A decomposition algorithm model for singly linearly constrained problems subject to lower and upper bounds

Chih-Jen Lin*,
S. Lucidi**, L. Palagi**,
A. Risi***,
M. Sciandrone****

* Department of Computer Science
National Taiwan University - Taiwan

** Dipartimento di Informatica e Sistemistica “Antonio Ruberti”
Università di Roma “La Sapienza” - Italy

*** Istituto di Analisi dei Sistemi ed Informatica “Antonio Ruberti”
Consiglio Nazionale delle Ricerche - Italy

**** Dipartimento di Sistemi e Informatica
Università di Firenze - Italy

e-mail (Lin): cjlin@csie.ntu.edu.tw
e-mail (Lucidi): lucidi@dis.uniroma1.it
e-mail (Palagi): palagi@dis.uniroma1.it
e-mail (Risi): risi@iasi.cnr.it
e-mail (Sciandrone): sciandro@dsi.unifi.it


1This work was supported in part by MIUR, Research program PRIN 2005 Problemi e metodi innovativi nell’ottimizzazione non lineare, Italy.
ABSTRACT

Many real applications can be formulated as nonlinear minimization problems with a single linear equality constraint and box constraints. We are interested in solving problems where the number of variables is so huge that basic operations, such as the updating of the gradient or the evaluation of the objective function, are very time consuming. Thus, for the considered class of problems (including dense quadratic programs) traditional optimization methods cannot be directly applied. In this paper we define a decomposition algorithm model which employs, at each iteration, a descent search direction selected among a suitable set of sparse feasible directions. The algorithm is characterized by an acceptance rule of the updated point which, from one hand permits to choose the variables to be modified with a certain degree of freedom, from the other one does not require the exact solution of any subproblem. The global convergence of the algorithm model is proved under the assumption that the objective function is continuously differentiable. Numerical results on large-scale quadratic problems arising in the training of support vector machines show the effectiveness of an implemented decomposition scheme derived from the general algorithm model.

Keywords: large scale optimization, decomposition methods, Support Vector Machines.

1 Introduction

Let us consider the problem

\[
\min \quad f(x) \\
\begin{align*}
a^T x &= b \\
l &\leq x &\leq u,
\end{align*}
\]  

where \(x \in \mathbb{R}^n\), \(f : \mathbb{R}^n \rightarrow \mathbb{R}\) is a continuously differentiable function and \(a, l, u \in \mathbb{R}^n\), with \(l < u\), \(b \in \mathbb{R}\). We allow the possibility that some of the variables are unbounded by permitting both \(l_i = -\infty\) and \(u_i = \infty\) for some \(i \in \{1, \ldots, n\}\). Moreover, we assume, without loss of generality that \(a_i \neq 0\) for all \(i = 1, \ldots, n\), thought our approach can be extended with minor modifications to include the case where \(a_i = 0\) for some \(i\).

We focus the attention on large dimensional problems. We suppose that the dimension \(n\) is so large that traditional optimization methods cannot be directly employed since basic operations, such as the updating of the gradient or the evaluation of the objective function, are too time consuming.
There are many real applications that can be modelled by optimization problems of the form (1). For instance, optimal control problems, portfolio selection problems, traffic equilibrium problems, multicommodity network flow problems (see, e.g., [1], [5], [10], [13]) are specific instances of (1). Further, a continuous formulation of a classical problem in graph theory, the maximum clique problem, is formulated as (1) with indefinite quadratic function (see e.g. [2]).

Moreover, an important machine learning methodology, called Support Vector Machine (SVM) [23], leads to huge problems of the form (1) with quadratic objective function. Different kinds of algorithms have been developed for large dimensional SVM training problems. Among them there are interior point methods [8], semismooth algorithms [9], methods based on unconstrained smooth reformulations [14], [20], active set methods [19, 22], projected gradient methods [7], decomposition methods (see, e.g., [16, 17, 18, 24]).

Here we focus our attention on a decomposition-based approach which, involving at each iteration the updating of a small number of variables, is suitable for large problems with dense hessian matrix. In a general decomposition framework, at the generic iteration $k$ the components of the current feasible point $x^k$ are partitioned into two subsets $W^k$, usually called working set, and $\overline{W}^k = \{1, \ldots, l\} \setminus W^k$ (for notational convenience from now on we omit the dependence of $W, \overline{W}$ on $k$). The variables corresponding to $W$ are unchanged at the new iteration, while the components corresponding to $\overline{W}$ are set equal to those of a stationary point $x^*_W$ of the subproblem

$$\min f(x_W, x_W^k)$$
$$a_W^T x_W = -a_W^T x_W^k + b$$
$$l_W \leq x_W \leq u_W$$

Thus the new iterate is $x^{k+1} = (x^{k+1}_W, x^{k+1}_{\overline{W}}) = (x^*_W, x^k_{\overline{W}})$.

In order to guarantee the global convergence of a decomposition method the working set selection cannot be arbitrary but must satisfy suitable rules (see e.g. [3, 16, 21]). However, up to our knowledge, the proposed convergent decomposition algorithms and their convergence analysis are based on the assumption that, at each iteration, the computed point $x^*_W$ exactly satisfies the optimality conditions of the generated subproblem (2). This can be a strong requirement, for instance whenever a stationary point of (2) cannot be analytically determined, but an iterative method must be necessarily applied. Then, the aim of the paper is the definition of a general decomposition algorithm model where, from one hand, the above requirement is removed, from the other, a degree of freedom in the choice of the variables to be updated at any
iteration is introduced.

In particular, the general scheme that will be defined is characterized by the following features.

- At each iteration, a finite subset of sparse feasible directions having only two nonzero elements is defined. This set has cardinality $O(n)$, contains the generators of the set of feasible directions and has a structure well-suited for large scale problems.

- In the set mentioned above, a suitable direction is selected and an inexact line search along it is performed by means of an Armijo-type method; in this way a new candidate point is obtained without the need of computing an exact solution of a subproblem.

- The candidate point provides a reference value for the objective function and a new arbitrary iterate can be accepted if the corresponding function value does not exceed the reference value previously determined.

- The global convergence of the algorithm is proved under the only assumption that the objective function is continuously differentiable.

The paper is organized as follows. In Section 2, we introduce some basic notation and preliminary technical results. In Section 3 we state theoretical properties of a special class of feasible directions having only two nonzero components and characterizing our decomposition approach. In Section 4 we recall the well-known Armijo-type algorithm and its properties. In Section 5 we describe the decomposition algorithm model, whose theoretical properties are analyzed in Section 6. The numerical results obtained on large-scale SVM training problems in comparison with LIBSVM of [4] are reported in Section 7. Finally, a technical result concerning polyhedral sets and used in our convergence analysis is proved in the Appendix.

2 Notation and preliminary results

In this section we introduce some basic notation and definitions, and we report some results proved in [21] that will be used in the sequel.

