A Linearly Convergent Linear-Time First-Order Algorithm for Support Vector Classification with a Core Set Result

Piyush Kumar

Department of Computer Science, Florida State University, Tallahassee, FL 32306-4530, USA,
piyush@cs.fsu.edu

E. Alper Yıldırım

Department of Industrial Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey,
yildirim@bilkent.edu.tr

We present a simple, first-order approximation algorithm for the support vector classification problem. Given a pair of linearly separable data sets and $\epsilon \in (0, 1)$, the proposed algorithm computes a separating hyperplane whose margin is within a factor of $(1 - \epsilon)$ of that of the maximum-margin separating hyperplane. We discuss how our algorithm can be extended to nonlinearly separable and inseparable data sets. The running time of our algorithm is linear in the number of data points and in $1/\epsilon$. In particular, the number of support vectors computed by the algorithm is bounded above by $O(\zeta/\epsilon)$ for all sufficiently small $\epsilon > 0$, where ζ is the square of the ratio of the distances between the farthest and closest points in the two data sets. Furthermore, we establish that our algorithm exhibits linear convergence. We adopt the real number model of computation in our analysis.

Key words: Support vector machines, support vector classification, Frank-Wolfe algorithm, approximation algorithms, core sets, linear convergence.

AMS Subject Classification: 65K05, 90C20, 90C25.

History: Submitted April, 2009.

---

1. Introduction

Support vector machines (SVMs) are one of the most commonly used methodologies for classification, regression, and outlier detection. Given a pair of linearly separable data sets $\mathcal{P} \subset \mathbb{R}^n$ and $\mathcal{Q} \subset \mathbb{R}^n$, the support vector classification problem asks for the computation of a hyperplane that separates $\mathcal{P}$ and $\mathcal{Q}$ with the largest margin. Using kernel functions, the support vector classification problem can also be extended to nonlinearly separable data sets. Furthermore, classification errors can be incorporated into the problem to handle inseparable

data sets. SVMs have proven to be very successful in various real world applications including data mining, human computer interaction, image processing, bio-informatics, graphics, visualization, robotics, and many others [28, 8]. In theory, large margin separation implies good generalization bounds [8].

Support vector classification problem can be formulated as a convex quadratic programming problem (see Section 2), which can, in theory, be solved in polynomial time using interior-point methods. In practice, however, the resulting optimization problem is usually too large to be solved using direct methods. Therefore, previous research on solution approaches has either focused on decomposition methods (see, e.g., [21, 22, 15, 29]) or on approximation algorithms (see, e.g., [16, 14, 7, 11]). In this paper, we take the latter approach and aim to compute a separating hyperplane whose margin is a close approximation to that of the maximum-margin separating hyperplane.

Given $\epsilon \in (0, 1)$, an ϵ-core set is a subset of the input data points $\mathcal{P}' \cup \mathcal{Q}'$, where $\mathcal{P}' \subseteq \mathcal{P}$ and $\mathcal{Q}' \subseteq \mathcal{Q}$, such that the maximum margin that separates $\mathcal{P}$ and $\mathcal{Q}$ is within a factor of $(1-\epsilon)$ of the maximum margin that separates $\mathcal{P}'$ and $\mathcal{Q}'$. Small core sets constitute the building blocks of efficient approximation algorithms for large-scale optimization problems. Recently, several approximation algorithms have been developed for various classes of geometric optimization problems based on the existence of small core sets [5, 17, 3, 25, 18, 1, 24, 31, 19]. Computational experience indicates that such algorithms are especially well-suited for large-scale instances, for which a moderately small accuracy (e.g., $\epsilon = 10^{-3}$) suffices.

In this paper, we propose a simple algorithm that computes an approximation to the maximum-margin hyperplane that separates a pair of linearly separable data sets $\mathcal{P}$ and $\mathcal{Q}$. Given $\epsilon \in (0, 1)$, our algorithm computes a $(1 - \epsilon)$-approximate solution, i.e., a hyperplane that separates $\mathcal{P}$ and $\mathcal{Q}$ with a margin larger than $(1-\epsilon)\mu^*$, where μ^* denotes the maximum margin. Our algorithm is an adaptation of the Frank-Wolfe algorithm [9] with Wolfe's away steps [30] applied to the dual formulation of the support vector classification problem, which coincides with the formulation of the problem of finding the closest pair of points in two disjoint polytopes (see Section 2). We establish that our algorithm computes a $(1 - \epsilon)$-approximate solution to the support vector classification problem in $O(\zeta/\epsilon)$ iterations, where ζ is the square of ratio of the distances between the farthest and closest points in $\mathcal{P}$ and $\mathcal{Q}$. We also discuss how our algorithm can be extended to the nonlinearly separable and inseparable data sets without sacrificing the iteration complexity. Since our algorithm relies only on the first-order approximation of the quadratic objective function, the computational cost of

each iteration is fairly low. In particular, we establish that the number of kernel function evaluations at each iteration is $O(|\mathcal{P}| + |\mathcal{Q}|)$. As a byproduct, our algorithm explicitly computes an ϵ-core set of size $O(\zeta/\epsilon)$. Finally, our algorithm exhibits linear convergence, which implies that the dual optimality gap at each iteration asymptotically decreases at a linear rate.

For the support vector classification problem, one of the earlier core-set-based approaches is due to [27, 26], in which the authors reformulate the problem as a variant of the minimum enclosing ball problem and apply earlier core-set-based approaches developed for this latter problem [3, 17]. Har-Peled et al. [14] use a direct algorithm, which, starting off with one point from each input set, adds one input point at each iteration until the maximum-margin hyperplane that separates this subset is a $(1 - \epsilon)$-approximate solution. They establish that this direct procedure terminates in $O(\zeta/\epsilon)$ iterations, which readily yields a core set bound of $O(\zeta/\epsilon)$. Despite the simplicity of their approach, the algorithm and the analysis require the strong assumption of the availability of an exact solver for the computation of the largest-margin separating hyperplane for smaller instances of the support vector classification problem at each iteration.

More recently, Clarkson [7] studies the general problem of maximizing a concave function over the unit simplex. The dual formulation of the support vector classification problem can be reformulated in this form at the expense of increasing the number of decision variables. More specifically, the problem of computing the closest pair of points in two disjoint polytopes is equivalent to that of computing the point with the smallest norm in the Minkowski difference of these two polytopes. Therefore, the support vector classification problem can be viewed as a special case in his framework. By introducing the concept of an *additive* ϵ-core set for the general problem, Clarkson establishes core set results for several variants of the Frank-Wolfe algorithm, including a version that uses *away* steps. In particular, Clarkson specializes his results to the linearly separable support vector classification problem to establish a core set size of $O(\zeta/\epsilon)$. Motivated by his results, Gärtner and Jaggi [11] focus on the problem of computing the closest pair of points in two disjoint polytopes. They observe that Gilbert's algorithm [12] that computes the point with the smallest norm in a polytope is precisely the Frank-Wolfe algorithm specialized to this problem (see also Section 3). They establish that the running time of this algorithm is linear in the number of points and in $1/\epsilon$, which asymptotically matches the running time of our algorithm. Furthermore, their algorithm computes a core set of size $O(\varsigma/\epsilon)$ for the support vector classification problem,

where ς is a geometric measure that satisfies $(\sqrt{\zeta} - 1)^2 \leq \varsigma \leq \zeta - 1$. They also establish a lower bound of $\varsigma/(2\epsilon) + 2$ on the size of an ϵ-core set. Using Clarkson's results, they prove that Clarkson's variant of the Frank-Wolfe algorithm with away steps computes a core set whose size is asymptotically twice this lower bound.

The variant of the Frank-Wolfe algorithm that uses away steps in [7] is different from the version that we adopt in this paper. In particular, Clarkson's algorithm starts off by computing the closest pair of points in the two input sets, which already is more expensive than the overall complexity of our algorithm for fixed $\epsilon > 0$. Furthermore, Clarkson assumes that each iterate of the algorithm is an optimal solution of the original problem on the smallest face of the unit simplex that contains this iterate (see Algorithms 4.2 and 5.1). Therefore, similar to [14], his algorithm requires an exact solver for smaller subproblems. This assumption enables Clarkson to prove the existence of core set sizes with smaller constants. In particular, Gärtner and Jaggi [11] also rely on this result to establish that the specialization of Clarkson's algorithm to the polytope distance problem computes a core set whose size is closer to the lower bound. In contrast, we simply apply the original Frank-Wolfe algorithm with away steps [30] to the support vector classification problem without any modifications. As such, our algorithm does not require an optimal solution of smaller subproblems at any stage. Our core set bound asymptotically matches the previous bounds and differs from the lower bound by a constant factor. The running time of our algorithm is linear in $1/\epsilon$ and the cost of each iteration is linear in the number of input points. Finally, we establish the nice property that our algorithm enjoys linear convergence, which is a property that is *not* in general satisfied by Gilbert's algorithm and hence the first algorithm of [11] (see, e.g., [13]). In summary, our main contribution in this paper is the proof of the existence of a small core set result for the support vector classification problem using a simple, first-order algorithm with good theoretical complexity bounds and desirable convergence properties that are not necessarily shared by other similar algorithms.

There are many solvers available on the Internet to solve the support vector classification problem (see, e.g., `http://www.support-vector-machines.org`) . We do not present computational results relative to these solvers for the following reasons: In contrast to our algorithm, the currently best known specialized solvers such as LIBSVM [6] also rely on the second-order information. Secondly, such solvers are constantly being improved in accordance with the advances in solution approaches. Finally, these solvers have been engineered and optimized over several years. Therefore, we believe that a comparison of our algorithm

with the best known implementations would not be fair and meaningful. On the other hand, our previous computational experience with similar algorithms indicates that this type of first-order algorithms exhibit good performance for large-scale instances if a moderate level of accuracy suffices (see, e.g., [31, 24, 19]). Moreover, none of the earlier papers that establish similar core set results (e.g., [7, 14, 11]) reports any computational experiments due to similar reasons. However, we should mention that there exist implementations of core-set based SVM solutions that seem promising [25].

This paper is organized as follows. In the remainder of this section, we define our notation. In Section 2, we discuss optimization formulations for the support vector classification problem for linearly separable, nonlinearly separable, and inseparable data sets. Section 3 describes the approximation algorithm and establishes the computational complexity, core set, and the linear convergence results. Finally, Section 4 concludes the paper.

Notation: Vectors are denoted by lower-case Roman letters. For a vector p, p_i denotes its ith component. Inequalities on vectors apply to each component. We reserve e^j for the jth unit vector, $\mathbf{1}_n$ for the n-dimensional vector of all ones, and I for the identity matrix in the appropriate dimension, which will always be clear from the context. Upper-case Roman letters are reserved for matrices and M_{ij} denotes the (i, j) component of the matrix M. We use $\log(\cdot)$ and $\text{sgn}(\cdot)$ to denote the natural logarithm and the sign function, respectively. For a set $\mathcal{S} \subset \mathbb{R}^n$, $\text{conv}(\mathcal{S})$ denotes the convex hull of $\mathcal{S}$. Functions and operators are denoted by upper-case Greek letters. Scalars except for m, n, and r are represented by lower-case Greek letters, unless they represent components of a vector or elements of a sequence of scalars, vectors, or matrices. We reserve i, j, and k for such indexing purposes. Upper-case script letters are used for all other objects such as sets and hyperplanes.

