we come closer to the optimal value (phase 2), the effect of the neglected second order terms becomes more significant and the convergence slows down (at least within our perhaps somewhat pessimistic analysis). Phase 1 leads to the term $\max\{0, \lceil 2\ln(\Delta_0/\varepsilon)/\alpha \rceil\}$ in (8) whereas phase 2 leads to the term $\lceil (2qL_{max}S_{max}^2)/(\alpha^2\varepsilon) \rceil$.

3.2 Proof of the 1st Statement in Theorem 4

We will use the notation introduced in Theorem 4 and consider an arbitrary iteration of the $(\vec{\alpha}, \vec{q})$-rate certifying algorithm $\mathcal{A}$ when it is applied on input $\mathcal{P}$. To this end, let x denote a feasible but sub-optimal solution, and let x^* be an optimal feasible solution. Let I be the subset of the working set that satisfies $|I| \le q$ and (7). Let x' be a feasible solution that satisfies $x_i = x'_i$ for every $i \notin I$ and

$$\sigma(x|I) = \nabla f(x)^\top (x - x') = \nabla f(x)^\top d \ , \tag{9}$$

where $d = x - x'$. Combining (7) with (9), we get

$$\nabla f(x)^\top d \ge \alpha(f(x) - f(x^*)) = \alpha\Delta \ , \tag{10}$$

where $\Delta = f(x) - f(x^*) \ge 0$. For some parameter $0 \le \lambda \le 1$ (suitably chosen later), consider the feasible solution

$$x'' = x - \lambda d$$

on the line segment between x and x'. Taylor-expansion around x allows to relate $f(x)$ and $f(x'')$ as follows:

$$f(x) - f(x'') = \lambda \nabla f(x)^\top d - \lambda^2 \frac{1}{2} d^\top Q d \ .$$

Note that x'' (like x') satisfies $x''_i = x_i$ for every $i \notin I$. Thus, $x'' = x - \lambda d$ is a feasible solution that coincides with x outside the current working set. Thus, $\mathcal{A}$ (finding an optimal feasible solution that coincides with x outside the working set) achieves in the next iteration a "cost reduction" of at least $f(x) - f(x'')$.[9] x'' depends on the parameter $0 \le \lambda \le 1$. In the sequel, we tune parameter λ such as to obtain a "large" cost reduction. To be on the

9. "Achieving a cost reduction of a in the next iteration" means that the next iteration decreases the distance between the value of the current feasible solution and the value of an optimal feasible solution by at least a.

safe side, we will however perform a "pessimistic analysis" where we substitute worst case bounds for $\nabla f(x)^\top d$ and $d^\top Qd$ respectively. Clearly

$$\max_{0\le\lambda\le 1}\left(\lambda\nabla f(x)^\top d-\lambda^2\frac{1}{2}d^\top Qd\right)\ge\max_{0\le\lambda\le 1}\left(\lambda B-\frac{1}{2}\lambda^2B'\right)$$

for any lower bound B on $\nabla f(x)^\top d$ and any upper bound B' on $d^\top Qd$. According to (10), $\alpha\Delta$ can serve as a lower bound on $\nabla f(x)^\top d$. It is easily seen that the following parameter U can serve as an upper bound on $d^\top Qd$:

$$U := qL_{max}S_{max}^2 \ge d^\top Qd \ . \tag{11}$$

This immediately follows from the definition of L_{max} and S_{max} and from the fact that d has at most q non-zero components. We conclude from this discussion that, for every $0\le\lambda\le 1$, $\mathcal{A}$ achieves in the next iteration a cost reduction of at least

$$h(\lambda) := \lambda\alpha\Delta-\frac{1}{2}\lambda^2U \ .$$

It is easily seen that function $h(\lambda)$ is maximized by setting

$$\lambda := \begin{cases} 1 & \text{if } \Delta > U/\alpha \\ \alpha\Delta/U & \text{if } \Delta\le U/\alpha \end{cases} \ .$$

Case 1 $\Delta > U/\alpha$. Then $\mathcal{A}$ achieves a cost reduction of at least

$$h(1)=\alpha\Delta-\frac{1}{2}U > \alpha\Delta/2 \ .$$

Thus the difference $\Delta = f(x)-f(x^*)$ will shrink after the next iteration to

$$\Delta-\frac{\alpha}{2}\Delta=\Delta\left(1-\frac{\alpha}{2}\right) \ ,$$

or to a smaller value, which is a proper contraction.

