Lower Bounds on Rate of Convergence of Cutting Plane Methods

Xinhua Zhang  
Dept. of Computing Science  
University of Alberta  
xinhua2@ualberta.ca

Ankan Saha  
Dept. of Computer Science  
University of Chicago  
ankans@cs.uchicago.edu

S.V.N. Vishwanathan  
Dept. of Statistics and  
Dept. of Computer Science  
Purdue University  
vishy@stat.purdue.edu

Abstract

In a recent paper Joachims [1] presented SVM-Perf, a cutting plane method (CPM) for training linear Support Vector Machines (SVMs) which converges to an $\epsilon$ accurate solution in $O(1/\epsilon^2)$ iterations. By tightening the analysis, Teo et al. [2] showed that $O(1/\epsilon)$ iterations suffice. Given the impressive convergence speed of CPM on a number of practical problems, it was conjectured that these rates could be further improved. In this paper we disprove this conjecture. We present counter examples which are not only applicable for training linear SVMs with hinge loss, but also hold for support vector methods which optimize a multivariate performance score. However, surprisingly, these problems are not inherently hard. By exploiting the structure of the objective function we can devise an algorithm that converges in $O(1/\sqrt{\epsilon})$ iterations.

1 Introduction

There has been an explosion of interest in machine learning over the past decade, much of which has been fueled by the phenomenal success of binary Support Vector Machines (SVMs). Driven by numerous applications, recently, there has been increasing interest in support vector learning with linear models. At the heart of SVMs is the following regularized risk minimization problem:

$$\min_w J(w) := \lambda \|w\|^2 + R_{\text{emp}}(w)$$

with $R_{\text{emp}}(w) := \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y_i \langle w, x_i \rangle)$. (1)

Here we assume access to a training set of $n$ labeled examples $\{(x_i, y_i)\}_{i=1}^{n}$ where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, and use the square Euclidean norm $\|w\|^2 = \sum_i w_i^2$ as the regularizer. The parameter $\lambda$ controls the trade-off between the empirical risk and the regularizer.

There has been significant research devoted to developing specialized optimizers which minimize $J(w)$ efficiently. In an award winning paper, Joachims [1] presented a cutting plane method (CPM), SVM-Perf, which was shown to converge to an $\epsilon$ accurate solution of (1) in $O(1/\epsilon^2)$ iterations, with each iteration requiring $O(nd)$ effort. This was improved by Teo et al. [2] who showed that their Bundle Method for Regularized Risk Minimization (BMRM) (which encompasses SVM-Perf as a special case) converges to an $\epsilon$ accurate solution in $O(nd/\epsilon)$ time.

While online learning methods are becoming increasingly popular for solving (1), a key advantage of CPM such as SVM-Perf and BMRM is their ability to directly optimize nonlinear multivariate performance measures such as $F_1$-score, ordinal regression loss, and ROC Area which are widely used in some application areas. In this case $R_{\text{emp}}$ does not decompose into a sum of losses over individual data points like in (1), and hence one has to employ batch algorithms. Letting $\Delta(y, \bar{y})$ denote the multivariate discrepancy between the correct labels $y := (y_1, \ldots, y_n)$ and a candidate labeling $\bar{y}$ (to be concretized later), the $R_{\text{emp}}$ for the multivariate measure is formulated by [3] as

---

1In this paper we use the term cutting plane methods to denote specialized solvers employed in machine learning. While clearly related, they must not be confused with cutting plane methods used in optimization.
At every iteration, \( BMRM \) replaces \( R_{\text{emp}}(w) \) by a piecewise linear lower bound \( R_k^{\text{cp}} \) and optimizes [2]

\[
\min_w J_k(w) := \frac{\lambda}{2} \|w\|^2 + R_k^{\text{cp}}(w), \quad \text{where } R_k^{\text{cp}}(w) := \max_{1 \leq i \leq k} \langle w, a_i \rangle + b_i,
\]

to obtain the next iterate \( w_k \). Here \( a_i \in \partial R_{\text{emp}}(w_{i-1}) \) denotes an arbitrary subgradient of \( R_{\text{emp}} \) at \( w_{i-1} \) and \( b_i = R_{\text{emp}}(w_{i-1}) - \langle w_{i-1}, a_i \rangle \). The piecewise linear lower bound is successively tightened until the gap

\[
\epsilon_k := \min_{0 \leq i \leq k} J(w_i) - J_k(w_k)
\]

falls below a predefined tolerance \( \epsilon \).

Since \( J_k \) in (3) is a convex objective function, one can compute its dual. Instead of minimizing \( J_k \) with respect to \( w \) one can equivalently maximize the dual [2] over the \( k \) dimensional simplex:

\[
D_k(\alpha) = -\frac{1}{2\lambda} \|A_k \alpha\|^2 + \langle b_k, \alpha \rangle, \quad \text{where } \alpha \in \Delta_k,
\]

In another award winning paper by Joachims [3], the regularized risk minimization problems corresponding to these measures are optimized by using a \( CPM \).