2. Optimization Formulations

2.1. Linearly Separable Case

Let $\mathcal{P} = \{p^1, \ldots, p^m\} \subset \mathbb{R}^n$ and $\mathcal{Q} = \{q^1, \ldots, q^r\} \subset \mathbb{R}^n$ denote the two linearly separable data sets, i.e., we assume that $\text{conv}(\mathcal{P}) \cap \text{conv}(\mathcal{Q}) = \emptyset$. We discuss the extensions to the nonlinearly separable and inseparable data sets in Sections 2.2 and 2.3, respectively.

Let us define $P = [p^1, \ldots, p^m] \in \mathbb{R}^{n \times m}$ and $Q = [q^1, \ldots, q^r] \in \mathbb{R}^{n \times r}$. The support vector

classification problem admits the following optimization formulation [4]:

$$(\mathbb{P}) \quad \max_{w,\alpha,\beta} \quad -\tfrac{1}{2}\|w\|^2 + \alpha - \beta$$
$$\text{s.t.}$$
$$P^T w - \alpha \mathbf{1}_m \geq 0,$$
$$-Q^T w + \beta \mathbf{1}_r \geq 0,$$

where $w \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$, and $\beta \in \mathbb{R}$ are the decision variables. The Lagrangian dual of $(\mathbb{P})$ is given by

$$(\mathbb{D}) \quad \min_{u,v} \quad \Psi(u,v) \; := \; \tfrac{1}{2}\|Pu - Qv\|^2$$
$$\text{s.t.}$$
$$(\mathbf{1}_m)^T u = 1,$$
$$(\mathbf{1}_r)^T v = 1,$$
$$u \geq 0,$$
$$v \geq 0,$$

where $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^r$ are the decision variables. Note that $(\mathbb{D})$ is precisely the formulation of the problem of finding the closest pair of points in $\mathrm{conv}(\mathcal{P})$ and $\mathrm{conv}(\mathcal{Q})$.

Since $(\mathbb{P})$ is a convex optimization problem with linear constraints, $(w^*, \alpha^*, \beta^*) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ is an optimal solution of $(\mathbb{P})$ if and only if there exist $u^* \in \mathbb{R}^m$ and $v^* \in \mathbb{R}^r$ such that

$$P^T w^* - \alpha^* \mathbf{1}_m \geq 0, \tag{1a}$$

$$-Q^T w^* + \beta^* \mathbf{1}_r \geq 0, \tag{1b}$$

$$Pu^* - Qv^* = w^*, \tag{1c}$$

$$(\mathbf{1}_m)^T u^* = 1, \tag{1d}$$

$$(\mathbf{1}_r)^T v^* = 1, \tag{1e}$$

$$u_i^*((p^i)^T w^* - \alpha^*) = 0, \quad i = 1, \ldots, m, \tag{1f}$$

$$v_j^*(\beta^* - (q^j)^T w^*) = 0, \quad j = 1, \ldots, r, \tag{1g}$$

$$u^* \geq 0, \tag{1h}$$

$$v^* \geq 0. \tag{1i}$$

If we sum over i in (1f) and j in (1g), we obtain

$$\alpha^* = (w^*)^T P u^*, \quad \beta^* = (w^*)^T Q v^*, \tag{2}$$

where we used (1d), and (1e). It follows from (1c) that

$$\beta^* + \|w^*\|^2 - \alpha^* = 0, \quad \text{or} \quad -(1/2)\|w^*\|^2 + \alpha^* - \beta^* = (1/2)\|w^*\|^2, \tag{3}$$

6

which implies that $(u^*, v^*) \in \mathbb{R}^m \times \mathbb{R}^r$ is an optimal solution of (𝔻) and that strong duality holds between (ℙ) and (𝔻) . Therefore, the optimal separating hyperplane is given by

$$\mathcal{H} := \{x \in \mathbb{R}^n : (w^*)^T x = \gamma^*\},$$

where $\gamma^* := (\alpha^* + \beta^*)/2$ and the maximum margin between $\mathrm{conv}(\mathcal{P})$ and $\mathrm{conv}(\mathcal{Q})$ is

$$\mu^* := \frac{\alpha^* - \beta^*}{\|w^*\|} = \|w^*\| = \|Pu^* - Qv^*\|. \tag{4}$$

2.2.   Nonlinearly Separable Case

One of the main advantages of support vector machines is their ability to incorporate the transformation of nonlinearly separable input sets to linearly separable input sets by using kernel functions. Kernel functions significantly expand the application of support vector machines.

Let $\mathcal{P}$ and $\mathcal{Q}$ be two input sets in $\mathbb{R}^n$ that are not linearly separable but can be separated by a nonlinear manifold. The main idea is to lift the input data to a higher dimensional inner product space $\mathcal{S}$ (called the feature space) so that the lifted input sets are linearly separable in $\mathcal{S}$. More specifically, let $\Phi : \mathbb{R}^n \to \mathcal{S}$ denote this transformation. One can then aim to linearly separate the new input sets $\mathcal{P}' := \{\Phi(p^1), \ldots, \Phi(p^m)\}$ and $\mathcal{Q}' := \{\Phi(q^1), \ldots, \Phi(q^r)\}$ in $\mathcal{S}$. The primal formulation (ℙ) can be accordingly modified for the lifted input set.

However, the explicit evaluation of the function Φ can be too costly or even intractable since the feature space $\mathcal{S}$ may be extremely high-dimensional or even infinite-dimensional. This observation restricts the use of the primal formulation (ℙ). On the other hand, the objective function of the corresponding dual formulation is given by

$$\Psi(u, v) \;=\; \sum_{i=1}^{m} \sum_{j=1}^{m} u_i u_j \langle \Phi(p^i), \Phi(p^j) \rangle - 2 \sum_{i=1}^{m} \sum_{j=1}^{r} u_i v_j \langle \Phi(p^i), \Phi(q^j) \rangle$$
$$+ \sum_{i=1}^{m} \sum_{j=1}^{r} v_i v_j \langle \Phi(q^i), \Phi(q^j) \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in $\mathcal{S}$. It follows that the dual objective function requires only the computation of inner products in $\mathcal{S}$ rather than the actual transformations themselves. Therefore, if we define a function $\kappa : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ by

$$\kappa(x, y) := \langle \Phi(x), \Phi(y) \rangle, \tag{5}$$

then it suffices to be able to evaluate the function κ, known as the kernel function, rather than the transformation Φ in order to solve the dual optimization problem. Note that we recover the linearly separable case by simply defining $\kappa(x, y) = x^T y$, which is known as the linear kernel.

The use of kernel functions enables one to separate nonlinearly separable data using the dual formulation. In contrast with the primal formulation ($\mathbb{P}$), the number of variables in the dual formulation depends only on $|\mathcal{P}|$ and $|\mathcal{Q}|$, but is entirely independent of the dimension of the feature space $\mathcal{S}$.

Similar to the linearly separable case, the optimal separating hyperplane in $\mathcal{S}$ is given by

$$\mathcal{H}' := \{y \in \mathcal{S} : \langle w^*, y \rangle = \gamma^* \},$$

where $\gamma^* = (\alpha^* + \beta^*)/2$. Unlike the linearly separable case, the explicit construction of $w^* \in \mathcal{S}$, in general, is not possible. However, by (1c),

$$w^* = \sum_{i=1}^{m} u_i^* \Phi(p^i) - \sum_{j=1}^{r} v_j^* \Phi(q^j), \tag{6}$$

which implies that $\langle w^*, \Phi(x) \rangle$ can be easily computed using the kernel function κ for any test point $x \in \mathbb{R}^n$.

2.3. Inseparable Case

In most applications of the support vector classification problem, it is not known a priori if the input sets are linearly or nonlinearly separable. Therefore, it is essential to modify the formulation of the support vector classification problem so that classification violations are allowed. Such violations are usually penalized using additional terms in the objective function. In this paper, we focus on the formulation that penalizes the sum of squared violations:

$$(\mathbb{IP}) \quad \max_{w,\alpha,\beta,\xi,\psi} \quad -\tfrac{1}{2}\langle w, w \rangle + \alpha - \beta - \tfrac{\chi}{2}\left(\sum_{i=1}^{m} \xi_i^2 + \sum_{j=1}^{r} \psi_j^2\right)$$
$$\text{s.t.}$$
$$\begin{aligned} \langle \Phi(p^i), w \rangle - \alpha &\geq -\xi_i, \quad i = 1, \ldots, m, \\ -\langle \Phi(q^j), w \rangle + \beta &\geq -\psi_j, \quad j = 1, \ldots, r, \end{aligned}$$

where $\chi > 0$ is the penalty parameter, and $\xi \in \mathbb{R}^m$ and $\psi \in \mathbb{R}^r$ denote the decision variables corresponding to the classification violations in $\mathcal{P}$ and $\mathcal{Q}$, respectively.

As observed in [10], the optimization problem ($\mathbb{IP}$) can be converted into a separable instance using the following transformation. Let $\tilde{\mathcal{S}} := \mathcal{S} \times \mathbb{R}^m \times \mathbb{R}^r$ with the inner product defined by $\langle (w^1, y^1, z^1), (w^2, y^2, z^2) \rangle := \langle w^1, w^2 \rangle + (y^1)^T(y^2) + (z^1)^T(z^2)$. Then, if we define

$$
\begin{aligned}
\tilde{w} &:= (w^T, \sqrt{\chi}\, \xi^T, \sqrt{\chi}\, \psi^T)^T, \\
\widetilde{\Phi}(p^i) &:= (\Phi(p^i)^T, (1/\sqrt{\chi})(e^i)^T, 0)^T, \quad i = 1, \ldots, m, \\
\widetilde{\Phi}(q^j) &:= (\Phi(q^j)^T, 0, -(1/\sqrt{\chi})(e^j)^T)^T, \quad j = 1, \ldots, r, \\
\tilde{\alpha} &:= \alpha, \\
\tilde{\beta} &:= \beta,
\end{aligned}
$$

it is easy to verify that the problem ($\mathbb{IP}$) can be formulated as the problem ($\mathbb{P}$) on the input sets $\widetilde{\mathcal{P}} := \{\widetilde{\Phi}(p^1), \ldots, \widetilde{\Phi}(p^m)\}$ and $\widetilde{\mathcal{Q}} := \{\widetilde{\Phi}(q^1), \ldots, \widetilde{\Phi}(q^r)\}$ with decision variables $(\tilde{w}, \tilde{\alpha}, \tilde{\beta})$. Furthermore, for each $x, y \in \mathcal{P} \cup \mathcal{Q}$, the kernel function for the transformed instance satisfies

$$
\tilde{\kappa}(x, y) = \kappa(x, y) + \frac{1}{\chi} \delta_{xy},
$$

where $\delta_{xy} = 1$ if $x = y$ and zero otherwise. Therefore, the modified kernel function can be easily computed and the dual formulation ($\mathbb{D}$) can be used to solve the inseparable support vector classification problem.