Case 2 $\Delta\le U/\alpha$. Then $\mathcal{A}$ achieves a cost reduction of at least

$$h\left(\frac{\alpha\Delta}{U}\right)=\frac{\alpha^2\Delta^2}{2U}=\gamma\Delta^2 \ ,$$

where

$$\gamma := \frac{\alpha^2}{2U} \ . \tag{12}$$

Thus, the difference $\Delta = f(x)-f(x^*)$ will shrink after the next iteration to

$$\Delta-\gamma\Delta^2$$

or to a smaller value.

Recall from (2) that the sequence $(\Delta_n)_{n\ge 0}$ keeps track of the difference between the value of the current feasible solution and the value of an optimal solution. In view of the two cases described above, our pessimistic analysis obviously runs through two phases:

Phase 1 As long as $\Delta_n > U/\alpha$, we calculate with a cost reduction of at least $\alpha\Delta_n/2$.

Phase 2 As soon as $\Delta_n \le U/\alpha$, we calculate with a cost reduction of at least $\gamma\Delta_n^2$.

Let us first assume that $\varepsilon < U/\alpha$ (and postpone the case $\varepsilon \ge U/\alpha$ to the end of this subsection). The number of iterations in phase 1 is not larger than the smallest $n_0 \ge 0$ that satisfies the second inequality in

$$\Delta_0 \left(1 - \frac{\alpha}{2}\right)^{n_0} < \Delta_0 e^{-n_0\alpha/2} \le \varepsilon < \frac{U}{\alpha} \ ,$$

that is

$$n_0 := \max\left\{0, \left\lceil \frac{2}{\alpha} \ln\left(\frac{\Delta_0}{\varepsilon}\right)\right\rceil\right\} \ . \tag{13}$$

In phase 2, $(\Delta_n)_{n\ge n_0}$ evolves according to

$$\Delta_{n+1} \le \Delta_n - \gamma\Delta_n^2 = \Delta_n(1-\gamma\Delta_n) \ . \tag{14}$$

Recall that $\Delta_i = f(x^i) - f(x^*) \ge 0$ for every i. As for the iterations considered in phase 2 within our analysis, we can make the stronger (pessimistic) assumption $\Delta_i > 0$.[10] Following Dunn (1979), we can therefore consider the reciprocals $\delta_n := 1/\Delta_n$. Note that (14) and $\Delta_{n+1} > 0$ imply that $0 \le \gamma\Delta_n < 1$ and so for each $n \ge n_0$ the following relation holds:

$$\delta_{n+1} - \delta_n \ge \frac{1}{\Delta_n(1-\gamma\Delta_n)} - \frac{1}{\Delta_n} = \frac{\gamma}{1-\gamma\Delta_n} \ge \gamma \ .$$

Therefore

$$\delta_n = \delta_{n_0} + \sum_{j=n_0}^{n-1}(\delta_{j+1} - \delta_j) \ge \gamma(n - n_0)$$

and consequently

$$\Delta_n = \frac{1}{\delta_n} \le \frac{1}{\gamma(n-n_0)} \ .$$

It follows that $\Delta_n \le \varepsilon$ after

$$n - n_0 := \left\lceil \frac{1}{\gamma\varepsilon} \right\rceil$$

iterations in phase 2. Thus the total number of iterations in both phases needed to be within ε of optimality is bounded by

$$n_0 + \left\lceil \frac{1}{\gamma\varepsilon} \right\rceil \overset{(12),(13)}{\le} \left\lceil \frac{2U}{\alpha^2\varepsilon} \right\rceil + \max\left\{0, \left\lceil \frac{2}{\alpha}\ln\left(\frac{\Delta_0}{\varepsilon}\right)\right\rceil\right\}$$

iterations. Plugging in the definition of U from (11), we obtain (8).
Let us now finally discuss the case $U = qL_{max}S_{max}^2 \le \varepsilon\alpha$. Since $\varepsilon \ge U/\alpha$, we come within ε of optimality during phase 1. The number of iterations required for this is the smallest n_0 that satisfies

$$\Delta_0 \left(1 - \frac{\alpha}{2}\right)^{n_0} < \Delta_0 e^{-n_0\alpha/2} \le \varepsilon \ .$$

10. Otherwise phase 2 ends with an optimal solution even earlier.

Thus, we are within ε of optimality after

$$\max\left\{0, \left\lceil \frac{2}{\alpha} \ln\left(\frac{\Delta_0}{\varepsilon}\right)\right\rceil\right\}$$

iterations. This completes the proof of the first statement in Theorem 4.