Given the widespread use of \( CPM \) in machine learning, it is important to understand their convergence guarantees in terms of the upper and lower bounds on the number of iterations needed to converge to an \( \epsilon \) accurate solution. The tightest, \( O(1/\epsilon) \), upper bounds on the convergence speed of \( CPM \) is due to Teo et al. [2], who analyzed a restricted version of \( BMRM \) which only optimizes over one dual variable per iteration. However, on practical problems the observed rate of convergence is significantly faster than predicted by theory. Therefore, it had been conjectured that the upper bounds might be further tightened via a more refined analysis. In this paper we construct counter examples for both decomposable \( R_{\text{emp}} \) like in equation (1) and non-decomposable \( R_{\text{emp}} \) like in equation (2), on which \( CPM \) requires \( \Omega(1/\epsilon) \) iterations to converge, thus disproving this conjecture\(^2\). We will work with \( BMRM \) as our prototypical \( CPM \). As Teo et al. [2] point out, \( BMRM \) includes many other \( CPM \) such as \( SVM-\text{Perf} \) as special cases.

Our results lead to the following natural question: Do the lower bounds hold because regularized risk minimization problems are fundamentally hard, or is it an inherent limitation of \( CPM \)? In other words, to solve problems such as (1), does there exist a solver which requires less than \( O(nd/\epsilon) \) effort (better in \( n, d \) and \( \epsilon \))? We provide partial answers. To understand our contribution one needs to understand the two standard assumptions that are made when proving convergence rates:

- **A1**: The data points \( x_i \) lie inside a \( L_2 \) (Euclidean) ball of radius \( R \), that is, \( \|x_i\| \leq R \).
- **A2**: The subgradient of \( R_{\text{emp}} \) is bounded, i.e., at any point \( w \), there exists a subgradient \( g \) of \( R_{\text{emp}} \) such that \( \|g\| \leq G < \infty \).

Clearly assumption **A1** is more restrictive than **A2**. By adapting a result due to [6] we show that one can devise an \( O(nd/\sqrt{\epsilon}) \) algorithm for the case when assumption **A1** holds. Finding a fast optimizer under assumption **A2** remains an open problem.

**Notation**: Lower bold case letters (e.g., \( w, \mu \)) denote vectors, \( w_i \) denotes the \( i \)-th component of \( w \), \( 0 \) refers to the vector with all zero components, \( e_i \) is the \( i \)-th coordinate vector (all 0's except 1 at the \( i \)-th coordinate) and \( \Delta_k \) refers to the \( k \) dimensional simplex. Unless specified otherwise, \( \langle \cdot, \cdot \rangle \) denotes the Euclidean dot product \( \langle x, w \rangle = \sum_i x_i w_i \), and \( \|\cdot\| \) refers to the Euclidean norm \( \|w\| := (\langle w, w \rangle)^{1/2} \). We denote \( \mathbb{R} := \mathbb{R} \cup \{\infty\} \), and \( [t] := \{1, \ldots, t\} \).

Our paper is structured as follows. We briefly review **BMRM** in Section 2. Two types of lower bounds are subsequently defined in Section 3, and Section 4 contains descriptions of various counter examples that we construct. In Section 5 we describe an algorithm which provably converges to an \( \epsilon \) accurate solution of (1) in \( O(1/\sqrt{\epsilon}) \) iterations under assumption **A1**. The paper concludes with a discussion and outlook in Section 6. Technical proofs and a ready reckoner of the convex analysis concepts used in the paper can be found in [7, Appendix A].

### 2 BMRM

At every iteration, **BMRM** replaces \( R_{\text{emp}} \) by a piecewise linear lower bound \( R_k^{\text{cp}} \) and optimizes [2]

\[
\min_w J_k(w) := \frac{\lambda}{2} \|w\|^2 + R_k^{\text{cp}}(w), \quad \text{where } R_k^{\text{cp}}(w) := \max_{1 \leq i \leq k} \langle w, a_i \rangle + b_i,
\]

to obtain the next iterate \( w_k \). Here \( a_i \in \partial R_{\text{emp}}(w_{i-1}) \) denotes an arbitrary subgradient of \( R_{\text{emp}} \) at \( w_{i-1} \) and \( b_i = R_{\text{emp}}(w_{i-1}) - \langle w_{i-1}, a_i \rangle \). The piecewise linear lower bound is successively tightened until the gap

\[
\epsilon_k := \min_{0 \leq i \leq k} J(w_i) - J_k(w_k)
\]

falls below a predefined tolerance \( \epsilon \).