Given a vector $x \in \mathbb{R}^n$, and an index set $W \subseteq \{1, \ldots, n\}$, we have already introduced the notation $x_W \in \mathbb{R}^{|W|}$ to indicate the subvector of $x$ made up of the component $x_i$ with $i \in W$. 
Given a set \( Y = \{y^1, \ldots, y^m\} \subset \mathbb{R}^n \), we indicate by \( \text{cone}(Y) \) the convex cone of \( Y \) defined as follows

\[
\text{cone}(Y) = \{ y \in \mathbb{R}^n : y = \sum_{h=1}^{m} \mu_h y^h, \ \mu_h \geq 0, \ h = 1, \ldots, m \}.
\]

We indicate by \( \mathcal{F} \) the feasible set of Problem (1), namely

\[
\mathcal{F} = \{ x \in \mathbb{R}^n : a^T x = b, \ l \leq x \leq u \}.
\]

For every feasible point \( x \), we denote the sets of indices of active (lower and upper) bounds as follows:

\[
L(x) = \{ i : x_i = l_i \}, \quad U(x) = \{ i : x_i = u_i \}.
\]

The set of the feasible directions at a point \( x \in \mathcal{F} \) is the cone

\[
D(x) = \{ d \in \mathbb{R}^n : a^T d = 0, \ d_i \geq 0, \ \forall i \in L(x), \text{ and } d_i \leq 0, \ \forall i \in U(x) \}.
\]

We say that a feasible point \( x^* \) is a stationary point of Problem (1) if

\[
\nabla f(x^*)^T d \geq 0 \quad \text{for all } d \in D(x^*).
\]

Since the constraints of Problem (1) are linear we have that a feasible point \( x^* \) is a stationary point of Problem (1) if and only if the Karush-Kuhn-Tucker (KKT) conditions are satisfied, i.e. a scalar \( \lambda^* \) exists such that

\[
(\nabla f(x^*))_i + \lambda^* a_i \begin{cases} 
\geq 0 & \text{if } i \in L(x^*) \\
\leq 0 & \text{if } i \in U(x^*) \\
= 0 & \text{if } i \not\in L(x^*) \cup U(x^*) \end{cases}
\]

The KKT conditions can be written in a different form. To this aim the sets \( L \) and \( U \) can be split in \( L^- \), \( L^+ \), and \( U^- \), \( U^+ \) respectively, where

\[
L^-(x) = \{ i \in L(x) : a_i < 0 \}, \quad L^+(x) = \{ i \in L(x) : a_i > 0 \}
\]

\[
U^-(x) = \{ i \in U(x) : a_i < 0 \}, \quad U^+(x) = \{ i \in U(x) : a_i > 0 \}.
\]

We report the KKT conditions in the following proposition.

**Proposition 1 (Optimality conditions)** A point \( x^* \in \mathcal{F} \) is a stationary point of Problem (1) if and only if there exists a scalar \( \lambda^* \) satisfying

\[
\lambda^* \geq -\frac{\nabla f(x^*)_i}{a_i} \quad \forall i \in L^+(x^*) \cup U^-(x^*)
\]

\[
\lambda^* \leq -\frac{\nabla f(x^*)_i}{a_i} \quad \forall i \in L^-(x^*) \cup U^+(x^*)
\]

\[
\lambda^* = -\frac{\nabla f(x^*)_i}{a_i} \quad \forall i \not\in L(x^*) \cup U(x^*)
\]
In correspondence to a feasible point $x$, we introduce the index sets
\[
R(x) = L^+(x) \cup U^-(x) \cup \{i : l_i < x_i < u_i\},
\]
\[
S(x) = L^-(x) \cup U^+(x) \cup \{i : l_i < x_i < u_i\}.
\]

**Proposition 2** A feasible point $x^*$ is a stationary point of Problem (1) if and only if for any pair of indices $(i, j)$, with $i \in R(x^*)$ and $j \in S(x^*)$, we have
\[
\frac{\nabla f(x^*)_i}{a_i} \geq \frac{\nabla f(x^*)_j}{a_j}.
\]

**Proposition 3** Let $\{x^k\}$ be a sequence of feasible points convergent to a point $\bar{x}$. Then for sufficiently large values of $k$ we have
\[
R(\bar{x}) \subseteq R(x^k) \text{ and } S(\bar{x}) \subseteq S(x^k).
\]

### 3 Sets of sparse feasible directions

In this section we consider a special class of feasible directions having only two nonzero components and we show their important properties that will be employed in the decomposition approach presented later.

Given $i, j \in \{1, \ldots, n\}$, with $i \neq j$, we indicate by $d^{i,j}$ a vector belonging to $R^n$ such that
\[
d^{i,j}_h = \begin{cases} 
\frac{1}{a_i}, & \text{if } h = i \\
\frac{1}{a_j}, & \text{if } h = j \\
0, & \text{otherwise.}
\end{cases}
\]

Given $x \in \mathcal{F}$ and the corresponding index sets $R(x)$ and $S(x)$, we indicate by $D_{RS}(x)$ the set of directions $d^{i,j}$ with $i \in R(x)$ and $j \in S(x)$, namely
\[
D_{RS}(x) = \bigcup_{\substack{i \in R(x) \\ j \in S(x) \\ i \neq j}} d^{i,j}.
\]

The following proposition was proved in [17].

**Proposition 4** Let $\hat{x}$ be a feasible point. For each pair $i \in R(\hat{x})$ and $j \in S(\hat{x})$, the direction $d^{i,j} \in R^n$ is a feasible direction at $\hat{x}$, i.e. $d \in D(\hat{x})$.

The result stated below follows immediately from Proposition 2.
Proposition 5 A feasible point \( x^* \) is a stationary point of Problem (1) if and only if
\[
\nabla f(x^*)^T d^{i,j} \geq 0 \quad \forall d^{i,j} \in D_{RS}(x^*). \tag{8}
\]

The next proposition shows that for any feasible point \( x \) the set \( D_{RS}(x) \) contains feasible directions and the generators of \( D(x) \).

Proposition 6 Given \( \bar{x} \in \mathcal{F} \), we have
\[
D_{RS}(\bar{x}) \subseteq D(\bar{x}), \tag{9}
\]
and
\[
cone\{D_{RS}(\bar{x})\} = D(\bar{x}). \tag{10}
\]

Proof. Condition (9) is a consequence of Proposition 4.