3.3 Proof of the 2nd Statement in Theorem 4

Before we present the proof, we recall some definitions and facts from the theory of Linear Programming.[11]

An *instance of Linear Programming* is given by a linear target function $c^\top x$ that has to be minimized (or maximized) subject to some linear equality and inequality constraints. A vector that satisfies the constraints is called a *feasible solution*. The instance is called *feasible* if it has a feasible solution. It is called *bounded* if it has a finite optimal value. An instance of the form

$$\max_x c^\top x \text{ s.t. } Ax = b\ ,\ \ x \geq 0\ \ ,$$

where $b, c \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m \times n}$, $m, n \in \mathbb{N}$, is said to be in *standard form*. We can assume that A has rank m (after removal of redundant equations if necessary). An $(m \times m)$-sub-matrix B of A formed by m linearly independent columns of A, say $A_{i_1}, \ldots, A_{i_m}$, $1 \leq i_1 < \cdots < i_m \leq n$, is called a *basis*. The *basic solution corresponding to* B is obtained by setting $x_i = 0$ for every $i \notin \{i_1, \ldots, i_m\}$ and by setting x_{i_l} equal to the l'th component of $B^{-1}b$ for $l = 1, \ldots, m$. Note that any basic solution (corresponding to some basis) has at most $m = \text{rank}(A)$ many nonzero components. If a basic solution happens to be feasible, it is called a *basic feasible solution*. It is well-known that, for every feasible and bounded instance of Linear Programming in standard form, there exists a basic feasible solution that is optimal (among all feasible solutions).

We are now prepared for the proof of the second statement in Theorem 4. We start with a short outline. Let x be a feasible but sub-optimal solution for $\mathcal{P}$. We have to show that one can efficiently construct a working set I such that $|I| \leq k+1$ and

$$\sigma(x|I) \geq \frac{1}{m}\sigma(x)\ \ . \tag{15}$$

To this end, we will proceed as follows. We consider auxiliary instances $\mathcal{P}_x, \mathcal{P}'_x, \mathcal{P}''_x$ of Linear Programming (denoted as LP in the sequel). The optimal values for $\mathcal{P}_x$ and $\mathcal{P}'_x$, are both shown to coincide with $\sigma(x)$. From $\mathcal{P}'_x$, we derive (basically by aggregating several equality constraints into a single-one) the instance $\mathcal{P}''_x$, whose optimal basic solution will represent a working set I of size at most $k+1$. A comparison of the three problem instances will finally reveal that I satisfies (15).

We now move on to the technical implementation of this plan. Recall from (5) that

$$\sigma(x) = \sup_{x' \in R(\mathcal{P})} \nabla f(x)^\top (x - x') = \nabla f(x)^\top d\ \ ,$$

11. See any standard text book about Combinatorial Optimization (for example Papadimitriou and Steiglitz, 1982).

where $d = x - x'$. Thus $\sigma(x)$ is the optimal value of the following instance $\mathcal{P}_x$ of LP:

$$\max_d \nabla f(x)^\top d \text{ s.t. } Ad = \vec{0}, l \le x - d \le r \ .$$

We set

$$\mu^+ := x - l \text{ and } \mu^- := r - x$$

and split d into positive and non-positive components d^+ and d^- respectively:

$$d = d^+ - d^-,\ d^+, d^- \ge \vec{0},\ \forall i = 1, \ldots, m : d_i^+ d_i^- = 0 \ . \tag{16}$$

With these notations, $\sigma(x)$ also coincides with the optimal value of the following instance $\mathcal{P}'_x$ of LP:

$$\max_{d^+, d^-} \begin{pmatrix} \nabla f(x) \\ -\nabla f(x) \end{pmatrix}^\top \begin{pmatrix} d^+ \\ d^- \end{pmatrix}$$

subject to

$$\begin{aligned} [A, -A] \begin{pmatrix} d^+ \\ d^- \end{pmatrix} &= \vec{0} \\ \vec{0} \le d^+ \le \mu^+ \quad &, \quad \vec{0} \le d^- \le \mu^- \end{aligned}$$