Since \( J_k \) in (3) is a convex objective function, one can compute its dual. Instead of minimizing \( J_k \) with respect to \( w \) one can equivalently maximize the dual [2] over the \( k \) dimensional simplex:

\[
D_k(\alpha) = -\frac{1}{2\lambda} \|A_k \alpha\|^2 + \langle b_k, \alpha \rangle, \quad \text{where } \alpha \in \Delta_k,
\]

\(^2\)Because of the specialized nature of these solvers, lower bounds for general convex optimizers such as those studied by Nesterov [4] and Nemirovski and Yudin [5] do not apply.
Theorem 1 ([2])

Iteration, [2] proposed using a one-dimensional line search to calculate an approximate maximizer the size of the QP also increases. In order to avoid the growing cost of the dual optimization at each scaled version \( cJ \)

A technical point. Instead of minimizing the objective function its relationship with the upper bounds. At this juncture it is also important to clarify an important

et al. [2] could only show that both variants of \( \text{BMRM} \) \( \text{qp-bmrm} \)

Even though \( \text{qp-bmrm} \) solves a more expensive optimization problem \( D_k(\alpha) \) per iteration, Teo et al. [2] could only show that both variants of \( \text{BMRM} \) converge at \( O(1/\epsilon) \) rates:

\[ \log_2 \frac{\lambda J(0)}{G^2} + \frac{8G^2}{\lambda \epsilon} - 1. \]

Algorithm 1: \( \text{qp-bmrm}: \) solving the inner loop of \( \text{BMRM} \) exactly via full QP.

\begin{algorithm}
\begin{algorithmic}[1]
\Require Previous subgradients \( \{a_i\}_{i=1}^k \) and intercepts \( \{b_i\}_{i=1}^k \).
1: Set \( A_k := (a_1, \ldots, a_k), b_k := (b_1, \ldots, b_k)^\top. \)
2: \( \alpha_k \leftarrow \arg \max_{\alpha \in \Delta_k} \left\{ -\frac{1}{2\lambda} \|A_k\alpha\|^2 + \langle \alpha, b_k \rangle \right\}. \)
3: \( \text{return } w_k = -\lambda^{-1} A_k \alpha_k. \)
\end{algorithmic}
\end{algorithm}

Algorithm 2: \( \text{ls-bmrm}: \) solving the inner loop of \( \text{BMRM} \) approximately via line search.

\begin{algorithm}
\begin{algorithmic}[1]
\Require Previous subgradients \( \{a_i\}_{i=1}^k \) and intercepts \( \{b_i\}_{i=1}^k \).
1: Set \( A_k := (a_1, \ldots, a_k), b_k := (b_1, \ldots, b_k)^\top. \)
2: Set \( \alpha(\eta) := (\eta \alpha_{k-1}^\top, 1 - \eta)^\top. \)
3: \( \eta_k \leftarrow \arg \max_{\eta \in [0,1]} \left\{ \frac{1}{2\lambda} \|A_k\alpha(\eta)\|^2 + \langle \alpha(\eta), b_k \rangle \right\}. \)
4: \( \alpha_k \leftarrow (\eta_k \alpha_{k-1}^\top, 1 - \eta_k)^\top. \)
5: \( \text{return } w_k = -\lambda^{-1} A_k \alpha_k. \)
\end{algorithmic}
\end{algorithm}

and set \( \alpha_k = \arg \max_{\alpha \in \Delta_k} D_k(\alpha) \). Note that \( A_k \) and \( b_k \) in (5) are defined in Algorithm 1. Since maximizing \( D_k(\alpha) \) is a quadratic programming (QP) problem, we call this algorithm \( \text{qp-bmrm} \).

Pseudo-code can be found in Algorithm 1.

Note that at iteration \( k \) the dual \( D_k(\alpha) \) is a QP with \( k \) variables. As the number of iterations increases the size of the QP also increases. In order to avoid the growing cost of the dual optimization at each iteration, [2] proposed using a one-dimensional line search to calculate an approximate maximizer \( \alpha_k \) on the line segment \( \{(\eta \alpha_{k-1}^\top, 1 - \eta)^\top : \eta \in [0,1]\} \), and we call this variant \( \text{ls-bmrm} \).

Pseudo-code can be found in Algorithm 2. We refer the reader to [2] for details.

Even though \( \text{qp-bmrm} \) solves a more expensive optimization problem \( D_k(\alpha) \) per iteration, Teo et al. [2] could only show that both variants of \( \text{BMRM} \) converge at \( O(1/\epsilon) \) rates:

\[ \log_2 \frac{\lambda J(0)}{G^2} + \frac{8G^2}{\lambda \epsilon} - 1. \]

Algorithm 2

\[ \text{Require: Previous subgradients } \{a_i\}_{i=1}^k \text{ and intercepts } \{b_i\}_{i=1}^k. \]
1: Set \( A_k := (a_1, \ldots, a_k), b_k := (b_1, \ldots, b_k)^\top. \)
2: Set \( \alpha(\eta) := (\eta \alpha_{k-1}^\top, 1 - \eta)^\top. \)
3: \( \eta_k \leftarrow \arg \max_{\eta \in [0,1]} \left\{ \frac{1}{2\lambda} \|A_k\alpha(\eta)\|^2 + \langle \alpha(\eta), b_k \rangle \right\}. \)
4: \( \alpha_k \leftarrow (\eta_k \alpha_{k-1}^\top, 1 - \eta_k)^\top. \)
5: \( \text{return } w_k = -\lambda^{-1} A_k \alpha_k. \)

3 Upper and Lower Bounds

Since most rates of convergence discussed in the machine learning community are upper bounds, it is important to rigorously define the meaning of a lower bound with respect to \( \epsilon \), and to study its relationship with the upper bounds. At this juncture it is also important to clarify an important technical point. Instead of minimizing the objective function \( J(w) \) defined in (1), if we minimize a scaled version \( cJ(w) \) this scales the approximation gap (4) by \( c \).