These observations indicate that the dual formulation ($\mathbb{D}$) can quite generally be used to solve the support vector classification problem. Therefore, similar to the previous studies in this field, our algorithm works exclusively with the dual formulation. We first present and analyze our algorithm for the linearly separable case and subsequently extend it to the nonlinearly separable case. The applicability of our algorithm for the inseparable case directly follows from the nonlinearly separable case using the transformation in this section.

3. The Algorithm

3.1. Linearly Separable Case

Let $\mathcal{P} = \{p^1, \ldots, p^m\} \subset \mathbb{R}^n$ and $\mathcal{Q} = \{q^1, \ldots, q^r\} \subset \mathbb{R}^n$ denote the two linearly separable data sets. In this section, we present and analyze our algorithm that computes an approximate solution to the dual problem ($\mathbb{D}$).

Note that the problem ($\mathbb{D}$) is a convex quadratic programming problem. The main difficulty in practical applications stems from the size of the data sets. In particular, the

matrix whose entries are given by $\kappa(x, y)$, where $x, y \in \mathcal{P} \cup \mathcal{Q}$, is typically huge and dense. Therefore, direct solution approaches are usually not applicable. In this paper, our focus is on computing an approximate solution of ($\mathbb{D}$) using a simple algorithm that is scalable with the size of the data.

Let us describe Algorithm 1 in more detail. The algorithm generates a sequence of improving estimates $(Pu^k, Qv^k) \in \text{conv}(\mathcal{P}) \times \text{conv}(\mathcal{Q})$ of the pair of closest points. The sequence is initialized by computing the closest point $q^{j*} \in \mathcal{Q}$ to $p^1 \in \mathcal{P}$ and then computing the closest point $p^{i*} \in \mathcal{P}$ to q^{j*}. Therefore, (p^{i*}, q^{j*}) constitutes the first term of the aforementioned sequence.

For each k, the points u^k and v^k lie on the unit simplices in $\mathbb{R}^m$ and $\mathbb{R}^r$, respectively. Therefore, (u^k, v^k) is a feasible solution of the dual problem ($\mathbb{D}$). At iteration k, the algorithm computes the minimizing vertex $p^{i'} \in \text{conv}(\mathcal{P})$ and the maximizing vertex $q^{j'} \in \text{conv}(\mathcal{Q})$ for the linear function $(w^k)^T x$, where $w^k := Pu^k - Qv^k$, and sets $z^k := p^{i'} - q^{j'}$. The "signed" distance between the parallel hyperplanes $\mathcal{H}_+^k := \{x \in \mathbb{R}^n : (w^k)^T x = (w^k)^T p^{i'}\}$ and $\mathcal{H}_-^k := \{x \in \mathbb{R}^n : (w^k)^T x = (w^k)^T q^{j'}\}$ is given by $(w^k)^T z^k / \|w^k\|$, which is clearly a lower bound on the maximum margin μ^* between $\text{conv}(\mathcal{P})$ and $\text{conv}(\mathcal{Q})$. Note that a negative distance indicates that the current estimate of the hyperplane does not yet separate $\text{conv}(\mathcal{P})$ and $\text{conv}(\mathcal{Q})$. Furthermore, $\|w^k\|$ is an upper bound on μ^* by the dual feasibility of (u^k, v^k). Therefore,

$$\frac{(w^k)^T(z^k)}{\|w^k\|} = (1 - \epsilon_+^k)\|w^k\| \le \mu^* = \|Pu^* - Qv^*\| \le \|w^k\|, \tag{7}$$

where (u^*, v^*) is an optimal solution of ($\mathbb{D}$). Since $\epsilon^k \ge \epsilon_+^k$, it follows that (u^k, v^k) is a $(1 - \epsilon^k)$-approximate solution of the support vector classification problem.

Let us now take the primal perspective and let us define (cf. (2))

$$\alpha^k := (w^k)^T Pu^k, \quad \beta^k := (w^k)^T Qv^k. \tag{8}$$

Note that $(w^k, \alpha^k, \beta^k) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ may not necessarily be a feasible solution of ($\mathbb{P}$). In fact, primal feasibility is achieved if only if (u^k, v^k) is an optimal solution of ($\mathbb{D}$) by (1). However, we now establish an upper bound on the primal infeasibility.

First of all, by Steps 28 and 29 of Algorithm 1, we have

$$(w^k)^T q^{j''} \le \beta^k \le (w^k)^T q^{j'}, \quad (w^k)^T p^{i'} \le \alpha^k \le (w^k)^T p^{i''}. \tag{9}$$

Furthermore, we have

$$(w^k)^T(p^{i'} - q^{j'}) = (w^k)^T z^k = (1 - \epsilon_+^k)\|w^k\|^2 \ge (1 - \epsilon^k)\|w^k\|^2 \tag{10}$$

10

Algorithm 1 The algorithm that computes a $(1 - \epsilon)$-approximate solution to the support vector classification problem.

Require: Input data sets $\mathcal{P} = \{p^1, \ldots, p^m\} \subset \mathbb{R}^n$, $\mathcal{Q} = \{q^1, \ldots, q^r\} \subset \mathbb{R}^n$, and $\epsilon > 0$

1: $k \leftarrow 0$;
2: $j_* \leftarrow \arg\min_{j=1,\ldots,r} \|p^1 - q^j\|^2$;
3: $i_* \leftarrow \arg\min_{i=1,\ldots,m} \|p^i - q^{j*}\|^2$;
4: $u_i^k \leftarrow 0, \quad i = 1, \ldots, m; \quad u_{i_*}^k \leftarrow 1$;
5: $v_j^k \leftarrow 0, \quad j = 1, \ldots, r; \quad v_{j_*}^k \leftarrow 1$;
6: $w^k \leftarrow p^{i*} - q^{j*}$;
7: $i' \leftarrow \arg\min_{i=1,\ldots,m}(w^k)^T p^i; \quad i'' \leftarrow i_*$;
8: $j' \leftarrow \arg\max_{j=1,\ldots,r}(w^k)^T q^j; \quad j'' \leftarrow j_*$;
9: $z^k \leftarrow p^{i'} - q^{j'}; \quad y^k \leftarrow w^k$;
10: $\epsilon_+^k \leftarrow 1 - [(z^k)^T(w^k)/(w^k)^T(w^k)]; \quad \epsilon_-^k \leftarrow 0$;
11: $\epsilon^k \leftarrow \max\{\epsilon_+^k, \epsilon_-^k\}$;
12: While $\epsilon^k > \epsilon$, do
13: **loop**
14:    **if** $\epsilon^k > \epsilon_-^k$ **then**
15:      $d^k \leftarrow w^k - z^k$;
16:      $\lambda^k \leftarrow \min\{1, (w^k)^T(d^k)/(d^k)^T(d^k)\}$;
17:      $u^{k+1} \leftarrow (1 - \lambda^k)u^k + \lambda^k e^{i'}$;
18:      $v^{k+1} \leftarrow (1 - \lambda^k)v^k + \lambda^k e^{j'}$;
19:      $w^{k+1} \leftarrow (1 - \lambda^k)w^k + \lambda^k z^k$;
20:    **else**
21:      $b^k \leftarrow y^k - w^k$;
22:      $\lambda^k \leftarrow \min\{(w^k)^T(b^k)/(b^k)^T(b^k), u_{i''}^k/(1 - u_{i''}^k), v_{j''}^k/(1 - v_{j''}^k)\}$;
23:      $u^{k+1} \leftarrow (1 + \lambda^k)u^k - \lambda^k e^{i''}$;
24:      $v^{k+1} \leftarrow (1 + \lambda^k)v^k - \lambda^k e^{j''}$;
25:      $w^{k+1} \leftarrow (1 + \lambda^k)w^k - \lambda^k y^k$;
26:    **end if**
27:    $k \leftarrow k + 1$;
28:    $i' \leftarrow \arg\min_{i=1,\ldots,m}(w^k)^T p^i; \quad i'' \leftarrow \arg\max_{i:u_i^k>0}(w^k)^T p^i$;
29:    $j' \leftarrow \arg\max_{j=1,\ldots,r}(w^k)^T q^j; \quad j'' \leftarrow \arg\min_{j:v_j^k>0}(w^k)^T q^j$;
30:    $z^k \leftarrow p^{i'} - q^{j'}; \quad y^k \leftarrow p^{i''} - q^{j''}$;
31:    $\epsilon_+^k \leftarrow 1 - [(z^k)^T(w^k)/(w^k)^T(w^k)]; \quad \epsilon_-^k \leftarrow [(y^k)^T(w^k)/(w^k)^T(w^k)] - 1$;
32:    $\epsilon^k \leftarrow \max\{\epsilon_+^k, \epsilon_-^k\}$;
33: **end loop**
34: $\mathcal{X} \leftarrow \{p^i : u_i^k > 0\} \cup \{q^j : v_j^k > 0\}$;
35: $\gamma \leftarrow (1/2)((w^k)^T p^{i'} + (w^k)^T q^{j'})$;
36: **Output** $u^k, v^k, \mathcal{X}, w^k, \gamma$.

---

and

$$(w^k)^T(p^{i''} - q^{j''}) = (w^k)^T y^k = (1 + \epsilon_-^k)\|w^k\|^2 \leq (1 + \epsilon^k)\|w^k\|^2. \tag{11}$$

By (9), for any $p^i \in \mathcal{P}$,

$$\begin{aligned}
(w^k)^T p^i - \alpha^k &\geq (w^k)^T \left(p^{i'} - p^{i''} \right), \\
&= (w^k)^T \left(p^{i'} - q^{j'} + q^{j'} - p^{i''} \right), \\
&\geq (1 - \epsilon^k)\|w^k\|^2 + (w^k)^T \left(q^{j''} - p^{i''} \right), \\
&\geq -2\epsilon^k\|w^k\|^2,
\end{aligned} \tag{12}$$

where we used (10) and (11). Similarly, for any $q^j \in \mathcal{Q}$, it is easy to verify that

$$\beta^k - (w^k)^T q^j \geq -2\epsilon^k\|w^k\|^2. \tag{13}$$

Furthermore, by definition of $p^{i''}$, for each $p^i \in \mathcal{P}$ such that $u_i^k > 0$, we have

$$(w^k)^T p^i - \alpha^k \leq (w^k)^T \left(p^{i''} - p^{i'} \right) \leq 2\epsilon^k\|w^k\|^2, \tag{14}$$

where we used (9) and (12), which, together with (12), implies that

$$\left| (w^k)^T p^i - \alpha^k \right| \leq 2\epsilon^k\|w^k\|^2 \quad \text{for all } i \in \{1, \ldots, m\} \text{ such that } u_i^k > 0. \tag{15}$$

Using the definition of $q^{j''}$, a similar derivation reveals that

$$\left| \beta^k - (w^k)^T q^j \right| \leq 2\epsilon^k\|w^k\|^2 \quad \text{for all } j \in \{1, \ldots, r\} \text{ such that } v_j^k > 0. \tag{16}$$

It follows from (12) and (13) that (w^k, α^k, β^k) is a feasible solution of a perturbation of the primal problem ($\mathbb{P}$). Similarly, $(w^k, \alpha^k, \beta^k, u^k, v^k)$ satisfies the approximate version of the optimality conditions (1), i.e., the conditions (1a), (1b), (1f), and (1g) are approximately satisfied while the remaining ones are exactly satisfied. This observation is crucial in establishing the linear convergence of Algorithm 1 in Section 3.3.