The third instance $\mathcal{P}''_x$ of LP that we consider has an additional slack variable ξ and is given as follows:

$$\max_{d^+, d^-, \xi} \begin{pmatrix} \nabla f(x) \\ -\nabla f(x) \end{pmatrix}^\top \begin{pmatrix} d^+ \\ d^- \end{pmatrix}$$

subject to

$$\forall i = 1, \ldots, m : \mu_i^- = 0 \Rightarrow d_i^- = 0 \quad , \quad \mu_i^+ = 0 \Rightarrow d_i^+ = 0 \tag{17}$$

$$[A, -A] \begin{pmatrix} d^+ \\ d^- \end{pmatrix} = \vec{0} \tag{18}$$

$$\sum_{i:\mu_i^+ > 0} \frac{1}{\mu_i^+} d_i^+ + \sum_{i:\mu_i^- > 0} \frac{1}{\mu_i^-} d_i^- + \xi = 1 \tag{19}$$

$$d^+, d^- \ge \vec{0} \quad , \quad \xi \ge 0 \tag{20}$$

We briefly note that we do not have to count the equality constraints in (17) because we may simply remove the variables that are set to zero from the problem instance. Recall that matrix A represents k equality constraints. Thus, $\mathcal{P}''_x$ is a linear program in standard form with $k + 1$ equality constraints. Its optimal basic feasible solution has therefore at most $k + 1$ non-zero components. The following observations (where d, d^+, d^- are related according to (16)) are easy to verify:

1. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}'_x$ with value p, then $\frac{1}{m} \begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}''_x$ with value p/m.

2. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}''_x$ with value p, then $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is also a feasible solution for $\mathcal{P}'_x$ with value p.

Recall that $\sigma(x)$ is the value of an optimal solution for $\mathcal{P}'_x$. We may conclude from our observations that the value of an optimal solution for $\mathcal{P}''_x$, say $\sigma'(x)$, satisfies $\sigma'(x) \geq \sigma(x)/m$. Now consider an optimal basic feasible solution $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ for $\mathcal{P}''_x$ with value $\sigma'(x)$. Let

$$I := \{i \in \{1, \ldots, m\} |\ d_i^+ \neq 0 \text{ or } d_i^- \neq 0\} \ .$$

Clearly, $|I| \leq k+1$. Since $d = d^+ - d^-$ is a feasible solution for $\mathcal{P}$ (still with value $\sigma'(x)$) that differs from x only in coordinates from I, we may conclude that

$$\sigma(x|I) \geq \sigma'(x) \geq \frac{1}{m}\sigma(x) \ .$$

In other words, working set I satisfies (15). The time required to compute I is dominated by the time required to solve the linear program $\mathcal{P}''_x$. This can be done in polynomial time. Since a linear program with a constant number of variables (or a linear program in standard form with a constant number of equality constraints[12]) can be solved in linear time (Megiddo, 1984), the proof for the 2nd statement of Theorem 4 is completed.

3.4 Relation between Certifying Sets and Sparse Witnesses of Sub-optimality

Before we come to the main issue of this section, we briefly recall some well-known definitions and facts from the theory of Linear Programming (see Papadimitriou and Steiglitz, 1982, Definition 3.1 and Theorems 3.1 and 3.2).

For every (primal) instance of Linear Programming, one can construct the so-called *dual instance* according to the following scheme:

primal		dual
$\min_x c^\top x$		$\max_\lambda \lambda^\top b$
$a_i^\top x = b_i$	$i \in M$	—
$a_i^\top x \geq b_i$	$i \in \bar{M}$	$\lambda_i \geq 0$
$x_j \geq 0$	$j \in N$	$\lambda^\top A_j \leq c_j$
—	$j \in \bar{N}$	$\lambda^\top A_j = c_j$

In this scheme, $A \in \mathbb{R}^{m \times n}$ is a matrix with column vectors $A_1, \ldots, A_n$ and row vectors $a_1^\top, \ldots, a_m^\top$, $N \subseteq \{1, \ldots, n\}$ is the set of indices corresponding to primal variables that are constrained to be non-negative, and $M \subseteq \{1, \ldots, m\}$ is the set of indices corresponding to equality constraints (as opposed to inequality constraints) in the primal. It is well known that the optimal values of the primal and the dual coincide (assuming that the optimal values exist). Moreover, the dual of the dual coincides with the primal.

Let us now return to the main issue of this section. Consider again the function $\sigma(x|I)$ from (6) and the function $C_I(x)$ from (4). Here comes the main result of this section:

12. That means the dual linear program has constant number of variables.

Lemma 5 $\sigma(x|I) = C_I(x)$.