Assumptions such as \( \text{A1} \) and \( \text{A2} \) fix this degree of freedom by bounding the scale of the objective function.

Given a function \( f \in \mathcal{F} \) and an optimization algorithm \( A \), suppose \( \{w_k\} \) are the iterates produced by the algorithm \( A \) when minimizing \( f \). Define \( T(\epsilon; f, A) \) as the first step index \( k \) when \( w_k \) becomes an \( \epsilon \) accurate solution\(^3\):

\[ T(\epsilon; f, A) = \min \{ k : f(w_k) - \min_w f(w) \leq \epsilon \}. \]  \hspace{1cm} (6)

Upper and lower bounds are both properties for a pair of \( \mathcal{F} \) and \( A \). A function \( g(\epsilon) \) is called an upper bound of \( (\mathcal{F}, A) \) if for all functions \( f \in \mathcal{F} \) and all \( \epsilon > 0 \), it takes at most order \( g(\epsilon) \) steps for \( A \) to reduce the gap to less than \( \epsilon \), i.e.,

\[ \text{UB} \]

\[ \forall \epsilon > 0, \forall f \in \mathcal{F}, T(\epsilon; f, A) \leq g(\epsilon). \]  \hspace{1cm} (7)

\(^3\) The initial point also matters, as in the best case we can just start from the optimal solution. Thus the quantity of interest is actually \( T(\epsilon; f, A) := \max_{w_0} \min \{ k : f(w_k) - \min_w f(w) \leq \epsilon \}, \) starting point being \( w_0 \). However, without loss of generality we assume some pre-specified way of initialization.
The proof relies on two lemmata. The first shows that the iterates generated by
\begin{align*}
\text{Theorem 2} & \quad \lim_{k \to \infty} \frac{1}{k} (J(w_k) - J(w^*)) = \frac{1}{4}, \text{ i.e. } J(w_k) \text{ converges to } J(w^*) \text{ at } 1/k \text{ rate.}
\end{align*}

The proof relies on two lemmata. The first shows that the iterates generated by \textbf{ls-bmrm} on \( J(w) \) satisfy the following recursive relations.
Lemma 4 For $k \geq 1$, the following recursive relations hold true

\[
\begin{align*}
\alpha_{2k+1} &= 2 + \frac{8\alpha_{2k-1,1}}{w_{2k-1}} \left( w_{2k-1} - 4\alpha_{2k-1,1} \right) > 2, \quad \text{and} \quad w_{2k} = 2 - \frac{8\alpha_{2k-1,1}}{w_{2k-1}} \in (1, 2). \\
\alpha_{2k+1,1} &= \frac{w_{2k-1}^2 + 16\alpha_{2k-1,1}}{(w_{2k-1} + 4\alpha_{2k-1,1})^2} \alpha_{2k-1,1}, \text{where } \alpha_{2k+1,1} \text{ is the first coordinate of } \alpha_{2k+1}.
\end{align*}
\]  

The proof is lengthy and is available at [7, Appendix B]. These recursive relations allow us to derive the convergence rate of $\alpha_{2k-1,1}$ and $w_k$ (see proof in [7, Appendix C]):

Lemma 4 \(\lim_{k \to \infty} k\alpha_{2k-1,1} = \frac{1}{4}\). Combining with (11), we get \(\lim_{k \to \infty} k|2 - w_k| = 2\).

Now that $w_k$ approaches 2 at the rate of $O(1/k)$, it is finally straightforward to translate it into the rate at which $J(w_k)$ approaches $J(w^*)$. See the proof of Theorem 2 in [7, Appendix D].

4.2 Weak Lower Bounds for Solving Linear SVMs using \texttt{qp-bmrm}

Theorem 1 gives an upper bound on the convergence rate of \texttt{qp-bmrm}, assuming that $R_{\text{emp}}$ satisfies the assumption A2. In this section we further demonstrate that this $O(1/\epsilon)$ rate is also a WLB (hence tight) even when the $R_{\text{emp}}$ is specialized to SVM objectives satisfying A2.

Given $\epsilon > 0$, define $n = \lceil 1/\epsilon \rceil$ and construct a dataset \(\{(x_i, y_i)\}_{i=1}^n\) as $y_i = (-1)^i$ and $x_i = (-1)^i (n\epsilon_i + 1 + \sqrt{n}\epsilon_1) \in \mathbb{R}^{n+1}$. Then the corresponding objective function (1) is

\[
J(w) = \frac{1}{2} \frac{||w||^2}{2} + R_{\text{emp}}(w), \quad \text{where } R_{\text{emp}}(w) = \frac{1}{n} \sum_{i=1}^n |1 - y_i(w, x_i)|_+ = \frac{1}{n} \sum_{i=1}^n [1 - \sqrt{n}w_1 - nw_{i+1}]_+.
\]  

It is easy to see that the minimizer $w^* = \frac{1}{2} \left( \frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}} \right)^\top$ and $J(w^*) = \frac{1}{4\sqrt{n}}$. In fact, simply check that $y_i \langle w^*, x_i \rangle = 1$, so $\partial J(w^*) = \left\{ w^* - \left( \frac{1}{\sqrt{n}} \sum_{i=1}^n \alpha_i, \alpha_1, \ldots, \alpha_n \right)^\top : \alpha_i \in [0, 1] \right\}$, and setting all $\alpha_i = \frac{1}{2n}$ yields the subgradient 0. Our key result is the following theorem.