In order to prove (10), using (9), we must show that \( d \in D(\bar{x}) \) implies \( d \in cone\{D_{RS}(\bar{x})\} \). Assume by contradiction that the thesis is false, so that, there exists a vector \( \bar{d} \in D(\bar{x}) \) such that \( \bar{d} \notin cone\{D_{RS}(\bar{x})\} \). Hence we have that the linear system
\[
\bar{d} = |D_{RS}(\bar{x})| \sum_{h=1}^{D_{RS}(\bar{x})} \mu_h d^h, \quad \mu_h \geq 0, \; h = 1, \ldots, |D_{RS}(\bar{x})|
\]
has no solution, where \( d^h \in D_{RS}(\bar{x}) \). Using Farkas’s lemma we have that there exists a vector \( c \in \mathbb{R}^n \) such that
\[
c^T d^h \geq 0 \quad \forall \; d^h \in D_{RS}(\bar{x}) \tag{11}
\]
and
\[
c^T \bar{d} < 0. \tag{12}
\]

Now consider the linear function \( F(x) = c^T x \) and the convex problem
\[
\min \; F(x) = c^T x \quad \quad a^T x = b \quad \quad l \leq x \leq u. \tag{13}
\]

Condition (11) can be written as
\[
\nabla F(\bar{x})^T d^h \geq 0 \quad \forall d^h \in D_{RS}(\bar{x}). \tag{14}
\]

Using (14) and Proposition 5, we get that \( \bar{x} \) is an optimal solution of Problem (13). On the other hand, \( \bar{d} \) is a feasible direction at \( \bar{x} \), and by
(12) we have $\nabla F(\bar{x})^T \bar{d} < 0$. Then, $\bar{d}$ is a feasible descent direction at $\bar{x}$, and this contradicts the fact that $\bar{x}$ is an optimal solution of Problem (13). □

We observe that the set $D_{RS}(\bar{x})$ has cardinality, depending on $R(\bar{x})$ and $S(\bar{x})$, which is in the worst case $O(n^2)$.

Now, under a suitable condition on the feasible point $\bar{x}$, we show that it can be easily defined a set of feasible directions containing, as $D_{RS}(\bar{x})$, the generators of $D(\bar{x})$, but having cardinality $O(n)$. In particular, let $\bar{x}$ be a feasible point with at least one “free” components, i.e. such that

$$l_h < \bar{x}_h < u_h,$$

for some index $h \in \{1, \ldots, n\}$. In correspondence to a point $\bar{x}$ satisfying (15) we define the following set of directions

$$D^h(\bar{x}) = \{d^{i,h} \in R^n : i \in R(\bar{x})\} \cup \{d^{h,j} \in R^n : j \in S(\bar{x})\}. \quad (16)$$

**Remark 1** We note that $n - 1 \leq |D^h(\bar{x})| \leq 2(n - 1)$. In fact in correspondence to each “free” index $t \neq h$, the set $D^h(\bar{x})$ must include the direction $d^{t,h}$ as $t \in R(\bar{x})$, and the direction $d^{h,t}$ as $t \in S(\bar{x})$. Hence if $R(\bar{x}) \cap S(\bar{x}) = \{1, \ldots, n\}$, then $|D^h(\bar{x})| = 2(n - 1)$. Conversely, for each index $t \notin R(\bar{x}) \cap S(\bar{x})$ there exists only one element of $D^h(\bar{x})$. Therefore, if $R(\bar{x}) \cap S(\bar{x}) = \{h\}$, then $|D^h(\bar{x})| = n - 1$. □

As an example concerning the definition of $D^h(\bar{x})$, consider Problem (1) with $n = 4$, $l = 0$, $u = 3$, and $a = (1, -1, 1, -1)^T$. Let $\bar{x} = (0, 1, 3, 2)^T$ be a feasible point, so that $R(\bar{x}) = \{1, 2, 4\}, R(\bar{x}) \cap S(\bar{x}) = \{2, 4\}, S(\bar{x}) = \{2, 3, 4\}$.

Setting $h = 2$ we obtain

$$D^2(\bar{x}) = \left\{ \begin{pmatrix} \frac{1}{a_1} \\ \frac{1}{a_2} \\ \frac{1}{a_3} \\ 0 \\ 0 \\ \frac{1}{a_4} \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{1}{a_2} \\ \frac{1}{a_3} \\ 0 \\ \frac{1}{a_4} \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{1}{a_2} \\ 0 \\ \frac{1}{a_4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{1}{a_3} \\ 0 \\ \frac{1}{a_4} \end{pmatrix} \right\}$$

In the next proposition we prove that the set $D^h(\bar{x})$ defined in (16) is indeed a subset of feasible directions and further that it contains the generators of the set of feasible directions $D(\bar{x})$. 

---

8
**Proposition 7** Let $\bar{x} \in \mathcal{F}$, and let $h$ be an index such that $l_h < \bar{x}_h < u_h$. Then we have

\[ D^h(\bar{x}) \subseteq D(\bar{x}), \tag{17} \]

and

\[ \text{cone}\{D^h(\bar{x})\} = D(\bar{x}), \tag{18} \]

where $D^h(\bar{x})$ is defined by (16).

**Proof.** Note that, by assumption, the index $h$ is such that $h \in R(\bar{x}) \cap S(\bar{x})$. Consider any $i \in R(\bar{x})$ (or any $j \in S(\bar{x})$) such that $d^{i,h} \in D^h(\bar{x})$ (or $d^{h,j} \in D^h(\bar{x})$). Then, $d^{i,h}$ ($d^{h,j}$) is such that $i \in R(\bar{x})$ and $h \in S(\bar{x})$ ($h \in R(\bar{x})$ and $j \in S(\bar{x})$), so that, condition (17) follows from Proposition 4.

In order to prove (18), using Proposition 6, it is sufficient to show that

\[ \text{cone}\{D^h(\bar{x})\} = \text{cone}\{D_{RS}(\bar{x})\}. \tag{19} \]

Given any $d^{i,j} \in D_{RS}(\bar{x})$ we can write

\[ d^{i,j} = d^{ih} + d^{hj}. \tag{20} \]

On the other hand, by definition of $D^h(\bar{x})$, we have necessarily $d^{ih}, d^{hj} \in D^h(\bar{x})$, and hence (18) follows from (20) and (19).

As a consequence of Propositions 5 and 7, we get directly the following result.

**Proposition 8** Let $\bar{x} \in \mathcal{F}$ be a point such that (15) holds, and let $h$ be an index such that $l_h < \bar{x}_h < u_h$. The feasible point $\bar{x}$ is a stationary point of Problem (1) if and only if

\[ \nabla f(\bar{x})^T d \geq 0 \quad \forall d \in D^h(\bar{x}), \tag{21} \]

where $D^h(\bar{x})$ is defined by (16).

### 4 Armijo-type line search algorithm

In this section we describe the well-known Armijo-type line search along a feasible direction. The procedure will be used in the decomposition method presented in the next section. We state also some theoretical results useful in our convergence analysis.

Let $d^k$ be a feasible direction at $x^k \in \mathcal{F}$. We denote by $\beta^k_{\bar{x}}$ the maximum feasible steplength along $d^k$, namely $\beta^k_{\bar{x}}$ satisfies

\[ l \leq x^k + \beta d^k \leq u \quad \text{for all } \beta \in [0, \beta^k_{\bar{x}}], \]
and (since $-\infty \leq l < u \leq \infty$) we have that either $\beta^k_F = +\infty$ or at least an index $i \in \{1, \ldots, n\}$ exists such that
\[ x^k_i + \beta^k_F d^k_i = l_i \quad \text{or} \quad x^k_i + \beta^k_F d^k_i = u_i. \]

Let $\beta_u$ be a positive scalar and set
\[ \beta^k = \min\{\beta^k_F, \beta_u\}. \tag{22} \]

**Assumption 1** Assume that $\{d^k\}$ is a sequence of feasible search directions such that
(a) for all $k$ we have $\|d^k\| \leq M$ for a given number $M > 0$; 
(b) for all $k$ we have $\nabla f(x^k)^T d^k < 0$.