Proof Given a subset $I \subset \{1, \ldots, m\}$, $|I| = q$, and a vector $x \in \mathbb{R}^m$, we will write shortly $x_I \in \mathbb{R}^q$ for the vector consisting only of entries $x_i, i \in I$. Recall that $A_i \in \mathbb{R}^k$ denotes the i-th column of matrix A and write $A_I \in \mathbb{R}^{k \times q}$ for the matrix consisting of columns A_i such that $i \in I$.
An inspection of (6) reveals that $\sigma(x|I)$ can equivalently be written as follows:

$$\sigma(x|I) = \sup_{d \in \mathbb{R}^q} \left\{ \nabla f(x)_I^\top d \mid A_I d = 0,\ l_I \le x_I - d \le r_I \right\} .$$

This is an (obviously feasible and bounded) instance of Linear Programming with the following inequality constraints

$$\begin{pmatrix} \mathrm{Id}_q \\ -\mathrm{Id}_q \end{pmatrix} d \le \begin{pmatrix} x_I - l_I \\ r_I - x_I \end{pmatrix} ,$$

where $\mathrm{Id}_q \in \mathbb{R}^{q \times q}$ denotes the identity-matrix. According to LP-duality, $\sigma(x|I)$ can also be viewed as the minimum cost of the dual minimization problem:

$$\sigma(x|I) = \inf_{\substack{\alpha, \beta \in \mathbb{R}^q \\ \lambda \in \mathbb{R}^k}} \left\{ \begin{pmatrix} \alpha \\ \beta \end{pmatrix}^\top \begin{pmatrix} x_I - l_I \\ r_I - x_I \end{pmatrix} \,\middle|\, A_I^\top \lambda + \alpha - \beta = \nabla f(x)_I,\ \alpha, \beta \ge 0 \right\} . \qquad (21)$$

It is easy to see that the optimal choice of α and β is as follows:

$$\alpha_i = \max\{0, \nabla f(x)_i - A_i^\top \lambda\} \qquad \text{and} \qquad \beta_i = \max\{0, A_i^\top \lambda - \nabla f(x)_i\} .$$

Plugging these settings into (21), we finally get

$$\begin{aligned} &\sigma(x|I) = \\ &\quad \inf_{\lambda \in \mathbb{R}^k} \sum_{i \in I} \Big((x_i - l_i) \max\{0, \nabla f(x)_i - A_i^\top \lambda\} + (r_i - x_i) \max\{0, A_i^\top \lambda - \nabla f(x)_i\} \Big) \\ &\quad = C_I(x) . \end{aligned}$$

■

Lemma 5 is remarkable because the coincidence of the two functions was not known before and their definitions originated from different motivations:

- $C_I(x)$ was designed such as to satisfy the axioms for sparse witnesses of sub-optimality.
- $\sigma(x|I)$ was designed as a generalization of the corresponding function for rate certifying pairs.

Definition 6 (WSS) *The problem "Working Set Selection for CQO", denoted briefly as WSS(CQO), is the following computational problem: given an instance of CQO (see Definition 1) augmented by a feasible solution x and an upper bound $q \ge k + 1$ on the size of the working set, find a working set $I_* \subseteq \{1, \ldots, n\}$ that maximizes $\sigma(x|I)$ subject to $|I| \le q$.*

An approximation algorithm for WSS(CQO) is called ρ-optimal[13] *if, for every instance of WSS(CQO), it finds a working set I_0 such that $|I_0| \le q$ and $\sigma(x|I_0) \ge \rho(m) \max_{I:|I|\le q} \sigma(x|I)$. The corresponding notions for the subproblem WSS(SVO) are defined analogously.*

Rate certifying algorithms and approximation algorithms for WSS(CQO) are closely related:

Corollary 7 *The following holds for WSS(CQO):*[14]

1. *A strong $(\vec{\alpha}, k+1)$-rate certifying algorithm can be viewed as an $\vec{\alpha}(m)$-optimal approximation algorithm for WSS(CQO).*
2. *An approximation algorithm for WSS(CQO) with performance ratio $\rho(m)$ can be viewed as a strong $(\vec{\alpha}, k+1)$-rate certifying algorithm where $\vec{\alpha}(m) = \rho(m)/m$.*