Theorem 5 Let $w_0 = \left( \frac{1}{\sqrt{n}}, 0, 0, \ldots \right)^\top$. Suppose running \texttt{qp-bmrm} on the objective function (13) produces iterates $w_1, \ldots, w_k, \ldots$. Then it takes \texttt{qp-bmrm} at least $\left\lceil \frac{2}{3\epsilon} \right\rceil$ steps to find an $\epsilon$ accurate solution. Formally,

\[
\min_{i \in [k]} J(w_i) - J(w^*) = \frac{1}{2} + \frac{1}{4\sqrt{n}} \quad \text{for all } k \in [n], \text{hence } \min_{i \in [k]} J(w_i) - J(w^*) > \epsilon \quad \text{for all } k < \frac{2}{3\epsilon}.
\]

Indeed, after taking $n$ steps, $w_n$ will cut a subgradient $a_{n+1} = 0$ and $b_{n+1} = 0$, and then the minimizer of $J_{n+1}(w)$ gives exactly $w^*$.

Proof Since $R_{\text{emp}}(w_0) = 0$ and $\partial R_{\text{emp}}(w_0) = \left\{ \frac{1}{n} \sum_{i=1}^n \alpha_i y_i x_i : \alpha_i \in [0, 1] \right\}$, we can choose

\[
a_1 = -\frac{1}{n} y_1 x_1 = \left( -\frac{1}{\sqrt{n}}, -1, 0, \ldots \right)^\top, \quad b_1 = R_{\text{emp}}(w_0) - \langle a_1, w_0 \rangle = 0 + \frac{1}{n} = \frac{1}{n}, \quad \text{and}
\]

\[
w_1 = \arg\min_w \left\{ \frac{1}{2} \frac{||w||^2}{2} - \frac{1}{\sqrt{n}} w_1 - w_2 + \frac{1}{n} \right\} = \left( \frac{1}{\sqrt{n}}, 1, 0, \ldots \right)^\top.
\]

In general, we claim that the $k$-th iterate $w_k$ produced by \texttt{qp-bmrm} is given by

\[
w_k = \left( \frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}, 0, \ldots \right)^\top.
\]

We prove this claim by induction on $k$. Assume the claim holds true for steps $1, \ldots, k$, then it is easy to check that $R_{\text{emp}}(w_k) = 0$ and $\partial R_{\text{emp}}(w_k) = \left\{ \frac{1}{n} \sum_{i=k+1}^n \alpha_i y_i x_i : \alpha_i \in [0, 1] \right\}$. Thus we can again choose
\[ a_{k+1} = -\frac{1}{n} y_{k+1} x_{k+1}, \quad \text{and} \quad b_{k+1} = R_{\text{emp}}(w_k) - \langle a_{k+1}, w_k \rangle = \frac{1}{n}, \] so
\[
w_{k+1} = \text{argmin}_w \left\{ \frac{1}{2} \|w\|^2 + \max_{1 \leq i \leq k+1} \{ \langle a_i, w \rangle + b_i \} \right\} = \left( \frac{1}{\sqrt{n}}, \frac{1}{k+1}, \ldots, \frac{1}{k+1}, 0, \ldots \right) \top^{k+1 \text{ copies}},
\]
which can be verified by checking that \( \partial J_{k+1}(w_{k+1}) = \left\{ w_{k+1} + \sum_{i \in [k+1]} a_i a_i : a \in \Delta_{k+1} \right\} \supseteq 0. \) All that remains is to observe that \( J(w_k) = \frac{1}{2n} + \frac{1}{2n} \) while \( J(w^*) = \frac{1}{2n} \) from which it follows that \( J(w_k) - J(w^*) = \frac{1}{2n} + \frac{1}{4n} \) as claimed.

As an aside, the subgradient of the \( R_{\text{emp}} \) in (13) does have Euclidean norm \( \sqrt{2n} \) at \( w = 0. \) However, in the above run of \( \text{qp-bmrm}, \) \( \partial R_{\text{emp}}(w_0), \ldots, \partial R_{\text{emp}}(w_n) \) always contains a subgradient with norm 1. So if we restrict the feasible region to \( \{ n^{-1/2} \} \times [0, \infty]^n, \) then \( J(w) \) does satisfy the assumption \( A2 \) and the optimal solution does not change. This is essentially a local satisfaction of \( A2. \) In fact, having a bounded subgradient of \( R_{\text{emp}} \) at all \( w_k \) is sufficient for \( \text{qp-bmrm} \) to converge at the rate in Theorem 1.

However when we assume \( A1 \) which is more restrictive than \( A2, \) it remains an open question to determine whether the \( O(1/\epsilon) \) rates are optimal for \( \text{qp-bmrm} \) on SVM objectives. Also left open is the SLB for \( \text{qp-bmrm} \) on SVMs.