Having established the properties of the iterates generated by Algorithm 1, we now explain how iterates are updated at each iteration. At iteration k, the algorithm computes the two parameters ϵ_+^k and ϵ_-^k by (10) and (11). Since

$$\begin{aligned}
(1 - \epsilon_+^k)\|w^k\|^2 &= (w^k)^T(z^k) = (w^k)^T p^{i'} - (w^k)^T q^{j'} &\leq \alpha^k - \beta^k = \|w^k\|^2, \\
(1 + \epsilon_-^k)\|w^k\|^2 &= (w^k)^T(y^k) = (w^k)^T p^{i''} - (w^k)^T q^{j''} &\geq \alpha^k - \beta^k = \|w^k\|^2,
\end{aligned}$$

where we used (8), it follows that $\epsilon_+^k \geq 0$ and $\epsilon_-^k \geq 0$.

If $\epsilon^k = \epsilon_+^k$, Algorithm 1 sets $(u^{k+1}, v^{k+1}) = (1 - \lambda^k)(u^k, v^k) + \lambda^k(e^{i'}, e^{j'})$, where λ^k is given by

$$\lambda^k := \arg\min_{\lambda \in [0,1]} \Psi((1 - \lambda)(u^k, v^k) + \lambda(e^{i'}, e^{j'})). \tag{17}$$

The range of λ ensures the dual feasibility of (u^{k+1}, v^{k+1}). Note that $w^{k+1} = Pu^{k+1} - Qv^{k+1} = (1 - \lambda^k)w^k + \lambda^k z^k$, which implies that the algorithm computes the point with the smallest norm on the line segment joining w^k and z^k in this case. It is straightforward to verify that the choice of λ^k in Algorithm 1 satisfies (17).

On the other hand, if $\epsilon^k = \epsilon_+^k$, Algorithm 1 uses the update $(u^{k+1}, v^{k+1}) = (1 + \lambda^k)(u^k, v^k) - \lambda^k(e^{i''}, e^{j''})$, where λ^k is given by

$$\lambda^k := \arg\min_{\lambda \in [0,\lambda_{\max}^k]} \Psi((1 + \lambda)(u^k, v^k) - \lambda(e^{i''}, e^{j''})). \tag{18}$$

Here, $\lambda_{\max}^k := \min\{u_{i''}^k/(1 - u_{i''}^k), v_{j''}^k/(1 - v_{j''}^k)\}$ is chosen to ensure the nonnegativity (and hence the dual feasibility) of (u^{k+1}, v^{k+1}). In this case, $w^{k+1} = Pu^{k+1} - Qv^{k+1} = (1+\lambda^k)w^k - \lambda^k y^k = w^k + \lambda^k(w^k - y^k)$, i.e., w^{k+1} is given by the point with the smallest norm on the line segment joining w^k and $w^k + \lambda_{\max}^k(w^k - y^k)$.

Algorithm 1 is the Frank-Wolfe algorithm [9] with Wolfe's away steps [30] applied to the support vector classification problem. The algorithm is based on linearizing the quadratic objective function $f(u, v)$ at the current iterate (u^k, v^k) and solving a linear programming problem at each iteration. From (u^k, v^k), the algorithm either moves towards the vertex $(e^{i'}, e^{j'})$ of the dual feasible region that minimizes this linear approximation or away from the vertex $(e^{i''}, e^{j''})$ that maximizes this approximation, where the maximization is restricted to the smallest face of the feasible region containing (u^k, v^k). In either case, the step size is determined so as to minimize the dual objective function (see (17) and (18)). As such, Algorithm 1 only relies on the first-order information about the optimization problem $(\mathbb{D})$.

We discuss the relation of Algorithm 1 with other similar algorithms developed for the problem of computing the closest pair of points in two disjoint polytopes. One of the earliest iterative algorithms known for this problem is due to Gilbert [12]. Similar to Algorithm 1, Gilbert's algorithm also generates a sequence of improving estimates for the pair of closest points. In particular, the updates used in his algorithm coincide exactly with our update (17) at an add-iteration. This implies that Gilbert's algorithm is precisely the same as the original Frank-Wolfe algorithm [9] without the away steps. This observation along with

13

the fact that Gilbert's algorithm computes a small ϵ-core set appeared recently in [11]. However, it is well-known that the Frank-Wolfe algorithm does not enjoy linear convergence in general [13]. Another related iterative algorithm is due to Mitchell et al. [20]. This algorithm uses a very similar update to our update (18) at a decrease- or drop-iteration. The only difference is that they perform their line search on $w^k + \lambda(z^k - y^k)$ as opposed to $w^k + \lambda(w^k - y^k)$ used in our line search. Keerthi et al. [16] propose combining these two updates. They also establish that their algorithm computes an approximate solution in a finite number of iterations. However, they neither give a bound on the number of iterations to achieve a desired level of accuracy nor do they establish a core set result. Finally, it is not clear if their algorithm exhibits linear convergence.

3.2. Analysis of the Algorithm

In this section, we establish the computational complexity of Algorithm 1. The analysis is driven by establishing a lower bound on the improvement of the dual objective function $\Psi(u, v)$ evaluated at successive iterates (u^k, v^k) generated by the algorithm.

Let us first define a parameter δ by

$$\delta := \frac{1}{2} \max_{i=1,\ldots,m; \ j=1,\ldots,r} \left\| p^i - q^j \right\|^2. \tag{19}$$

It follows that the optimal value of $(\mathbb{D})$ satisfies

$$\Psi^* := \Psi(u^*, v^*) \leq \delta, \tag{20}$$

where (u^*, v^*) denotes any optimal solution of $(\mathbb{D})$.

In Algorithm 1, we say that iteration k is an add-iteration if $\epsilon^k = \epsilon_+^k$. If $\epsilon^k = \epsilon_-^k$ and $\lambda^k < \lambda_{\max}^k$, we call it a decrease-iteration. Finally, if $\epsilon^k = \epsilon_-^k$ and $\lambda^k = \lambda_{\max}^k$, then iteration k is a drop-iteration, in which case at least one of the positive components of u^k and/or v^k drops to zero. The first lemma establishes a lower bound on the improvement at each add- or decrease-iteration.

Lemma 3.1 *Suppose that iteration k of Algorithm 1 is an add- or decrease-iteration. Then,*

$$\Psi^{k+1} \leq \Psi^k \left(1 - \frac{(\epsilon^k)^2 \Psi^*}{(\epsilon^k)^2 \Psi^* + \delta} \right), \tag{21}$$

where $\Psi^k := \Psi(u^k, v^k)$.

14

Proof. Note that

$$\|(1-\lambda)g + \lambda h\|^2 = (1-\lambda)\|g\|^2 + \lambda\|h\|^2 - \lambda(1-\lambda)\|g-h\|^2, \tag{22}$$

for all $g, h \in \mathbb{R}^n$ and all $\lambda \in \mathbb{R}$.

Let us first consider an add-iteration. In this case, $(u^{k+1}, v^{k+1}) = (1-\lambda^k)(u^k, v^k) + \lambda^k(e^{i'}, e^{j'})$, where λ^k is given by (17). By (22), we have

$$
\begin{aligned}
\Psi((1-\lambda)(u^k, v^k) + \lambda(e^{i'}, e^{j'})) &= (1/2)\left\|(1-\lambda)(Pu^k - Qv^k) + \lambda(p^{i'} - q^{j'})\right\|^2, \\
&= (1/2)\left\|(1-\lambda)w^k + \lambda z^k\right\|^2, \tag{23} \\
&= (1/2)\left[(1-\lambda)\|w^k\|^2 + \lambda\|z^k\|^2 - \lambda(1-\lambda)\|w^k - z^k\|^2\right],
\end{aligned}
$$

which implies that the unique unconstrained minimizer of the problem in (17) is given by

$$\lambda^* = \frac{\|w^k\|^2 - (w^k)^T(z^k)}{\|w^k - z^k\|^2}. \tag{24}$$

Let us first focus on λ^*. We can write

$$z^k = z_*^k + z_{**}^k,$$

where z_*^k is the projection of z^k onto $\mathrm{span}(\{w^k\})$. Therefore,

$$
\begin{aligned}
\|w^k - z^k\|^2 &= \|w^k\|^2 - 2(w^k)^T(z^k) + \|z_*^k\|^2 + \|z_{**}^k\|^2, \\
&= \|w^k\|^2(1 - 2(1-\epsilon^k) + (1-\epsilon^k)^2) + \|z_{**}^k\|^2, \tag{25} \\
&= (\epsilon^k\|w^k\|)^2 + \|z_{**}^k\|^2,
\end{aligned}
$$

where we used the fact that $(w^k)^T(z^k) = (1-\epsilon^k)\|w^k\|^2 = \mathrm{sgn}((w^k)^T(z^k))\|w^k\|\|z_*^k\|$ in the second equation. By (24) and (25),

$$\lambda^* = \frac{\|w^k\|^2 - (w^k)^T(z^k)}{\|w^k - z^k\|^2} = \frac{\epsilon^k\|w^k\|^2}{(\epsilon^k\|w^k\|)^2 + \|z_{**}^k\|^2} \geq 0. \tag{26}$$

Let us first assume that $\lambda^* \in (0, 1)$, which implies that $\lambda^k = \lambda^*$. By (23), (25), and (26), we

15

have

$$\begin{aligned}
\Psi^{k+1} &= (1/2)\left[(1-\lambda^k)\|w^k\|^2 + \lambda^k\|z^k\|^2 - \lambda^k(1-\lambda^k)\|w^k - z^k\|^2\right], \\
&= (1/2)\left[(1-\lambda^k)\|w^k\|^2 + \lambda^k(\|z_*^k\|^2 + \|z_{**}^k\|^2) - \lambda^k(1-\lambda^k)((\epsilon^k\|w^k\|)^2 + \|z_{**}^k\|^2)\right], \\
&= (1/2)\left[(1-\lambda^k)\|w^k\|^2 + \lambda^k((1-\epsilon^k)^2\|w^k\|^2 + \|z_{**}^k\|^2) - (1-\lambda^k)\epsilon^k\|w^k\|^2\right], \\
&= (1/2)\left[(1-\lambda^k)\|w^k\|^2 + \lambda^k(1-2\epsilon^k)\|w^k\|^2 + \epsilon^k\|w^k\|^2 - (1-\lambda^k)\epsilon^k\|w^k\|^2\right], \\
&= (1/2)\|w^k\|^2(1-\epsilon^k\lambda^k), \\
&= \Psi^k\left(1 - \frac{(\epsilon^k\|w^k\|)^2}{(\epsilon^k\|w^k\|)^2 + \|z_{**}^k\|^2}\right), \\
&\leq \Psi^k\left(1 - \frac{(\epsilon^k\|w^k\|)^2}{(\epsilon^k\|w^k\|)^2 + 2\delta}\right),
\end{aligned}$$

where we used the relationship $\|z_{**}^k\|^2 \leq \|z^k\|^2 \leq 2\delta$ to derive the last inequality. Note that the expression on the right-hand side of the last inequality is a decreasing function of $\|w^k\|^2$. Since $\|w^k\|^2 \geq 2\Psi^*$, we obtain

$$\Psi^{k+1} \leq \Psi^k\left(1 - \frac{2(\epsilon^k)^2\Psi^*}{2(\epsilon^k)^2\Psi^* + 2\delta}\right),$$

which establishes (21) for this case.