An Armijo-type line search algorithm is described below.

---

**Armijo-type line search ALS($x^k, d^k, \beta^k$)**

**Data:** Given $\alpha > 0$, $\delta \in (0, 1)$, $\gamma \in (0, 1/2)$ and the initial stepsize $\alpha^k = \min\{\beta^k, \alpha\}$.

**Step 1.** Set $\lambda = \alpha^k$, $j = 0$.

**Step 2.** If
\[ f(x^k + \lambda d^k) \leq f(x^k) + \gamma \lambda \nabla f(x^k)^T d^k \tag{23} \]
then set $\lambda^k = \lambda$ and stop.

**Step 3.** Set $\lambda = \delta \lambda$, $j = j + 1$ and go to Step 2.

---

The properties of Algorithm ALS are reported in the next proposition.

**Proposition 9** Let $\{x^k\}$ be a sequence of points belonging to the feasible set $\mathcal{F}$, and let $\{d^k\}$ be a sequence of search directions satisfying Assumption 1. Then:

(i) Algorithm ALS determines, in a finite number of iterations, a scalar $\lambda^k$ such that condition (23) holds, i.e.,
\[ f(x^k + \lambda^k d^k) \leq f(x^k) + \gamma \lambda^k \nabla f(x^k)^T d^k; \tag{24} \]
(ii) if \( \{x^k\} \) converges to \( \bar{x} \) and
\[
\lim_{k \to \infty} \left( f(x^k) - f(x^k + \lambda^k d^k) \right) = 0,
\]
then we have
\[
\lim_{k \to \infty} \beta^k \nabla f(x^k)^T d^k = 0,
\]
where \( \beta^k \) is given by (22).

**Proof.**

Point (i). Assume by contradiction that the algorithm does not terminate, so that we can write
\[
f(x^k + \alpha^k (\delta)^j d^k) > f(x^k) + \gamma \alpha^k (\delta)^j \nabla f(x^k)^T d^k \quad \text{for all } j.
\]
By applying the Mean Value theorem we have
\[
\nabla f(x^k + \theta(j) \alpha^k (\delta)^j d^k)^T d^k > \gamma \nabla f(x^k)^T d^k \quad \text{for all } j,
\]
with \( \theta(j) \in (0, 1) \). Taking limits in (27) for \( j \to \infty \) we obtain
\[
(1 - \gamma) \nabla f(x^k)^T d^k \geq 0,
\]
which implies, together with the fact that \( \gamma \in (0, 1/2) \), that \( \nabla f(x^k)^T d^k \geq 0 \), and this contradicts assumption (b) on \( d^k \).

Point (ii). The proof is by contradiction. Recalling that the sequences \( \{\beta^k\}, \{d^k\} \) are bounded, that \( x^k \to \bar{x} \) and that \( \nabla f \) is continuous, we can assume that there exists an infinite subset \( K \subseteq \{0, 1, \ldots\} \) such that
\[
\lim_{k \to \infty, k \in K} \beta^k = \bar{\beta} > 0,
\]
\[
\lim_{k \to \infty, k \in K} d^k = \bar{d},
\]
\[
\lim_{k \to \infty, k \in K} \nabla f(x^k)^T d^k = \nabla f(\bar{x})^T \bar{d} < 0.
\]
From (23) and (25) we get
\[
\lim_{k \to \infty, k \in K} \lambda^k \nabla f(x^k)^T d^k = 0,
\]
which implies, together with (30), that
\[
\lim_{k \to \infty, k \in K} \lambda^k = 0.
\]
Then, from (29) and (31) we have
\[
\lim_{k \to \infty, k \in K} \lambda^k \|d^k\| = 0. \tag{32}
\]
Recalling that the initial tentative stepsize \( \alpha^k \) is such that \( \alpha^k = \min\{\beta^k, \alpha\} \) and using (28) we have that \( \alpha^k \) does not tend to zero for \( k \to \infty \) and \( k \in K \). Then, (31) implies that \( \lambda^k < \alpha^k \) for \( k \in K \) and \( k \) sufficiently large, so that, recalling the instructions of Algorithm ALS, we can write
\[
f \left( x^k + \frac{\lambda^k}{\delta} d^k \right) \geq f(x^k) + \gamma \left( \frac{\lambda^k}{\delta} \right) \nabla f(x^k)^T d^k.
\]
By applying the Mean Value theorem we obtain
\[
\nabla f \left( x^k + \theta^k \frac{\lambda^k}{\delta} d^k \right)^T d^k > \gamma \nabla f(x^k)^T d^k, \tag{33}
\]
with \( \theta^k \in (0, 1) \). Taking limits in (33) for \( k \to \infty, k \in K \), and using (32), it follows
\[
(1 - \gamma) \nabla f(\bar{x})^T d \geq 0,
\]
which contradicts (30) since \( \gamma \in (0, \frac{1}{2}) \).

5 A decomposition algorithm model

In this section we describe a decomposition algorithm based on the employment of the feasible directions having only two nonzero components. The results stated in Section 4 show that, given any feasible point \( x^k \), the set \( D(x^k) \) of feasible directions at \( x^k \) can be generated by finite sets of directions with only two nonzero components. In particular, Proposition 6 shows that the set \( D_{RS}(x^k) \) contains the generators of \( D(x^k) \), while Proposition 7 states that, under the assumption that \( x^k \) satisfies (15), the same result holds with the set \( D_i(x^k) \), where \( i \) is an index such that
\[
l_i < x_i^k < u_i.
\]
Thus both the sets \( D_{RS}(x^k) \) and \( D_i(x^k) \) contain “sufficient” information. In the worst case \( D_{RS}(x^k) \) has \( n(n-1)/2 \) elements, while \( D_i(x^k) \) is made of \( 2(n-1) \) vectors. Therefore, in order to develop a feasible direction-based approach for large dimensional problems, it appears to be more appropriate to exploit a set of directions of the form \( D_i(x^k) \). More in particular, we illustrate the strategy underlying our approach. In the case that the current feasible point \( x^k \) is such that (15) is verified, we
identify a “suitable” index $i(k)$ (we highlight the dependance from the iteration $k$) corresponding to a “free” component $x_{i(k)}^k$, and we consider the following problem

$$\min_{\beta \in R, d \in R^n} \beta \nabla f(x^k)^T d$$

$$0 \leq \beta \leq \beta_u$$

$$x^k + \beta d \in \mathcal{F}$$

$$d \in D^{i(k)}(x^k),$$

where $\beta_u$ is a given positive scalar and

$$D^{i(k)}(x^k) = \{d^{h,i(k)} \in R^n : h \in R(x^k)\} \cup \{d^{i(k),h} \in R^n : h \in S(x^k)\}.$$  

(35)

It is easy to see that problem (34) admits a solution.