4.3 Weak Lower Bounds for Optimizing \( F_1 \)-score using \( \text{qp-bmrm} \)

\( F_1 \)-score is defined by using the contingency table: \( F_1(y, y) := \frac{2a}{2a + b + c}. \)

Given \( \epsilon > 0, \) define \( n = \lceil 1/\epsilon \rceil + 1 \) and construct a dataset \( \{ (x_i, y_i) \}_{i=1}^n \) as follows: \( x_i = -\frac{n}{2\sqrt{n}} e_i - \frac{n}{2} e_{n+1} \in \mathbb{R}^{n+1} \) with \( y_i = -1 \) for all \( i \in [n-1], \)
and \( x_n = \frac{\sqrt{n}}{2} e_1 + \frac{n}{2} e_{n+1} \in \mathbb{R}^{n+1} \) with \( y_n = +1. \) So there is only one positive training example. Then the corresponding objective function is
\[
J(w) = \frac{1}{2} \|w\|^2 + \max_y \left[ 1 - F_1(y, \bar{y}) + \frac{1}{n} \sum_{i=1}^n y_i \langle w, x_i \rangle (y_i \bar{y}_i - 1) \right].
\]

**Theorem 6** Let \( w_0 = \frac{1}{\sqrt{3}} e_1. \) Then \( \text{qp-bmrm} \) takes at least \( \left\lfloor \frac{1}{2\epsilon} \right\rfloor \) steps to find an \( \epsilon \) accurate solution.

\[
J(w_k) - \min_w J(w) \geq \frac{1}{2} \left( \frac{1}{k} - \frac{1}{n-1} \right) \forall k \in [n-1], \text{ hence } \min_{i \in [k]} J(w_i) - \min_w J(w) > \epsilon \forall k < \frac{1}{2\epsilon}.
\]

**Proof** A rigorous proof can be found in [7, Appendix E], we provide a sketch here. The crux is to show
\[
w_k = \left( \frac{1}{\sqrt{3}}, \frac{1}{k}, \ldots, \frac{1}{k}, 0, \ldots \right) \top \forall k \in [n-1],
\]
(15)
We prove (15) by induction. Assume it holds for steps 1, \ldots, \( k. \) Then at step \( k + 1 \) we have
\[
\frac{1}{n} y_i \langle w_k, x_i \rangle = \begin{cases} 
\frac{1}{6} + \frac{1}{2k} & \text{if } i \in [k] \\
\frac{1}{6} & \text{if } k + 1 \leq i \leq n - 1 \\
\frac{1}{2} & \text{if } i = n 
\end{cases}
\]
(16)
For convenience, define the term in the \( \max \) in (14) as
\[
\Upsilon_k(y) := 1 - F_1(y, \bar{y}) + \frac{1}{n} \sum_{i=1}^n y_i \langle w_k, x_i \rangle (y_i \bar{y}_i - 1).
\]
Then it is not hard to see that the following assignments of \( \bar{y} \) (among others) maximize \( \Upsilon_k \): a) correct labeling, b) only misclassify the positive training example \( x_n \) (i.e., \( y_n = -1 \)), c) only misclassify one negative training example in \( x_{k+1}, \ldots, x_{n-1} \) into positive. And \( \Upsilon_k \) equals 0 at all these assignments. For a proof, consider two cases. If \( \bar{y} \) misclassifies the positive training example, then \( F_1(y, \bar{y}) = 0 \) and by (16) we have

\[
\Upsilon_k(\bar{y}) = 1 - \frac{2}{2 + t_1 + t_2} + \frac{1}{6} + \frac{1}{2k} \sum_{i=1}^{k} (y_i \bar{y}_i - 1) + \frac{1}{6} \sum_{i=k+1}^{n-1} (y_i \bar{y}_i - 1) \leq 0.
\]

Suppose \( \bar{y} \) correctly labels the positive example, but misclassifies \( t_1 \) examples in \( x_1, \ldots, x_k \) and \( t_2 \) examples in \( x_{k+1}, \ldots, x_{n-1} \) (into positive). Then \( F_1(y, \bar{y}) = \frac{2}{2 + t_1 + t_2} \), and

\[
\Upsilon_k(\bar{y}) = 1 - \frac{t_1 + t_2}{2 + t_1 + t_2} - \left( \frac{1}{3} + \frac{1}{k} \right) t_1 - \frac{1}{3} t_2 \leq \frac{t - t^2}{3(2 + t)} \leq 0 \quad (t := t_1 + t_2).
\]

So we can pick \( \bar{y} \) as \((-1, \ldots, -1, 1, -1, \ldots, -1, +1)^\top \) which only misclassifies \( x_{k+1} \), and get

\[
a_{k+1} = -\frac{2}{n} y_{k+1} x_{k+1} = -\frac{1}{\sqrt{3}} e_1 - e_{k+2}, \quad b_{k+1} = R_{\text{emp}}(w_k) - \langle a_{k+1}, w_k \rangle = 0 + \frac{1}{3} = \frac{1}{3},
\]

\[
w_{k+1} = \arg \min_w \frac{1}{2} \|w\|^2 + \max_{\alpha \in [k+1]} \{\langle a_{k+1}, w \rangle + b_{k+1} \} = \left( \frac{1}{\sqrt{3}}, \frac{1}{k+1}, \ldots, \frac{1}{k+1}, 0, \ldots \right)^\top.
\]

which can be verified by \( \partial J_{k+1}(w_{k+1}) = \{w_{k+1} + \sum_{i=1}^{k+1} \alpha_i a_i : \alpha \in \Delta_{k+1} \} \) \( \ni 0 \) (just set all \( \alpha_i = \frac{1}{k+1} \)). So (15) holds for step \( k + 1 \). End of induction.