Suppose now that $\lambda^* \geq 1$, which implies that $\lambda^k = 1$ by convexity (see (17)). By (26), this case happens if and only if

$$\epsilon^k\|w^k\|^2 \geq (\epsilon^k\|w^k\|)^2 + \|z_{**}^k\|^2,$$

which is equivalent to

$$\|z_{**}^k\|^2 \leq \|w^k\|^2\epsilon^k(1-\epsilon^k). \tag{27}$$

This implies that this case can happen only when $\epsilon^k \in (0,1)$. Since $(u^{k+1}, v^{k+1}) = (e^{i'}, e^{j'})$, we have

$$\Psi^{k+1} = (1/2)\|Pe^{i'} - Qe^{j'}\|^2 = (1/2)\|p^{i'} - q^{j'}\|^2 = (1/2)\|z^k\|^2.$$

By (27), we have

$$\begin{aligned}
\|z^k\|^2 &= \|z_*^k\|^2 + \|z_{**}^k\|^2, \\
&= (1-\epsilon^k)^2\|w^k\|^2 + \|z_{**}^k\|^2, \\
&\leq \|w^k\|^2[(1-\epsilon^k)^2 + \epsilon^k(1-\epsilon^k)], \\
&= \|w^k\|^2(1-\epsilon^k),
\end{aligned}$$

16

which implies that

$$\Psi^{k+1} \leq \Psi^k (1 - \epsilon^k).$$

Since $\epsilon^k \in (0, 1)$ in this case and $\delta \geq \Psi^*$, it is easy to verify that

$$\Psi^{k+1} \leq \Psi^k (1 - \epsilon^k) \leq \Psi^k \left(1 - \frac{(\epsilon^k)^2 \Psi^*}{(\epsilon^k)^2 \Psi^* + \delta} \right),$$

which implies that (21) is also satisfied in this case. This establishes the assertion at an add-iteration.

Let us now consider a decrease-iteration. In this case, $(u^{k+1}, v^{k+1}) = (1 + \lambda^k)(u^k, v^k) - \lambda^k(e^{i''}, e^{j''})$, where $\lambda^k < \lambda_{\max}^k$ is given by (18). By (22),

$$
\begin{aligned}
\Psi((1 + \lambda)(u^k, v^k) - \lambda(e^{i''}, e^{j''})) &= (1/2) \left\| (1 + \lambda)(Pu^k - Qv^k) - \lambda(p^{i''} - q^{j''}) \right\|^2, \\
&= (1/2) \left[(1 + \lambda)\|w^k\|^2 - \lambda \|y^k\|^2 + \lambda(1 + \lambda)\|w^k - y^k\|^2 \right],
\end{aligned}
$$

which readily implies that the unique unconstrained minimizer of the problem in (18) is given by

$$\lambda_* = \frac{(w^k)^T(y^k) - \|w^k\|^2}{\|w^k - y^k\|^2}.$$

Similarly, let

$$y^k = y_*^k + y_{**}^k,$$

where y_*^k is the projection of y^k onto $\mathrm{span}(\{w^k\})$. Therefore,

$$
\begin{aligned}
\|w^k - y^k\|^2 &= \|w^k\|^2 - 2(w^k)^T(y^k) + \|y_*^k\|^2 + \|y_{**}^k\|^2, \\
&= \|w^k\|^2 (1 - 2(1 + \epsilon^k) + (1 + \epsilon^k)^2) + \|y_{**}^k\|^2, \\
&= (\epsilon^k \|w^k\|)^2 + \|y_{**}^k\|^2,
\end{aligned}
$$

where we used $(w^k)^T(y^k) = (1 + \epsilon^k)\|w^k\|^2 = \mathrm{sgn}((w^k)^T(y^k))\|w^k\|\|y_*^k\|$ in the second equation. Therefore,

$$\lambda_* = \frac{\epsilon^k \|w^k\|^2}{(\epsilon^k \|w^k\|)^2 + \|y_{**}^k\|^2} \geq 0,$$

which implies that $\lambda^k = \lambda_* < \lambda_{\max}^k$ since it is a decrease-iteration. Similar to the first case

17

in an add-iteration, we obtain

$$
\begin{aligned}
\Psi^{k+1} &= (1/2)[(1+\lambda^k)\|w^k\|^2 - \lambda^k\|y^k\|^2 + \lambda^k(1+\lambda^k)\|w^k - y^k\|^2], \\
&= (1/2)[(1+\lambda^k)\|w^k\|^2 - \lambda^k(\|y_*^k\|^2 + \|y_{**}^k\|^2) + \lambda^k(1+\lambda^k)((\epsilon^k\|w^k\|)^2 + \|y_{**}^k\|^2)], \\
&= (1/2)[(1+\lambda^k)\|w^k\|^2 - \lambda^k((1+\epsilon^k)^2\|w^k\|^2 + \|y_{**}^k\|^2) + (1+\lambda^k)\epsilon^k\|w^k\|^2], \\
&= (1/2)[(1+\lambda^k)\|w^k\|^2 - \lambda^k(1+2\epsilon^k)\|w^k\|^2 - \epsilon^k\|w^k\|^2 + (1+\lambda^k)\epsilon^k\|w^k\|^2], \\
&= (1/2)\|w^k\|^2(1-\epsilon^k\lambda^k), \\
&= \Psi^k\left(1 - \frac{(\epsilon^k\|w^k\|)^2}{(\epsilon^k\|w^k\|)^2 + \|y_{**}^k\|^2}\right), \\
&\leq \Psi^k\left(1 - \frac{(\epsilon^k\|w^k\|)^2}{(\epsilon^k\|w^k\|)^2 + 2\delta}\right),
\end{aligned}
$$

where we used the relationship $\|y_{**}^k\|^2 \leq \|y^k\|^2 \leq 2\delta$ to derive the last inequality. The assertion follows from similar arguments as in the first case in an add-iteration. ⌣

Lemma 3.1 provides a lower bound on the improvement at each add- or decrease-iteration. Clearly, the objective function does not increase at a drop-iteration. However, the improvement in the objective function can longer be bounded from below at such an iteration since $\lambda_{\max}^k$ can be arbitrarily small. Nevertheless, we can still establish an upper bound on the number of iterations required by Algorithm 1 to compute a $(1-\epsilon)$-approximate solution. To this end, let us define

$$
\theta(\epsilon) = \min\{k : \epsilon^k \leq \epsilon\}. \tag{28}
$$

Similarly, let $\phi(\epsilon)$ and $\varphi(\epsilon)$ denote the number of drop-iterations and the total number of add- and decrease-iterations in the first $\theta(\epsilon)$ iterations of Algorithm 1. Clearly, $\theta(\epsilon) = \phi(\epsilon) + \varphi(\epsilon)$.

Theorem 3.1 *Given $\epsilon \in (0,1)$, Algorithm 1 computes a $(1-\epsilon)$-approximate solution to the support vector classification problem in*

$$
\theta(\epsilon) \leq
\begin{cases}
2 + 10\left(\frac{\delta}{f^*}\right)\log\left(\frac{\delta}{f^*}\right), & \text{if } \epsilon \in [1/2, 1), \\
6 + 10\left(\frac{\delta}{f^*}\right)\log\left(\frac{\delta}{f^*}\right) + 32\frac{\delta}{\epsilon f^*}, & \text{if } \epsilon \in (0, 1/2).
\end{cases}
\tag{29}
$$

iterations.

Proof. Let us first consider $\theta(1/2)$. By (19) and (20),

$$
\Psi^* \leq \Psi^0 \leq \delta.
$$

By Lemma 3.1, at each add- or decrease-iteration with $\epsilon^k > 1/2$, we have

$$\Psi^* \le \Psi^{k+1} \le \Psi^k \left(1 - \frac{(\epsilon^k)^2 \Psi^*}{(\epsilon^k)^2 \Psi^* + \delta}\right) \le \Psi^k \left(1 - \frac{(1/4)\Psi^*}{(1/4)\Psi^* + \delta}\right),$$

which implies that

$$\Psi^* \le \Psi^{\theta(1/2)} \le \Psi^0 \left(1 - \frac{(1/4)\Psi^*}{(1/4)\Psi^* + \delta}\right)^{\varphi(1/2)} \le \delta \left(1 - \frac{(1/4)\Psi^*}{(1/4)\Psi^* + \delta}\right)^{\varphi(1/2)}.$$

By taking logarithms, rearranging the terms, and using the inequality $\log(1+x) \ge x/(x+1)$, we obtain

$$\varphi(1/2) \le \frac{\log\left(\frac{\delta}{\Psi^*}\right)}{\log\left(1 + \frac{\Psi^*}{4\delta}\right)} \le \left(1 + \frac{4\delta}{\Psi^*}\right) \log\left(\frac{\delta}{\Psi^*}\right) \le 5 \left(\frac{\delta}{\Psi^*}\right) \log\left(\frac{\delta}{\Psi^*}\right). \tag{30}$$

At each drop-iteration, we can only guarantee that $\Psi^{k+1} \le \Psi^k$. However, at each such iteration, at least one component of u or v drops to zero. Therefore, every such iteration can be coupled with the most recent add- or decrease-iteration in which that component increased from zero. In order to account for the initial two positive entries of (u, v), we can add two to the total iteration count. It follows that

$$\theta(1/2) \le 2\varphi(1/2) + 2, \tag{31}$$

which, together with (30), establishes (29) for $\epsilon \in [1/2, 1)$.