Proof

1. A strong $(\vec{\alpha}, k+1)$-rate certifying algorithm applied to an instance of WSS(CQO) constructs a working set I_0 of size at most $k+1$ such that $\sigma(x|I_0) \ge \vec{\alpha}(m)\sigma(x)$. Clearly, $\sigma(x) \ge \max_{I:|I|\le q} \sigma(x|I)$. Thus, $\sigma(x|I_0)$ matches the optimal value up to factor $\vec{\alpha}(m)$.
2. An approximation algorithm for WSS(CQO) can be used as the procedure within a decomposition algorithm that performs the working set selection. At every iteration, it is applied to an instance of WSS(CQO) where $q = k+1$. It then constructs a working set I_0 such that $|I_0| \le q$ and $\sigma(x|I_0) \ge \rho(m) \max_{I:|I|\le k+1} \sigma(x|I)$. We know from Theorem 4 that $\max_{I:|I|\le k+1} \sigma(x|I) \ge \sigma(x)/m$. Thus $\sigma(x|I_0)$ matches the optimal value up to factor $\rho(m)/m$.

■

Corollary 7 immediately imlplies that the strong $(1/m, k+1)$-rate certifying algorithm described in Section 3.3 can be viewed as a $1/m$-optimal approximation algorithm for WSS(CQO). This result can however be strengthened:

Corollary 8 *The strong $(1/m, k+1)$-rate certifying algorithm described in Section 3.3 can be viewed as a $1/q$-optimal approximation algorithm for WSS(CQO).*

Proof We use again the notations (like, for example, $\mathcal{P}'_x, \mathcal{P}''_x$) that were introduced in Section 3.3. Every instance of WSS(CQO) can be casted as an instance $\mathcal{P}'_{x,q}$ resulting from an instance $\mathcal{P}'_x$ by adding the constraint that the total number of non-zero components in d^+, d^- is bounded by q. The algorithm from Section 3.3 returns an optimal solution for the instance $\mathcal{P}''_x$. The corollary is now an immediate consequence of the following observations:

1. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}'_{x,q}$ with value p, then $\frac{1}{q}\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a feasible solution for $\mathcal{P}''_x$ with value p/q.

13. Here, ρ may be a function in the input parameters.
14. The analogous statements for WSS(SVO) hold as well.

2. If $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is a basic feasible solution for $\mathcal{P}''_x$ with value p, then $\begin{pmatrix} d^+ \\ d^- \end{pmatrix}$ is also a feasible solution for $\mathcal{P}'_{x,q}$ with value p.

■

Our results nicely strengthen (or complement) what was known before about working set selection w.r.t. $C_I(x)$:

- List and Simon (2004) have shown that the "strict rule" of always selecting a working set I_* that maximizes $C_I(x)$ subject to $|I| \le q$ leads to cost-optimal feasible solutions in the limit (but no convergence rate was provided). Simon (2004) has shown a similar result for the "relaxed rule". Since $C_I(x) = \sigma(x|I)$, Theorem 4 extends these mere convergence results by providing a convergence rate.

- Simon (2004) has shown that (a canonical combinatorial abstraction of) WSS(SVO) is NP-complete, but admits a full polynomial time approximation scheme. According to Corollary 8, the algorithm described in Section 3.3 is an approximation algorithm with performance ratio $1/m$ for the more general problem WSS(CQO).

4. Conclusions and Open Problems

We have presented an analysis of a decomposition algorithm that leads to the to-date strongest theoretical performance guarantees within the "rate certifying pair" approach. Our analysis holds uniformly for any instance of Convex Quadratic Optimization (with box-constraints) and certainly covers most of the variants of SVM-optimization. Our results can be seen as an extension of (and an improvement on) earlier work by Hush and Scovel (2003) about rate certifying pairs, and by List and Simon (2004) and Simon (2004) about sparse witnesses of sub-optimality.
As explained in the introduction already, there are competing approaches like, for example, the approach based on maximally KKT-violating pairs or approaches based on an iterative inclusion of support vectors. Recent experiments by Hush et al. (2006) indicate that

- the approach based on maximally KKT-violating pairs converges faster than the approach based on rate certifying pairs,

- but a hybrid approach that makes a greedy choice among the maximally KKT-violating pair and pairs that come up in the rate certifying pair approach is often superior.

At the time being, these experimental results are not supported by theoretical analysis. We leave the task of providing a theoretical explanation for the empirically good performance of the hybrid approach (or even the original approach based on KKT-violating pairs) as an object of future research.