All that remains is to observe that \( J(w_k) = \frac{1}{2} (\frac{1}{n-1} + \frac{1}{3}) \) while \( \min_w J(w) \leq J(w_{n-1}) = \frac{1}{2} (\frac{1}{n} + \frac{1}{n-1}) \) from which it follows that \( J(w_k) - \min_w J(w) \geq \frac{1}{2} (\frac{1}{n} - \frac{1}{n-1}) \) as claimed in Theorem 6.

\section{5 An \( O(nd/\sqrt{\epsilon}) \) Algorithm for Training Binary Linear SVMs}

The lower bounds we proved above show that CPM such as BMRM require \( \Omega(1/\epsilon) \) iterations to converge. We now show that this is an inherent limitation of CPM and not an artifact of the problem. To demonstrate this, we will show that one can devise an algorithm for problems (1) and (2) which will converge in \( O(1/\sqrt{\epsilon}) \) iterations. The key difficulty stems from the non-smoothness of the objective function, which renders second and higher order algorithms such as L-BFGS inapplicable. However, thanks to [7, Theorem 7 in Appendix A], the Fenchel dual of (1) is a convex smooth function with a Lipschitz continuous gradient, which are easy to optimize.

To formalize the idea of using the Fenchel dual, we can abstract from the objectives (1) and (2) a composite form of objective functions used in machine learning with linear models:

\[
\min_{w \in Q_1} J(w) = f(w) + g^*(Aw), \quad \text{where } Q_1 \text{ is a closed convex set.} \tag{17}
\]

Here, \( f(w) \) is a strongly convex function corresponding to the regularizer, \( Aw \) stands for the output of a linear model, and \( g^* \) encodes the empirical risk measuring the discrepancy between the correct labels and the output of the linear model. Let the domain of \( g \) be \( Q_2 \). It is well known that [e.g. 8, Theorem 3.3.5] under some mild constraint qualifications, the adjoint form of \( J(w) \):

\[
D(\alpha) = -g(\alpha) - f^*(-A^\top \alpha), \quad \alpha \in Q_2 \tag{18}
\]

satisfies \( J(w) \geq D(\alpha) \) and \( \inf_{w \in Q_1} J(w) = \sup_{\alpha \in Q_2} D(\alpha) \).
Example 1: binary SVMs with bias. Let $A := -YX^T$ where $Y := \text{diag}(y_1, \ldots, y_n)$ and $X := (x_1, \ldots, x_n)$, $f(w) = \frac{1}{2} \|w\|^2$, $g^*(u) = \min_{w \in \mathbb{R}} \frac{1}{2} \sum_{i=1}^n [1 + u_i - y_i b_i]$ which corresponds to $g(\alpha) = -\sum_i \alpha_i$. Then the adjoint form turns out to be the well known SVM dual objective function:

$$D(\alpha) = \sum_i \alpha_i - \frac{1}{2\lambda} \alpha^T YX^T XY \alpha, \quad \alpha \in Q_2 = \left\{ \alpha \in [0, n^{-1}]^n : \sum_i y_i \alpha_i = 0 \right\}. \quad (19)$$

Example 2: multivariate scores. Denote $A$ as a $2^n$-by-$d$ matrix where the $\bar{y}$-th row is $\sum_{i=1}^n x_i (y_i - \bar{y})$ for each $\bar{y} \in \{-1, +1\}^n$, $f(w) = \frac{1}{2} \|w\|^2$, $g^*(u) = \max_y [\Delta(y, \bar{y}) + \frac{1}{2} u_\bar{y}]$ which corresponds to $g(\alpha) = -n \sum_y \Delta(y, \bar{y}) \alpha_y$, we recover the primal objective (2) for multivariate performance measure. Its adjoint form is

$$D(\alpha) = -\frac{1}{2\lambda} \alpha^T A A^T \alpha + n \sum_y \Delta(y, \bar{y}) \alpha_y, \quad \alpha \in Q_2 = \left\{ \alpha \in [0, n^{-1}]^{2^n} : \sum_y \alpha_y = \frac{1}{n} \right\}. \quad (20)$$

In a series of papers [6, 9, 10], Nesterov developed optimal gradient based methods for minimizing the composite objectives with primal (17) and adjoint (18). A sequence of $w_k$ and $\alpha_k$ is produced such that under assumption $A_1$ the duality gap $\bar{J}(w_k) - D(\alpha_k)$ is reduced to less than $\epsilon$ after at most $k = O(1/\sqrt{\epsilon})$ steps. We refer the readers to [9, 11] for details.