We now consider $\theta(2^{-\tau})$ for $\tau = 2, 3, \ldots$. Let $\tilde{k} := \theta(2^{1-\tau})$. We first establish an upper bound on the number of add- and decrease-iterations between the iterate $\tilde{k}$ and the iterate $\theta(2^{-\tau})$. Since $\epsilon^{\tilde{k}} \le 2^{1-\tau}$, we have

$$\left(1 - \frac{1}{2^{\tau-1}}\right) \Psi^{\tilde{k}} \le (1 - \epsilon^{\tilde{k}})\Psi^{\tilde{k}} \le \Psi^* \le \Psi^{\tilde{k}}.$$

Similarly, at each add- or decrease-iteration k with $\epsilon^k > 2^{-\tau}$, we have

$$\Psi^* \le \Psi^{k+1} \le \Psi^k \left(1 - \frac{(\epsilon^k)^2 \Psi^*}{(\epsilon^k)^2 \Psi^* + \delta}\right) \le \Psi^k \left(1 - \frac{(2^{-2\tau})\Psi^*}{(2^{-2\tau})\Psi^* + \delta}\right),$$

which implies that

$$\left(1 - \frac{1}{2^{\tau-1}}\right) \Psi^{\tilde{k}} \le \Psi^* \le \Psi^{\theta(2^{-\tau})} \le \Psi^{\tilde{k}} \left(1 - \frac{(2^{-2\tau})\Psi^*}{(2^{-2\tau})\Psi^* + \delta}\right)^{(\varphi(2^{-\tau}) - \varphi(2^{1-\tau}))}.$$

Once again, by taking logarithms and rearranging the terms, we obtain

$$
\begin{aligned}
\varphi(2^{-\tau}) - \varphi(2^{1-\tau}) \quad &\le \quad \frac{\log\left(1 + \frac{1}{(2^{\tau-1})-1}\right)}{\log\left(1 + \frac{(2^{-2\tau})\Psi^*}{\delta}\right)} \quad &\le \quad \frac{1}{(2^{\tau-1})-1}\left(1 + \frac{\delta}{(2^{-2\tau})\Psi^*}\right) \\
&\le \quad \frac{4}{2^\tau}\left(1 + \frac{\delta}{(2^{-2\tau})\Psi^*}\right) \quad &= \quad \frac{4}{2^\tau} + \frac{\delta(2^{\tau+2})}{\Psi^*},
\end{aligned}
$$

where we used the inequalities $\log(1+x) \leq x$, $\log(1+x) \geq x/(x+1)$, and $2^{\tau-2} \leq (2^{\tau-1}) - 1$ for $\tau = 2, 3, \ldots$. Using the same coupling argument for drop-iterations, we have

$$\theta(2^{-\tau}) - \theta(2^{1-\tau}) \leq 2(\varphi(2^{-\tau}) - \varphi(2^{1-\tau})) \leq \frac{8}{2^\tau} + \frac{\delta(2^{\tau+3})}{\Psi^*}. \tag{32}$$

Let $\epsilon \in (0, 1/2)$ and $\tilde{\tau}$ be an integer greater than one such that $2^{-\tilde{\tau}} \leq \epsilon \leq 2^{1-\tilde{\tau}}$. Then, we have

$$
\begin{aligned}
\theta(\epsilon) &\leq& \theta(2^{-\tilde{\tau}}), \\
&=& \theta(1/2) + \sum_{\tau=2}^{\tilde{\tau}} \left(\theta(2^{-\tau}) - \theta(2^{1-\tau}) \right), \\
&\leq& \theta(1/2) + \sum_{\tau=2}^{\tilde{\tau}} \left(\frac{8}{2^\tau} + \frac{\delta(2^{\tau+3})}{\Psi^*} \right), \\
&=& \theta(1/2) + \sum_{\tau=0}^{\tilde{\tau}-2} \left(\frac{2}{2^\tau} + \frac{\delta(2^{\tau+5})}{\Psi^*} \right), \\
&\leq& \theta(1/2) + 4 + 32 \frac{\delta(2^{\tilde{\tau}-1})}{\Psi^*}, \\
&\leq& 6 + 10 \left(\frac{\delta}{\Psi^*} \right) \log \left(\frac{\delta}{\Psi^*} \right) + 32 \frac{\delta}{\epsilon \Psi^*},
\end{aligned}
$$

which establishes (29) for $\epsilon \in (0, 1/2)$. $\quad\boxdot$

Next, we establish the overall complexity of Algorithm 1.

Theorem 3.2 *Given $\epsilon \in (0, 1)$, Algorithm 1 computes a $(1-\epsilon)$-approximate solution to the support vector classification problem in*

$$O \left(\frac{(m+r)n\delta}{\Psi^*} \left[\log \left(\frac{\delta}{\Psi^*} \right) + \frac{1}{\epsilon} \right] \right)$$

arithmetic operations.

Proof. The computation of the initial feasible solution (u^0, v^0) requires two furthest point computations, which can be performed in $O((m+r)n)$ operations. At each iteration, the dominating work is the computation of the indices i', j', i'', and j'', each of which requires the optimization of a linear function over the input points and can also be performed in $O((m+r)n)$ operations. The assertion now follows from Theorem 3.1. $\quad\boxdot$

Finally, we establish a core set result.

Theorem 3.3 *Given $\epsilon \in (0,1)$, the subset $\mathcal{X} \subseteq \mathcal{P} \cup \mathcal{Q}$ returned by Algorithm 1 is an ϵ-core set for the support vector classification problem such that*

$$|\mathcal{X}| = O\left(\frac{\delta}{\Psi^*}\left(\log\left(\frac{\delta}{\Psi^*}\right) + \frac{1}{\epsilon}\right)\right). \tag{33}$$

Proof. Let k^* denote the index of the final iterate computed by Algorithm 1. It is easy to verify that the restriction of (u^{k^*}, v^{k^*}) to its positive entries is a feasible solution of the dual formulation of the support vector classification problem for the input sets $(\mathcal{P} \cap \mathcal{X}, \mathcal{Q} \cap \mathcal{X})$. Let μ_* denote the maximum margin between $\text{conv}(\mathcal{P} \cap \mathcal{X})$ and $\text{conv}(\mathcal{Q} \cap \mathcal{X})$. Therefore, $\|w^{k^*}\| \geq \mu_*$. Similarly, let μ^* denote the maximum margin between $\text{conv}(\mathcal{P})$ and $\text{conv}(\mathcal{Q})$. By (7),

$$(1 - \epsilon^{k^*})\mu_* \leq (1 - \epsilon^{k^*})\|w^{k^*}\| \leq \mu^* \leq \mu_* \leq \|w^{k^*}\|.$$

Since $\epsilon^{k^*} \leq \epsilon$, we obtain

$$(1 - \epsilon)\mu_* \leq \mu^* \leq \mu_*.$$

Note that (u^0, v^0) has only two positive components. Each iteration can increase the number of positive components in (u^k, v^k) by at most two. The relation (33) follows from Theorem 3.1. □

3.3. Linear Convergence

In this section, we establish that Algorithm 1 exhibits linear convergence. As mentioned in Section 3.1, Algorithm 1 is the adaptation of the Frank-Wolfe algorithm [9] with Wolfe's away steps [30] to the support vector classification problem. For the general problem of minimizing a convex function over a polytope, Wolfe [30] and Guélat and Marcotte [13] established the linear convergence of this algorithm under the assumptions of Lipschitz continuity and strong convexity of the objective function and strict complementarity. Recently, Ahipaşaoğlu, Sun, and Todd [2] studied this algorithm for the more special problem of minimizing a convex function over the unit simplex and proved linear convergence under a slightly different set of assumptions. None of these previous results is applicable to Algorithm 1 since the dual problem ($\mathbb{D}$) does not have a unique optimal solution in general, which is a necessary consequence of the assumptions made in all previous studies.

Therefore, in order to establish the linear convergence of Algorithm 1, we employ a different technique which was first suggested in [2] and recently used in [31] to exhibit the

linear convergence of a similar algorithm for the minimum enclosing ball problem. The main idea is based on the argument that each iterate (u^k, v^k) generated by Algorithm 1 is an optimal solution of a slight perturbation of the primal problem $(\mathbb{P})$. It follows from the general stability results of Robinson [23] that the distance between (u^k, v^k) and the set of optimal solutions of the dual problem $(\mathbb{D})$ can then be uniformly bounded above for all sufficiently large k.

Let us consider the following perturbation of the primal problem $(\mathbb{P})$:

$$(\mathbb{P}(\tilde{u}, \tilde{v}, \tilde{\epsilon})) \quad \max_{w,\alpha,\beta} \quad -\tfrac{1}{2}\|w\|^2 + \alpha - \beta$$
$$\text{s.t.}$$
$$\begin{array}{rcll} (p^i)^T w - \alpha & \geq & b_i(\tilde{u}, \tilde{v}, \tilde{\epsilon}), & i = 1, \ldots, m, \\ -(q^j)^T w + \beta & \geq & c_j(\tilde{u}, \tilde{v}, \tilde{\epsilon}), & j = 1, \ldots, r, \end{array}$$

where $(\tilde{u}, \tilde{v})$ is any feasible solution of $(\mathbb{D})$, $\tilde{\epsilon} \geq 0$, $b(\tilde{u}, \tilde{v}, \tilde{\epsilon}) \in \mathbb{R}^m$ is defined as

$$b_i(\tilde{u}, \tilde{v}, \tilde{\epsilon}) := \left\{ \begin{array}{ll} (p^i)^T \tilde{w} - (P\tilde{u})^T \tilde{w}, & \text{if } \tilde{u}_i > 0, \\ -2\tilde{\epsilon}\|\tilde{w}\|^2, & \text{otherwise}, \end{array} \right.$$

$c(\tilde{u}, \tilde{v}, \tilde{\epsilon}) \in \mathbb{R}^r$ is given by

$$c_j(\tilde{u}, \tilde{v}, \tilde{\epsilon}) := \left\{ \begin{array}{ll} -(q^j)^T \tilde{w} + (Q\tilde{v})^T \tilde{w}, & \text{if } \tilde{v}_j > 0, \\ -2\tilde{\epsilon}\|\tilde{w}\|^2, & \text{otherwise}, \end{array} \right.$$

and

$$\tilde{w} := P\tilde{u} - Q\tilde{v}.$$

Let us now consider the problem $(\mathbb{P}(u^k, v^k, \epsilon^k))$. By (12) and (13), (w^k, α^k, β^k) is a feasible solution, where $w^k := Pu^k - Qv^k$, α^k and β^k are given by (8). It turns out that (w^k, α^k, β^k) is actually an optimal solution of $(\mathbb{P}(u^k, v^k, \epsilon^k))$.

Lemma 3.2 *For each $k = 0, 1, \ldots$, (w^k, α^k, β^k) is an optimal solution of $(\mathbb{P}(u^k, v^k, \epsilon^k))$ and the corresponding optimal value is $\Psi^k = (1/2)\|w^k\|^2$.*

Proof. The feasibility of (w^k, α^k, β^k) follows from the argument preceding the lemma. It is easy to verify that (w^k, α^k, β^k) along with the Lagrange multipliers (u^k, v^k) satisfy the optimality conditions, which are sufficient since $(\mathbb{P}(u^k, v^k, \epsilon^k))$ is a concave maximization problem with linear constraints. The optimal value is given by $-(1/2)\|w^k\|^2 + (\alpha^k - \beta^k) = (1/2)\|w^k\|^2$ by (8) and the definition of w^k. $\qquad\qquad \boxdot$

Next, we show that the sequence of optimization problems given by $(\mathbb{P}(u^k, v^k, \epsilon^k))$ yields smaller perturbations of the primal problem $(\mathbb{P})$ as ϵ^k tends to zero. Clearly, $b_i(u^k, v^k, \epsilon^k) = c_j(u^k, v^k, \epsilon^k) = -2\epsilon^k \|w^k\|^2$ for i and j such that $u_i^k = 0$ or $v_j^k = 0$. Together with (15) and (16), we obtain

$$|b_i(u^k, v^k, \epsilon^k)| \leq 2\epsilon^k \|w^k\|^2, \quad i = 1, \ldots, m; \quad |c_j(u^k, v^k, \epsilon^k)| \leq 2\epsilon^k \|w^k\|^2, \quad j = 1, \ldots, r, \quad (34)$$

which establishes our claim since $\|w^k\|^2 \leq 2\delta$.