The rationale underlying the definition of problem (34) is that of identifying, among the generators of $D(x^k)$ having only two nonzero components, a feasible direction which locally may produce a sufficient decrease of the function values. To this aim, beside to the directional derivatives, we must take into account the amplitudes of the feasible steeplengths. Denoting by $(\beta^k, d^k)$ a solution of (34), we have that $\beta^k$ is the minimum between the maximum feasible steplength along $d^k$ and the given positive scalar $\beta_u$.

Then, our strategy at each iteration $k$ is based on the following sequential steps:

- given the feasible current point $x^k$ satisfying (15), among the free indexes $R(x^k) \cap S(x^k)$, we identify an index $i(k)$ corresponding to a “sufficiently free” component, that is such that

$$\min\{x_{i(k)}^k - l_{i(k)}, u_{i(k)} - x_{i(k)}^k\} \geq \bar{\eta} > 0,$$

where $\bar{\eta}$ is a prefixed sufficiently small positive scalar; in the case that

$$\min\{x_{i}^k - l_{i}, u_{i} - x_{i}^k\} < \bar{\eta}, \quad \forall i \in \{1, \ldots, n\}$$

we select an index $i(k) \in \{1, \ldots, n\}$ such that

$$\min\{x_{i(k)}^k - l_{i(k)}, u_{i(k)} - x_{i(k)}^k\} \geq \min\{x_{i}^k - l_{i}, u_{i} - x_{i}^k\} \quad \forall i \in \{1, \ldots, n\},$$

that is $i(k)$ corresponds to the “most free” component;

- given the subset of feasible directions $D^{i(k)}(x^k)$ defined as in (35), we determine a solution $(\beta^k, d^k)$ of (34), which should produce a direction of “sufficient” decrease of the objective function;
we compute a feasible point $\tilde{x}^{k+1}$ by means of the Armijo-type line search along the direction $d^k$ to obtain a reference value $f_{ref}^k = f(\tilde{x}^{k+1})$;

- the current point $x^k$ is updated by any feasible point $x^{k+1}$ such that
  
  \[ f(x^{k+1}) \leq f_{ref}^k. \]

The algorithm is formally described below, where we denote by $\eta^k$ the following positive scalar

\[ \eta^k = \max_{h \in R(x^k) \cap S(x^k)} \left\{ \min\{x_h^k - l_h, u_h - x_h^k\} \right\}. \tag{36} \]
FEasible DIRections (FEDIR) Algorithm

Data. A feasible point $x^0$ such that $l_h < x^0_h < u_h$ for some $h \in \{1, \ldots, n\}$; $\bar{\eta} > 0$, $\beta_u > 0$.

Inizialization. Set $k = 0$

While (stopping criterion not satisfied)

1. select $i(k) \in R(x^k) \cap S(x^k)$ such that
   $$\min \{x^k_{i(k)} - l_{i(k)}, u_{i(k)} - x^k_{i(k)}\} \geq \min \{\eta^k, \bar{\eta}\}$$

2. Let $D^{i(k)}(x^k)$ be the set of feasible directions of $x^k$ defined by (35), then select $d^k, \beta^k$ such that
   $$\beta^k \nabla f(x^k)^T d^k = \min_{\beta \in R, d \in R^n} \beta \nabla f(x^k)^T d$$
   $$0 \leq \beta \leq \beta_u$$
   $$d \in D^{i(k)}(x^k)$$
   $$x^k + \beta d \in \mathcal{F}.$$ (37)

3. Compute the stepsize $\lambda^k$ using Algorithm ALS($x^k, d^k, \beta^k$).

4. Set $\bar{x}^{k+1} = x^k + \lambda^k d^k$ and $f_{ref}^k = f(\bar{x}^{k+1})$;

5. Find $x^{k+1} \in \mathcal{F}$ such that $f(x^{k+1}) \leq f_{ref}^k$.

6. set $k = k + 1$;

end while

Return $x^* = x^k$

The role of Step 1 and Step 2 is that of defining the two components of $x^k$, identified by the indices $i(k)$ and $j(k)$, that will be changed, by Step 3 and Step 4, to determine the reference point $\bar{x}^{k+1}$ and the corresponding reference value $f_{ref}^k = f(\bar{x}^{k+1})$. In other words, the pair $(i(k), j(k))$ identifies the working set $W$ which leads, in a traditional decomposition scheme, to subproblem (2).

Concerning Step 3, we have that Algorithm ALS performs an inexact one dimensional constrained minimization along the search direction $d^k$,
thus computing a suitable stepsize $\lambda^k$. Note that Proposition 8 guarantees that $d^k$ is such that $\nabla f(x^k)^T d^k < 0$ whenever $x^k$ is not a stationary point. Therefore, as $\|d^k\| \leq \sqrt{2} \|\frac{1}{\alpha}\|_\infty$, it follows that Assumption 1 is satisfied and hence the properties of Algorithm ALS (stated in Proposition 9) hold.

We remark that in the quadratic case (either convex or non convex) Step 3 can be simplified. Indeed, the optimal feasible stepsize $\lambda^*$ along $d^k$ satisfies the acceptance Armijo-type condition and it can be analytically determined without applying Algorithm ALS.

Finally, we point out that at Step 5 we have the possibility of defining the updated point $x^{k+1}$ in any way provided that $f(x^{k+1}) \leq f_{\text{ref}}$. Therefore, we can employ, in principle, a working set of arbitrary dimension and arbitrary elements to compute the new point $x^{k+1}$. This is an important aspect both from a theoretical and a practical point of view.

6 Convergence Analysis

Before analyzing the convergence properties of the decomposition algorithm model, we need to introduce the following assumption, which must guarantee that the every limit point of the generated sequence has at least a “free” component.

Assumption 2 Let $\{x^k\}$ be a sequence of feasible points. There exists a scalar $\epsilon > 0$ such that for all $k$ we have

$$\eta^k \geq \epsilon$$

where $\eta^k$ is defined in (36).

Note that Assumption 2 is automatically satisfied whenever at least one variable is unbounded, namely $l_i = -\infty$, $u_i = +\infty$ for some $i \in \{1, \ldots, n\}$.

The convergence properties of the algorithm are stated in the next proposition.

Proposition 10 Let $\{x^k\}$ be the sequence generated by the algorithm and assume that Assumption 2 holds. Then, every limit point of $\{x^k\}$ is a stationary point of Problem (1).

Proof. Let $\bar{x}$ be any limit point of $\{x^k\}$, i.e. there exists an infinite subset $K \subseteq \{0, 1, \ldots, \}$ such that $x^k \rightarrow \bar{x}$ for $k \in K, k \rightarrow \infty$.