### 5.1 Efficient Projections in Training SV Models with Optimal Gradient Methods

However, applying Nesterov’s algorithm is challenging, because it requires an efficient subroutine for computing projections onto the set of constraints $Q_2$. This projection can be either an Euclidean projection or a Bregman projection.

**Example 1: binary SVMs with bias.** In this case we need to compute the Euclidean projection to $Q_2$ defined by (19), which entails solving a Quadratic Programming problem with a diagonal Hessian, many box constraints, and a single equality constraint. We present an $O(n)$ algorithm for this task in [11, Section 5.5.1]. Plugging this into the algorithm described in [9] and noting that all intermediate steps of the algorithm can be computed in $O(nd)$ time directly yield an $O(nd/\sqrt{\epsilon})$ algorithm. More detailed description of the algorithm is available in [11].

**Example 2: multivariate scores.** Since the dimension of $Q_2$ in (20) is exponentially large in $n$, Euclidean projection is intractable and we resort to Bregman projection. Given a differentiable convex function $F$ on $Q_2$, a point $\alpha$, and a direction $g$, we can define the Bregman projection as:

$$V(\alpha, g) := \arg\min_{\alpha \in Q_2} F(\alpha) - \langle \nabla F(\alpha), g \rangle.$$ 

Scaling up $\alpha$ by a factor of $n$, we can choose $F(\alpha)$ as the negative entropy $F(\alpha) = -\sum_i \alpha_i \log \alpha_i$. Then the application of the algorithm in [9] will endow a distribution over all possible labelings:

$$p(y; w) \propto \exp \left( c \Delta(y, \bar{y}) + \sum_i a_i \langle x_i, w \rangle \bar{y}_i \right), \quad \text{where } c \text{ and } a_i \text{ are constant scalars.} \quad (21)$$

The solver will request the expectation $\mathbb{E}_y \left[ \sum_i a_i x_i \bar{y}_i \right]$ which in turn requires that marginal distribution of $p(\bar{y}_i)$, which is not as straightforward as in graphical models because $\Delta(y, \bar{y})$ may not decompose. Fortunately, for multivariate scores defined by contingency tables, it is possible to compute the marginals in $O(n^2)$ time by using dynamic programming, and this cost is similar to the algorithm proposed by [3]. The detail of the dynamic programming is given in [11, Section 5.4].

### 6 Outlook and Conclusion

CPM are widely employed in machine learning especially in the context of structured prediction [12]. While upper bounds on their rates of convergence were known, lower bounds were not studied before. In this paper we set out to fill this gap by exhibiting counter examples in binary classification on which CPM require $\Omega(1/\epsilon)$ iterations. Our examples are substantially different from the one in [13] which requires an increasing number of classes. The $\Omega(1/\epsilon)$ lower bound is a fundamental limitation of these algorithms and not an artifact of the problem. We show this by devising an $O(1/\sqrt{\epsilon})$ algorithm borrowing techniques from [9]. However, this algorithm assumes that the dataset is contained in a ball of bounded radius (assumption $A_1$ Section 1). Devising a $O(1/\sqrt{\epsilon})$ algorithm under the less restrictive assumption $A_2$ remains an open problem.
It is important to note that the linear time algorithm in [11, Section 5.5.1] is the key to obtaining a $O(nd/\sqrt{\epsilon})$ computational complexity for binary SVMs with bias mentioned in Section 5.1. However, this method has been rediscovered independently by many authors (including us), with the earliest known reference to the best of our knowledge being [14] in 1990. Some recent work in optimization [15] has focused on improving the practical performance, while in machine learning [16] gave an expected linear time algorithm via randomized median finding.

Choosing an optimizer for a given machine learning task is a trade-off between a number of potentially conflicting requirements. CPM are one popular choice but there are others. If one is interested in classification accuracy alone, without requiring deterministic guarantees, then online to batch conversion techniques combined with stochastic subgradient descent are a good choice [17]. While the dependence on $\epsilon$ is still $\Omega(1/\epsilon)$ or worse [18], one gets bounds independent of $n$. However, as we pointed out earlier, these algorithms are applicable only when the empirical risk decomposes over the examples.

On the other hand, one can employ coordinate descent in the dual as is done in the Sequential Minimal Optimization (SMO) algorithm of [19]. However, as [20] show, if the kernel matrix obtained by stacking $x_i$ into a matrix $X$ and $X^\top X$ is not strictly positive definite, then SMO requires $O(n/\epsilon)$ iterations with each iteration costing $O(nd)$ effort. However, when the kernel matrix is strictly positive definite, then one can obtain an $O(n^2\log(1/\epsilon))$ bound on the number of iterations, which has better dependence on $\epsilon$, but is prohibitively expensive for large $n$. Even better dependence on $\epsilon$ can be achieved by using interior point methods [21] which require only $O(\log(\log(1/\epsilon)))$ iterations, but the time complexity per iteration is $O(\min\{n^2d, d^2n\})$.

References