It is also useful to note that

$$\sum_{i=1}^{m} (u_i^k) b_i(u^k, v^k, \epsilon^k) = \sum_{i:u_i^k > 0} (u_i^k)[(p^i)^T(w^k) - (Pu^k)^T(w^k)] = 0, \quad (35)$$

where we used the fact that u^k lies on the unit simplex in $\mathbb{R}^m$. Similarly,

$$\sum_{j=1}^{r} (v_j^k) c_j(u^k, v^k, \epsilon^k) = \sum_{j:v_j^k > 0} (v_j^k)[-(q^j)^T(w^k) + (Qv^k)^T(w^k)] = 0. \quad (36)$$

Let $\Theta(b(\tilde{u}, \tilde{v}, \tilde{\epsilon}), c(\tilde{u}, \tilde{v}, \tilde{\epsilon}))$ denote the optimal value of the problem $(\mathbb{P}(\tilde{u}, \tilde{v}, \tilde{\epsilon}))$. It follows that Θ is a concave function of $(b(\tilde{u}, \tilde{v}, \tilde{\epsilon}), c(\tilde{u}, \tilde{v}, \tilde{\epsilon}))$. Furthermore, any Lagrange multiplier (u^*, v^*) corresponding to any optimal solution of the unperturbed problem $(\mathbb{P})$ is a subgradient of Θ at $(0, 0)$. Hence,

$$\begin{aligned}
\Theta(b^k, c^k) = \Psi^k &\leq \Theta(0, 0) + (u^*, v^*)^T(b^k, c^k), \\
&= \Psi^* + [(u^*, v^*) - (u^k, v^k)]^T(b^k, c^k), \quad (37) \\
&\leq \Psi^* + \|(u^*, v^*) - (u^k, v^k)\| \|(b^k, c^k)\|,
\end{aligned}$$

where we used (35), (36), and

$$(b^k, c^k) = (b(u^k, v^k, \epsilon^k), c(u^k, v^k, \epsilon^k)).$$

By (34) and (19),

$$\|(b(u^k, v^k, \epsilon^k), c(u^k, v^k, \epsilon^k))\| \leq 2(m + r)^{1/2} \epsilon^k \|w^k\|^2 \leq 4(m + r)^{1/2} \epsilon^k \delta. \quad (38)$$

Therefore, in order to compute an upper bound on $\Psi^k - \Psi^*$ in (37), it suffices to find an upper bound on $\|(u^*, v^*) - (u^k, v^k)\|$. In order to establish such an upper bound, we rely on the results of Robinson [23] on the stability of optimal solutions of a nonlinear optimization problem under perturbations of the problem. Robinson's results require that the unperturbed

problem ($\mathbb{P}$) satisfy certain assumptions. We simply need to adapt these assumptions to a maximization problem. Since ($\mathbb{P}$) is a concave maximization problem with linear constraints, the constraints are regular at any feasible solution. Let (w^*, α^*, β^*) be an optimal solution of ($\mathbb{P}$) with (u^*, v^*) as any corresponding Lagrange multipliers (i.e., any optimal solution of ($\mathbb{D}$)). Let $\mathcal{L}$ denote the Lagrangian function corresponding to the problem ($\mathbb{P}$) given by

$$\mathcal{L}((w, \alpha, \beta), (u, v)) = -(1/2)\|w\|^2 + (\alpha - \beta) + \sum_{i=1}^{m} u_i((p^i)^T w - \alpha) + \sum_{j=1}^{r} v_j(-(q^j)^T w + \beta).$$

We need to establish that Robinson's second-order constraint qualification is satisfied at (w^*, α^*, β^*). These conditions are driven by the requirement that all feasible directions at (w^*, α^*, β^*) that are orthogonal to the gradient of the objective function should necessarily lead to a feasible point of ($\mathbb{P}$) with a smaller objective function value. In particular, these conditions imply that the optimal solution of ($\mathbb{P}$) is unique since ($\mathbb{P}$) is a concave maximization problem.

To this end, we have

$$\nabla_{(w,\alpha,\beta)}\mathcal{L}((w, \alpha, \beta), (u, v)) = \begin{bmatrix} -w + \sum_{i=1}^{m} u_i p^i - \sum_{j=1}^{r} v_j q^j \\ 1 - \sum_{i=1}^{m} u_i \\ -1 + \sum_{j=1}^{r} v_j \end{bmatrix},$$

and

$$\nabla^2_{(w,\alpha,\beta)}\mathcal{L}((w, \alpha, \beta), (u, v)) = \begin{bmatrix} -I & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

where $I \in \mathbb{R}^{n \times n}$ is the identity matrix.

Let

$$\mathcal{I} = \{i \in \{1, \ldots, m\} : (p^i)^T w^* = \alpha^*\}, \quad \mathcal{J} = \{j \in \{1, \ldots, r\} : (q^j)^T w^* = \beta^*\}.$$

Every feasible direction $d := (h^T, \eta, \nu)^T \in \mathbb{R}^{n+2}$ at (w^*, α^*, β^*) satisfies

$$(p^i)^T h - \eta \geq 0, \quad i \in \mathcal{I}; \quad -(q^j)^T h + \nu \geq 0, \quad j \in \mathcal{J}. \tag{39}$$

For second-order conditions, we are only interested in feasible directions that are orthogonal to the gradient of the objective function of ($\mathbb{P}$) evaluated at (w^*, α^*, β^*), i.e., those directions that satisfy

$$-(w^*)^T h + \eta - \nu = 0. \tag{40}$$

Using the fact that $w^* = Pu^* - Qv^*$, it follows from (40) that

$$-\sum_{i \in \mathcal{I}} u_i^*(p^i)^T h + \sum_{j \in \mathcal{J}} v_j^*(q^j)^T h + \eta - \nu = -\sum_{i \in \mathcal{I}} u_i^* \left((p^i)^T h - \eta\right) + \sum_{j \in \mathcal{J}} v_j^* \left((q^j)^T h - \nu\right) = 0, \quad (41)$$

which, together with $u^* \geq 0$, $v^* \geq 0$, and (39), implies that

$$(p^i)^T h = \eta, \quad i \in \mathcal{I}; \quad (q^j)^T h = \nu, \quad j \in \mathcal{J}, \quad (42)$$

for all feasible directions $d = (h^T, \eta, \nu)^T$ satisfying (40). Since $|\eta| \leq \max_{i \in \mathcal{I}} \|p^i\| \|h\|$ and $|\nu| \leq \max_{j \in \mathcal{J}} \|q^j\| \|h\|$,

$$\|d\|^2 = \|h\|^2 + \eta^2 + \nu^2 \leq \|h\|^2 (1 + \max_{i \in \mathcal{I}} \|p^i\|^2 + \max_{j \in \mathcal{J}} \|q^j\|^2). \quad (43)$$

Therefore,

$$d^T \nabla_{(w,\alpha,\beta)}^2 \mathcal{L}((w,\alpha,\beta),(u,v))d = -\|h\|^2 \leq -\left(\frac{1}{1 + \max_{i \in \mathcal{I}} \|p^i\|^2 + \max_{j \in \mathcal{J}} \|q^j\|^2}\right) \|d\|^2,$$

which establishes that Robinson's second-order sufficient condition holds at (w^*, α^*, β^*) (see Definition 2.1 in [23]). By Theorem 4.2 in [23], there exists a constant $\ell > 0$ and an optimal solution (u^*, v^*) of $(\mathbb{D})$ such that, for all sufficiently small ϵ^k,

$$\|(u^k, v^k) - (u^*, v^*)\| \leq \ell \|(b(u^k, v^k, \epsilon^k), c(u^k, v^k, \epsilon^k))\| \leq 4\ell(m+r)^{1/2}\delta\epsilon^k, \quad (44)$$

where we used (38). Combining this inequality with (37), we obtain

$$\Psi^k - \Psi^* \leq 16\ell(m+r)\delta^2(\epsilon^k)^2, \quad (45)$$

for all sufficiently large k.

Let us now assume that $\epsilon^k \leq 1/2$. By Lemma 3.1, we have

$$\Psi^{k+1} \leq \Psi^k \left(1 - \frac{(\epsilon^k)^2 \Psi^*}{(\epsilon^k)^2 \Psi^* + \delta}\right) = \Psi^k - \frac{\Psi^k(\epsilon^k)^2 \Psi^*}{(\epsilon^k)^2 \Psi^* + \delta} \leq \Psi^k - \frac{(\epsilon^k)^2(\Psi^*)^2}{(1/4)\Psi^* + \delta}$$

at each add- or decrease-iteration. Combining this inequality with (45), we conclude that

$$\Psi^{k+1} - \Psi^* \leq \Psi^k - \Psi^* - \frac{(\epsilon^k)^2(\Psi^*)^2}{(1/4)\Psi^* + \delta} \leq \left(1 - \frac{(\Psi^*)^2}{((1/4)\Psi^* + \delta)16\ell(m+r)\delta^2}\right)(\Psi^k - \Psi^*) \quad (46)$$

for all sufficiently small ϵ^k, which establishes the linear convergence of Algorithm 1.

Theorem 3.4 *Algorithm 1 computes dual feasible solutions (u^k, v^k) with the property that the sequence $\Psi^k - \Psi^*$ is nonincreasing. Asymptotically, this gap reduces at least by the factor given in (46) at each add- or decrease-iteration. There exist data-dependent constants ρ and ϑ such that Algorithm 1 computes a $(1 - \epsilon)$-approximate solution to the support vector classification problem in $\rho + \vartheta \log(1/\epsilon)$ iterations for $\epsilon \in (0, 1)$.*

Proof. Let $\rho := \max\{\rho^*, \theta(1/2)\}$, where ρ^* is the smallest value of k such that the inequality (44) is satisfied. After iteration ρ, the improvement in each add- or decrease-iteration obeys (46). Let k^* denote the index of the final iterate computed by Algorithm 1. By (7), we have $(1 - \epsilon)^2 \Psi^{k^*} \leq (1 - \epsilon^{k^*})^2 \Psi^{k^*} \leq \Psi^* \leq \Psi^{k^*}$, which implies that $\Psi^{k^*} - \Psi^* \leq [1 - (1 - \epsilon)^2]\Psi^{k^*} = \epsilon(2 - \epsilon)\Psi^{k^*}$. Since $\epsilon \in (0, 1)$ and $\Psi^* \leq \Psi^{k^*}$, a sufficient condition for termination is given by $\Psi^{k^*} - \Psi^* \leq \epsilon\Psi^*$. At iteration ρ, $\Psi^\rho - \Psi^* \leq 3\Psi^*$ since $(1/4)\Psi^k \leq \Psi^* \leq \Psi^k$ for all $k \geq \rho$ by (7). Therefore, we simply need to compute an upper bound on the number of iterations to decrease the gap from $3\Psi^*$ to $\epsilon\Psi^*$. The result now follows from (46) and the previous argument that each drop-iteration can be paired with a previous add-iteration with a possible increase of two iterations to account for the initial positive components of (u^0, v^0). $\square$

We remark that the convergence result of Theorem 3.4 does not yield a global bound since it relies on data-dependent parameters such as ρ and ϑ. As such, it does not necessarily lead to a better convergence result than that of Theorem 3.2. The main result is that the asymptotic rate of convergence of Algorithm 1 is linear. However, the actual radius of convergence does depend on the input data.