First, we note that $i(k) \in \{1, \ldots, n\}$ so that we can extract a further subset of $K$ (relabelled again $K$) and an index $i \in \{1, \ldots, n\} \cap R(x^k) \cap S(x^k)$ such that

$$D^{i(k)}(x^k) = D^i(x^k)$$

16
for all $k \in K$.

Recalling the selection rule of index $i$ at step 1 and Assumption 2, we can write
\[
\min\{x^k_i - l_i, u_i - x^k_i\} \geq \min\{\eta^k, \bar{\eta}\} \geq \min\{\varepsilon, \bar{\eta}\}.
\]
Taking limits for $k \to \infty$ and $k \in K$ we obtain
\[
\min\{\bar{x}_i - l_i, u_i - \bar{x}_i\} \geq \min\{\varepsilon, \bar{\eta}\} > 0,
\]
from which we get that $i \in R(\bar{x}) \cap S(\bar{x})$. Then, (18) implies that
\[
cone(D^i(\bar{x})) = D(\bar{x}).
\]

Assume now, by contradiction, that $\bar{x}$ is not a stationary point of Problem (1). Then, by Proposition 8 there exists a direction $\hat{d} \in D^i(\bar{x})$ such that
\[
\nabla f(\bar{x})^T \hat{d} < 0. \tag{38}
\]
By definition (16) we have
\[
D^i(x^k) = \{d^r,i \in R^i : r \in R(x^k)\} \cup \{d^s,i \in R^n : s \in S(x^k)\}.
\]
Furthermore, from Proposition 3 we have $R(\bar{x}) \subseteq R(x^k)$ and $S(\bar{x}) \subseteq S(x^k)$ for $k \in K$ and $k$ sufficiently large. Therefore, for $k \in K$ and $k$ sufficiently large it follows that $D^i(\bar{x}) \subseteq D^i(x^k)$, and hence that
\[
\hat{d} \in D^i(x^k). \tag{39}
\]
Let $\hat{\beta}^k$ be the the maximum feasible stepsize along $\hat{d}$ starting from $x^k$.
From Proposition 11, reported in the Appendix, we have
\[
\hat{\beta} \leq \hat{\beta}^k \leq +\infty, \tag{40}
\]
for some $\hat{\beta} > 0$.
Note that from (38) and (39) it follows that the direction $d^k$ and the steplength $\beta^k$ selected at step 2 are such that
\[
\beta^k \nabla f(x^k)^T d^k \leq \min\{\hat{\beta}^k, \beta_u\} \nabla f(x^k)^T \hat{d} < 0. \tag{41}
\]
Furthermore, we have $\|d^k\| \leq \sqrt{2}\|\frac{1}{a}\|_\infty$ for all $k$, and hence Assumption 1 on $\{d^k\}$ holds. Step 5 of the algorithm implies that
\[
f(x^{k+1}) \leq f(\bar{x}^{k+1}) = f(x^k + \lambda^k d^k) \leq f(x^k) + \gamma \lambda^k \nabla f(x^k)^T d^k, \quad \gamma \in (0, 1/2) \tag{42}
\]
where \( \lambda^k \) is the stepsize computed by Algorithm ALS. The sequence \( \{ f(x^k) \} \) is monotone decreasing and convergent to a finite value since, by the continuity of \( f \), we have

\[
\lim_{k \to \infty, k \in K} f(x^k) = f(\bar{x}).
\]

Therefore, (42) implies

\[
\lim_{k \to \infty} \left( f(x^k) - f(x^k + \lambda^k d^k) \right) = 0,
\]

namely that condition (25) of Proposition 9 holds. Thus, from assertion (ii) of Proposition 9 we obtain

\[
\lim_{k \to \infty, k \in K} \beta^k \nabla f(x^k)^T d^k = 0,
\]

where \( \beta^k = \min\{ \beta^k_f, \beta^k_u \} \), and \( \beta^k_f \) is the maximum feasible stepsize along \( d^k \). Then, taking limits in (41) for \( k \to \infty, k \in K \), and using (43) and (40) we obtain

\[
\nabla f(\bar{x})^T \hat{d} = 0,
\]

but this contradicts (38). \( \square \)

7 Numerical experiments on SVM training problems

In this section we present a specific realization of FEDIR algorithm for the particular class of problems arising in large-scale SVM training. We report the numerical results obtained on standard test problems.

Given a training set of input-target pairs \((u^i, a^i)\), \( i = 1, \ldots, n \), with \( u^i \in \mathbb{R}^m \), and \( a^i \in \{-1, 1\} \), the SVM classification technique [23] requires the solution of the following quadratic programming problem

\[
\min f(x) = \frac{1}{2} x^T Q x - e^T x \\
\text{s.t.} \quad a^T x = 0 \\
0 \leq x \leq C,
\]

where \( x \in \mathbb{R}^n \), \( Q \) is a \( n \times n \) symmetric positive semidefinite matrix, \( e \in \mathbb{R}^n \) is the vector of all ones, \( a \in \{-1, 1\}^n \) and \( C \) is a positive scalar. The generic element \( q_{ij} \) of the matrix \( Q \) is given by \( a^i a^j K(u^i, u^j) \), where \( K(u, z) = \phi(u)^T \phi(z) \) is the kernel function related to the nonlinear
function $\phi$ that maps the data from the input space into the feature space.

One of the most popular and efficient algorithms for SVM training is the LIBSVM algorithm [4]. This is a decomposition method where, at each iteration $k$, the working set $W$ of subproblem (2) is determined by the two nonzero components of the solution of problem

$$\min_{d} \left\{ \nabla f(x^k)^T d : d \in D(x^k), \quad -e \leq d \leq e, \quad \{i : d_i \neq 0\} = 2 \right\}. \quad (45)$$

LIBSVM (version 2.71) algorithm can in turn be viewed as a special case of the SVM$^{tight}$ algorithm [12], which is based on a specific procedure for choosing the $q$ elements of the working set, being $q$ any even number. Note that problem (45) can be equivalently written as:

$$\min_{d \in \mathbb{R}^n} \nabla f(x^k)^T d$$

$$d \in D_{RS}(x^k), \quad (46)$$

and that its solution is a descent direction for $f$ at $x^k$ (see e.g. [16]). The selection of the two indices by solving problem (46) requires $O(n)$ operations. The two indices are those corresponding to the “maximal violation” of the KKT conditions (see Proposition 2), and hence the selection rule is usually referred as the Maximal Violation (MV) rule [11].

We observe that in the case of working set $W$ of cardinality two the optimal solution of subproblem (2) can be analytically computed (see, e.g., [6]).

We remark that we are assuming that the Hessian matrix cannot be stored, and hence in computational terms the most expensive step of a general decomposition method is the evaluation of the columns of the Hessian matrix, corresponding to the indices in the working set $W$, not stored in memory. Actually, these columns are needed for updating the gradient, being in the quadratic case

$$\nabla f(x^{k+1}) = \nabla f(x^k) + \sum_{i=1}^{n} Q_i (x^{k+1} - x^k)_i,$$

where $Q_i$ is the $i$-th column of $Q$. To reduce the computational time a commonly adopted strategy is based on the use of a caching technique that allocates some memory space (the cache) to store the recently used columns of $Q$ thus avoiding in some cases the recomputation of these columns.