Acknowledgments

This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views. This work was furthermore supported by the Deutsche Forschungsgemeinschaft Grant SI 498/7-1.

References

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

Steven Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Chih-Chung Chang, Chih-Wei Hsu, and Chih-Jen Lin. The analysis of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 11(4):248–250, 2000.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Available from http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

Pai-Hsuen Chen, Rong-En Fan, and Chih-Jen Lin. Training support vector machines via SMO-type decomposition methods. In *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, pages 45–62, 2005.

Pai-Hsuen Chen, Rong-En Fan, and Chih-Jen Lin. A study on SMO-type decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 17(4): 893–908, 2006.

J. Dunn. Rates of convergence for conditional gradient algorithms near singular and non-singular extremals. *SIAM J. Control and Optimization*, 17(2):187–211, 1979.

Chih-Wei Hsu and Chih-Jen Lin. A simple decomposition method for support vector machines. *Machine Learning*, 46(1–3):291–314, 2002.

Don Hush, Patrick Kelly, Clint Scovel, and Ingo Steinwart. QP Algorithms with Guaranteed Aaccuracy and Run Time for Support Vector Machines. *Journal of Machine Learning Research*, 7:733–769, May 2006.

Don Hush and Clint Scovel. Polynomial-time decomposition algorithms for support vector machines. *Machine Learning*, 51(1):51–71, 2003.

Thorsten Joachims. Making large scale SVM learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*, pages 169–184. MIT Press, 1998.

S. Sathiya Keerthi and E. G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning*, 46(1–3):351–360, 2002.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and K. R. K. Murthy. Improvements to SMO algorithm for SVM regression. *IEEE Transactions on Neural Networks*, 11(5):1188–1193, 2000.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13(3):637–649, 2001.

Pavel Laskov. Feasible direction decomposition algorithms for training support vector machines. *Machine Learning*, 46(1–3):315–349, 2002.

Shuo-Peng Liao, Hsuan-Tien Lin, and Chih-Jen Lin. A note on the decomposition methods for support vector regression. *Neural Computation*, 14(6):1267–1281, 2002.

Chih-Jen Lin. Linear convergence of a decomposition method for support vector machines, 2001a. Available from http://www.csie.ntu.edu.tw/∼cjlin/papers/linearconv.pdf.

Chih-Jen Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 12(6):1288–1298, 2001b.

Chih-Jen Lin. Asymptotic convergence of an SMO algorithm without any assumptions. *IEEE Transactions on Neural Networks*, 13(1):248–250, 2002a.

Chih-Jen Lin. A formal analysis of stopping criteria of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 13(5):1045–1052, 2002b.

Nikolas List. Convergence of a generalized gradient selection approach for the decomposition method. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, pages 338–349, 2004.

Nikolas List and Hans Ulrich Simon. A general convergence theorem for the decomposition method. In *Proceedings of the 17th Annual Conference on Computational Learning Theory*, pages 363–377, 2004.

Olvi L. Mangasarian and David R. Musicant. Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, 10(5):1032–1037, 1999.

Olvi L. Mangasarian and David R. Musicant. Active support vector machine classification. In *Advances in Neural Information Processing Systems 12*, pages 577–583. MIT Press, 2000.

Olvi L. Mangasarian and David R. Musicant. Lagrangian support vector machines. *Journal of Machine Learning Research*, 1:161–177, 2001.

Nimrod Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the Association on Computing Machinery*, 31(1):114–127, 1984.

Edgar E. Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

Christos H. Papadimitriou and Kenneth Steiglitz *Combinatorial Optimization*. Prentice Hall, 1982.

John C. Platt. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods—Support Vector Learning*, pages 185–208. MIT Press, 1998.

Craig Saunders, Mark O. Stitson, Jason Weston, Leon Bottou, Bernhard Schölkopf, and Alexander J. Smola. Support vector machine reference manual. Technical Report CSD-TR-98-03, Royal Holloway, University of London, Egham, UK, 1998.

Hans Ulrich Simon. On the complexity of working set selection. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, pages 324–337, 2004. A full version of the paper will appear in TCS.

Vladimir Vapnik. *Statistical Learning Theory*. Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons, 1998.

S. V. N. Vishwanthan, Alexander J. Smola, and M. Narasimha Murty. SimpleSVM. In *Proceedings of the 20th International Conference on Machine Learning*, pages 760–767, 2003.