Algorithms based on the Maximal Violation (MV) rule have theoretical convergence properties and are efficient from a computational point
of view, but they are not designed to fully exploit the information on the matrix stored in the cache. We define here a specific realization of FEDIR algorithm for SVM training, called FEDIR-SVM, where a caching strategy can be advantageously exploited.

We recall that at Step 4 of FEDIR algorithm a reference value \( f_{ref}^k = f(\tilde{x}^{k+1}) \) is calculated. In particular, the point \( \tilde{x}^{k+1} \) is the solution of subproblem (2), where \( W \) is defined by the two nonzero components of the direction \( d^k \) determined by Step 2.

Different realizations can be derived from FEDIR scheme by specifying the rule to select, at Step 5, the new iterate \( x^{k+1} \) for which the only condition to be satisfied is \( f(x^{k+1}) \leq f_{ref}^k \).

In our proposed approach, beside to the reference point \( \tilde{x}^{k+1} \), two tentative points, \( x^{k+1}_a \) and \( x^{k+1}_b \), are analytically generated: the point \( x^{k+1}_a \) is determined according to the MV strategy, the point \( x^{k+1}_b \) is defined to try to exploit the information in the cache memory thus reducing the computational time.

Concerning the vector \( x^{k+1}_a \), it is the solution of subproblem (2), where \( W \) is defined by the two nonzero components, identified by the indices \( i_a \) and \( j_a \), of the solution of (46). Thus, we have

\[
x^{k+1}_a = x^k + \lambda^k_a d^k_a,
\]

where \( d^k_a \) is the solution of (46) and

\[
\lambda^k_a = \arg \min_{\lambda \in [0, \beta^k_a]} f(x^k + \lambda d^k_a),
\]

being \( \beta^k_a \leq C \) the maximum feasible steplength along \( d^k_a \). Hence it follows that \( f(x^{k+1}_a) < f(x^k) \).

As regards the definition of the point \( x^{k+1}_b \), let \( I_C \subset \{1, \ldots, n\} \) be the set of indices (excluding \( i_a \) and \( j_a \)) of the columns of \( Q \) stored in the cache memory, and let

\[
R_c(x^k) = R(x^k) \cap I_C^k, \quad S_c(x^k) = S(x^k) \cap I_C^k.
\]

We apply the most violating strategy to the indices corresponding to the columns stored in the cache memory so that we define, in the case that both the sets \( R_c \) and \( S_c \) are not empty, the direction \( d^k_b \), with two nonzero components identified by the indices \( i_b \) and \( j_b \), as the solution of

\[
\min_{d \in \mathbb{R}^n} \nabla f(x^k)^T d \quad \text{subject to} \quad d \in D_{R_c S_c}(x^k),
\]

(47)
where
\[ D_{R_c S_c}(x) = \bigcup_{\substack{i \in R_c(x) \\ j \in S_c(x) \\ i \neq j}} d^{i,j}. \]

Note that it is not guaranteed that \( d_b^k \) is a descent direction for \( f \) at \( x^k \). In the case that \( \nabla f(x^k)^T d_b^k < 0 \), we find the value
\[
\lambda_b^k = \arg \min_{\lambda \in [0, \beta_b^k]} f(x^k + \lambda d_b^k),
\]
being \( \beta_b^k \leq C \) the maximum feasible steplength along \( d_b^k \), and we set
\[
x_b^{k+1} = x^k + \lambda_a^k d_a^k + \lambda_b^k d_b^k. \quad (48)
\]

The vector \( x_b^{k+1} \) can be seen as the result of a simple procedure for inexact solving the four-dimensional subproblem (2) with working set \( W = \{i_a, j_a, i_b, j_b\} \).

Summarizing, at Step 5 of FEDIR scheme we define \( x_b^{k+1} \) as the point whose function value corresponds to the minimum either between \( f_{ref}^k \) and \( f(x_a^{k+1}) \), or between \( f_{ref}^k \), \( f(x_a^{k+1}) \) and \( f(x_b^{k+1}) \) whenever the point \( x_b^{k+1} \) has been generated. Note that at each iteration either two or four variables are updated. In any case, the updated point is analytically determined, furthermore, thanks to the information stored in the cache memory, at most two columns of the Hessian matrix must be possibly recomputed for the gradient updating.

In order to make some fair computational comparison between LIBSVM and FEDIR-SVM, this latter has been implemented by suitably modifying the available code (written in C++) of LIBSVM.

In our computational experiments the classification test problems described below have been used. All the problems have been taken from LIBSVM Database [15], except for the random problem, and \( n, m \) denote the number of training pairs and the dimension of the input space respectively.

- \( a4a \): \( n = 4781, m = 123; \)
- \( mushrooms \): \( n = 8124, m = 112; \)
- \( w5a \): \( n = 9888, m = 300; \)
- \( rcv1_train.binary \): \( n = 20242, m = 47236; \)
- \( ijcnn1 \): \( n = 49990, m = 22; \)
- random: $n = 50000$, $m = 20$; it has been constructed starting from two linearly separable sets of points by changing the classification of a certain number (equal to 1% of the overall points) of randomly chosen observations [17, 8].

The experiments have been performed using in (44) the gaussian kernel $K(u, z) = e^{-\gamma \|u - z\|^2}$, with $\gamma = 1$, i.e., the generic element $q_{i,j}$ of the Hessian matrix $Q$ is $a_i a_j e^{-\|u^i - u^j\|^2}$, and setting the upper bound $C$ of the box constraints equal to 5.

The initial point $x^0$ in FEDIR-SVM has been determined starting from the feasible point $x = 0$ and applying the MV strategy, which corresponds to perform the first iterate of LIBSVM. Furthermore, we have set in FEDIR-SVM $\bar{\eta} = 10^{-6}$, $\beta_u = C = 5$, and we have used the same stopping criterion adopted by LIBSVM [4].

All the experiments have been carried out on a 2.60 GHz Pentium 4 with 512 megabytes of RAM and cache size of 40 MB.

The results are shown in Table I, where we report the number of iterations ($n_i$), the number of column evaluations of the matrix $Q (K_{ev})$, the attained function value ($f^*$), and the required cpu time (cpu) expressed in seconds.

<table>
<thead>
<tr>
<th>Problem (dimension $n$)</th>
<th>Algorithm</th>
<th>$n_i$</th>
<th>$K_{ev}$</th>
<th>$f^*$</th>
<th>cpu</th>
</tr>
</thead>
<tbody>
<tr>
<td>a4a $(n = 4781)$</td>
<td>FEDIR-SVM</td>
<td>5538</td>
<td>10331</td>
<td>-2358.74</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>LIBSVM</td>
<td>6599</td>
<td>10714</td>
<td>-2358.74</td>
<td>34</td>
</tr>
<tr>
<td>mushrooms $(n = 8124)$</td>
<td>FEDIR-SVM</td>
<td>12193</td>
<td>24386</td>
<td>-1072.91</td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>LIBSVM</td>
<td>16083</td>
<td>32152</td>
<td>-1072.91</td>
<td>203</td>
</tr>
<tr>
<td>w5a $(n = 9888)$</td>
<td>FEDIR-SVM</td>
<td>9141</td>
<td>17272</td>
<td>-789.39</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>LIBSVM</td>
<td>10217</td>
<td>19400</td>
<td>-789.39</td>
<td>113</td>
</tr>
<tr>
<td>rcv1_train.binary $(n = 20242)$</td>
<td>FEDIR-SVM</td>
<td>10460</td>
<td>20436</td>
<td>-2498.03</td>
<td>817</td>
</tr>
<tr>
<td></td>
<td>LIBSVM</td>
<td>13570</td>
<td>24358</td>
<td>-2498.03</td>
<td>971</td>
</tr>
<tr>
<td>ijcnn1 $(n = 49990)$</td>
<td>FEDIR-SVM</td>
<td>12666</td>
<td>13128</td>
<td>-12986.54</td>
<td>354</td>
</tr>
<tr>
<td></td>
<td>LIBSVM</td>
<td>26741</td>
<td>20196</td>
<td>-12986.54</td>
<td>544</td>
</tr>
<tr>
<td>random $(n = 50000)$</td>
<td>FEDIR-SVM</td>
<td>26435</td>
<td>52804</td>
<td>-7673.07</td>
<td>1430</td>
</tr>
<tr>
<td></td>
<td>LIBSVM</td>
<td>40559</td>
<td>80822</td>
<td>-7673.07</td>
<td>2230</td>
</tr>
</tbody>
</table>

From the results reported in Table I we can observe that the number of iterations and the number of matrix column evaluations required by FEDIR-SVM are always lower than those required by LIBSVM. Concerning the comparison in terms of number of iterations, we have that in
many cases FEDIR-SVM accepts as new iterate the point \( x_b^{k+1} \) (see (48)) thus updating four variables, while LIBSVM always updates two variables, and this motivates the better performances of FEDIR-SVM. As regards the better behaviour of FEDIR-SVM in terms of number of matrix column evaluations, from one hand it depends on the lower number of iterations, from the other one it depends on the fact that FEDIR-SVM is designed to exploit as much as possible the caching strategy. As consequence, FEDIR-SVM outperforms LIBSVM in terms of cpu time in all the runs except for problem \( a4a \). We can note that the difference of the performances of the two algorithms becomes significant as the dimension of the problem increases. Thus, the computational experiments performed on a class of SVM large-scale quadratic problems show the effectiveness of the implemented decomposition scheme derived from the general algorithm model proposed in this work.

A Technical result

We state here a theoretical result valid for polyhedral sets. This result is used in the convergence proof of the general algorithm. More specifically, we consider a polyhedron

\[
\Gamma = \{ x \in \mathbb{R}^n : Ax \leq b \}.
\]

where \( A \) is a \( m \times n \) matrix and \( b \in \mathbb{R}^m \).

Given \( x \in \Gamma \), we denote by \( I(x) \) the set of active constraints at \( x \), that is

\[
I(x) = \{ i \in \{1,\ldots,m\} : a_i^T x = b_i \}.
\]

The set of feasible directions \( D(x) \) at \( x \in \Gamma \) is

\[
D(x) = \{ d \in \mathbb{R}^n : a_i^T d \leq 0 \ \forall i \in I(x) \}.
\]

Given \( d \in D(x) \), we denote by \( \mathcal{H} = \{ i \in \{1,\ldots,m\} : a_i^T d > 0 \} \). The maximum feasible stepsize along \( d \) is

\[
\beta = \begin{cases} 
+\infty & \text{if } \mathcal{H} = \emptyset \\
\min_{i \in \mathcal{H}} \left\{ \frac{b_i - a_i^T x}{a_i^T d} \right\} & \text{otherwise}
\end{cases} \quad (49)
\]

**Proposition 11** Let \( A \in \mathbb{R}^{m \times n} \) and \( b \in \mathbb{R}^m \). Let \( \{x^k\} \) be a sequence of points, such that

\[
Ax^k \leq b
\]

(50)
for all \( k \). Assume further that

\[
\lim_{k \to \infty} x^k = \bar{x}.
\] (51)

Then, given any direction \( \bar{d} \in D(\bar{x}) \), there exists a scalar \( \hat{\beta} > 0 \) such that, for sufficiently large values of \( k \), we have

\[
A(x^k + \beta \bar{d}) \leq b, \quad \forall \beta \in [0, \hat{\beta}].
\] (52)

**Proof.** Assume first that \( \mathcal{H} = \emptyset \). Then, recalling (49), it follows that (52) holds for any \( \hat{\beta} > 0 \).

Now suppose that \( \mathcal{H} \neq \emptyset \) and let \( \bar{\beta} \) be the maximum feasible stepsize along \( \bar{d} \) starting from \( \bar{x} \), i.e.,

\[
\bar{\beta} = \min_{i \in \mathcal{H}} \left\{ \frac{b_i - a_i^T \bar{x}}{a_i^T \bar{d}} \right\} > 0
\]

(the inequality holds since \( \bar{d} \in D(\bar{x}) \)). Then we have

\[
a_i^T (\bar{x} + \beta \bar{d}) < b_i, \quad \forall i \in \mathcal{H}, \forall \beta \in \left[ 0, \frac{\bar{\beta}}{2} \right],
\]

and hence there exists a scalar \( \epsilon > 0 \) such that

\[
a_i^T (\bar{x} + \beta \bar{d}) \leq b_i - \epsilon, \quad \forall i \in \mathcal{H}, \forall \beta \in \left[ 0, \frac{\bar{\beta}}{2} \right].
\] (53)

Moreover we have

\[
a_j^T (\bar{x} + \beta \bar{d}) \leq b_j, \quad \forall j \notin \mathcal{H}, \forall \beta \in [0, +\infty].
\] (54)

Recalling (51), for \( k \) sufficiently large, we can write

\[
| a_i^T x^k - a_i^T \bar{x} | \leq \frac{\epsilon}{2}, \quad \forall i \in \{1, \ldots, m\}.
\] (55)

Finally, using (53) and (55), for \( k \) sufficiently large and \( \forall i \in \mathcal{H} \), we have

\[
a_i^T (x^k + \beta \bar{d}) = a_i^T (x^k + \bar{x} - \bar{x} + \beta \bar{d}) \leq b_i - \epsilon + \frac{\epsilon}{2} = b_i - \frac{\epsilon}{2}, \quad \forall \beta \in \left[ 0, \frac{\bar{\beta}}{2} \right]
\] (56)

so that, using (54), (52) is proved with \( \beta = \frac{\bar{\beta}}{2} \). \( \square \)
References