3.4. Nonlinearly Separable and Inseparable Cases

In Sections 3.1–3.3, we presented and analyzed Algorithm 1 for the linearly separable case, which uses the linear kernel function $\kappa(x, y) = x^T y$. We have chosen to illustrate and analyze the algorithm on such input sets for simplicity. We now discuss how to extend Algorithm 1 to the nonlinearly separable and inseparable cases without sacrificing the complexity bound, the core set size, and the linear convergence.

First, let us assume that the input sets are nonlinearly separable. Let $\Phi : \mathbb{R}^n \to \mathcal{S}$ denote the transformation of the given input points into the feature space $\mathcal{S}$ and let $\kappa : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ denote the kernel function given by $\kappa(x, y) = \langle \Phi(x), \Phi(y) \rangle$. As described in Section 2.2, we just need to call Algorithm 1 with the new input sets $\mathcal{P}' := \{\Phi(p^1), \ldots, \Phi(p^m)\}$ and

$\mathcal{Q}' := \{\Phi(q^1), \ldots, \Phi(q^r)\}$ in $\mathcal{S}$. However, since the transformation Φ may not be efficiently computable, Algorithm 1 needs to be modified so that explicit evaluations of the function Φ are avoided.

The computation of the initial dual feasible solution (u^0, v^0) requires two furthest point computations. Since

$$\langle \Phi(x) - \Phi(y), \Phi(x) - \Phi(y) \rangle = \kappa(x, x) - 2\kappa(x, y) + \kappa(y, y),$$

each distance computation in Algorithm 1 requires three kernel function evaluations. Therefore, the initial solution (u^0, v^0) can be computed in $O(m + r)$ kernel function evaluations.

In contrast with the linear kernel function, we can no longer explicitly compute and store $w^k \in \mathcal{S}$. However, at iteration k, we have

$$w^k = \sum_{i=1}^{m} u_i^k \Phi(p^i) - \sum_{j=1}^{r} v_j^k \Phi(q^j),$$

which implies that

$$\langle w^k, \Phi(x) \rangle = \sum_{i=1}^{m} u_i^k \kappa(p^i, x) - \sum_{j=1}^{r} v_j^k \kappa(q^j, x). \tag{47}$$

Therefore, the computation of the indices i', j', i'', and j'' at each iteration can be performed using kernel functions only. We remark that the number of kernel function evaluations can be significantly reduced since $w^{k+1} = (1 - \lambda^k)w^k + \lambda^k z^k$ at an add-iteration and $w^{k+1} = (1 + \lambda^k)w^k - \lambda^k y^k$ at a decrease- or drop-iteration. At an add-iteration,

$$\langle w^{k+1}, \Phi(x) \rangle = (1 - \lambda^k)\langle w^k, \Phi(x) \rangle + \lambda^k \kappa(p^{i'}, x) - \lambda^k \kappa(q^{j'}, x), \tag{48}$$

where we used $z^k = \Phi(p^{i'}) - \Phi(q^{j'})$, which implies that $\langle w^{k+1}, \Phi(x) \rangle$ can be updated using only a constant number of kernel function evaluations if $\langle w^k, \Phi(x) \rangle$ is stored for each $x \in \mathcal{P} \cup \mathcal{Q}$. A similar derivation at a decrease- or drop-iteration yields

$$\langle w^{k+1}, \Phi(x) \rangle = (1 + \lambda^k)\langle w^k, \Phi(x) \rangle - \lambda^k \kappa(p^{i''}, x) + \lambda^k \kappa(q^{j''}, x), \tag{49}$$

where we used $y^k = \Phi(p^{i''}) - \Phi(q^{j''})$. Since (u^0, v^0) contains only two positive components, $\langle w^k, \Phi(x) \rangle$ can be computed using a constant number of kernel function evaluations by (47), (48), and (49) for each iteration k.

Similarly, at an add-iteration, we have

$$\begin{aligned}
\langle w^{k+1}, w^{k+1} \rangle &= (1 - \lambda^k)^2 \langle w^k, w^k \rangle - 2\lambda^k(1 - \lambda^k)\left(\langle w^k, \Phi(p^{i'}) \rangle - \langle w^k, \Phi(q^{j'}) \rangle\right) \\
&\quad + (\lambda^k)^2 \left(\kappa(p^{i'}, p^{i'}) - 2\kappa(p^{i'}, q^{j'}) + \kappa(q^{j'}, q^{j'})\right),
\end{aligned}$$

and at a decrease- or drop-iteration,

$$
\begin{aligned}
\langle w^{k+1}, w^{k+1} \rangle \;=\;& (1+\lambda^k)^2 \langle w^k, w^k \rangle - 2\lambda^k(1+\lambda^k)\left(\langle w^k, \Phi(p^{i''}) \rangle - \langle w^k, \Phi(q^{j''}) \rangle \right) \\
& + (\lambda^k)^2 \left(\kappa(p^{i''}, p^{i''}) - 2\kappa(p^{i''}, q^{j''}) + \kappa(q^{j''}, q^{j''}) \right).
\end{aligned}
$$

Using a similar argument, since $\langle w^0, w^0 \rangle$ can be computed using a constant number of kernel function evaluations, $\langle w^k, w^k \rangle$ can be updated similarly by using a constant number of calls to the kernel function.

It is easy to verify that all the other computations in Algorithm 1 can be performed using only kernel functions and that each one can be performed using a constant number of kernel function evaluations. It follows then that each iteration requires $O(m + r)$ kernel function evaluations. Note that each iteration can be performed even faster if $\kappa(x, y)$ is computed a priori and stored in memory for each $x, y \in \mathcal{P} \cup \mathcal{Q}$. The analysis of the iteration complexity of Algorithm 1 is entirely independent of the structure of the input set. Therefore, Algorithm 1 maintains the same iteration complexity for nonlinearly separable input sets. Since the support vector classification problem for inseparable data sets can be reformulated as a separable instance as explained in Section 2.3, the same conclusion also holds for inseparable input sets.

Therefore, Algorithm 1 can easily be extended to compute an approximate solution for the support vector classification problem for nonlinearly separable and inseparable data sets. The iteration complexity and the core set results remain unchanged. The number of kernel function evaluations is $O(m + r)$ at each iteration. The number of overall arithmetic operations clearly depends on the complexity of the underlying kernel function. For the linearly separable case, each iteration requires $O((m + r)n)$ operations since each kernel function evaluation requires $O(n)$ operations.

We conclude this section by discussing the extent to which the linear convergence result continues to hold for nonlinearly separable and inseparable data sets. Note that the linear convergence result heavily relies on the stability results of Robinson [23] for nonlinear optimization problems in finite-dimensional space. As such, it immediately follows that Algorithm 1 continues to exhibit linear convergence as long as the feature space $\mathcal{S}$ is finite-dimensional, in which case the corresponding problem ($\mathbb{P}$) is still a finite-dimensional optimization problem. On the other hand, if the feature space is infinite-dimensional, then Robinson's results would not be applicable, which implies that Algorithm 1 may not retain linear convergence for such input sets.

4. Concluding Remarks

In this paper, we developed and analyzed a simple, first-order algorithm for the support vector classification problem. Our algorithm explicitly computes a core set, whose size is independent of the number of points and the dimension. Furthermore, the computational complexity of our algorithm is linear in the number of data points and also linear in $1/\epsilon$. This implies that the algorithm is well-suited for large-scale instances of the support vector classification problem for which a moderate accuracy would be sufficient. Finally, in contrast to previous algorithms that similarly compute a small core set, our algorithm exhibits linear convergence.

This work raises many interesting open problems. For instance, the investigation of core set results for other formulations of the support vector classification problem with linear penalizations of classification errors is an interesting research problem. An efficient implementation of the proposed algorithm would require a careful study. We intend to work on these problems in the near future.

5. Acknowledgments

References

[1] P. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximations via core-sets. In J. E. Goodman, J. Pach, and E. Welzl, editors, *Combinatorial and Computational Geometry*, volume 52 of *Mathematical Sciences Research Institute Publications*. Mathematical Sciences Research Institute Publications, 2005.

[2] D. Ahipaşaoğlu, P. Sun, and M. J. Todd. Linear convergence of a modified Frank-Wolfe algorithm for computing minimum-volume enclosing ellipsoids. *Optimization Methods and Software*, 23(1):5–19, 2008.

[3] M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *Proceedings of the 14th Annual Symposium on Discrete Algorithms*, pages 801–802, 2003.

[4] K. P. Bennett and E. J. Bredensteiner. Duality and geometry in SVM classifiers. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 57–64, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[5] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of 34th Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002.

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

[7] K. L. Clarkson. Coresets, sparse greedy approximation and the Frank-Wolfe algorithm. In *Proceedings of the 19th Annual Symposium on Discrete Algorithms*, 2008. To appear.

[8] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, New York, NY, USA, 2000.

[9] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

[10] T. T. Freiss. Support vector neural networks: The kernel-adatron with bias and soft-margin. Technical report, The University of Sheffield, 1999.

[11] B. Gärtner and M. Jaggi. Core-sets for polytope distance. *25th Annual Symposium on Computational Geometry*, 2009. To appear.

[12] E. G. Gilbert. Minimizing the quadratic form on a convex set. *SIAM Journal on Control*, 4:61–79, 1966.

[13] J. Guélat and P. Marcotte. Some comments on Wolfe's away steps. *Mathematical Programming*, 35:110–119, 1986.

[14] S. Har-Peled, D. Roth, and D. Zimak. Maximum margin coresets for active and noise tolerant learning. In *IJCAI*, pages 836–841, 2007.

[15] T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press, 1999.

[16] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. A fast iterative nearest point algorithm for support vector machine classifier design. *IEEE Transactions on Neural Networks*, 11(1):124–136, 2000.

[17] P. Kumar, J. S. B. Mitchell, and E. A. Yıldırım. Approximate minimum enclosing balls in high dimensions using core-sets. *The ACM Journal of Experimental Algorithmics*, 8(1), 2003.

[18] P. Kumar and E. A. Yıldırım. Minimum volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications*, 126(1):1–21, 2005.

[19] P. Kumar and E. A. Yıldırım. An algorithm and a core set result for the weighted euclidean one-center problem. *INFORMS Journal on Computing*, 2009. To appear.

[20] B. F. Mitchell, V. F. Dem'yanov, and V. N. Malozemov. Finding the point of a polyhedron closest to the origin. *SIAM Journal on Control*, 12:19–26, 1974.

[21] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *Proceedings of the IEEE Workshop on on Neural Networks for Signal Processing*, pages 276–285, New York, 1997. IEEE.

[22] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, 1999.

[23] S. M. Robinson. Generalized equations and their solutions, part ii: Applicat ions to nonlinear programming. *Mathematical Programming Study*, 19:200–221, 1982.

[24] M. J. Todd and E. A. Yıldırım. On Khachiyan's algorithm for the computation of minimum volume enclosing ellipsoids. *Discrete Applied Mathematics*, 155(13):1731–1744, 2007.

[25] I. Tsang, J. Kwok, and P.-M. Cheung. Very large SVM training using core vector machines. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, Jan 6-8, 2005, Savannah Hotel, Barbados*, pages 349–356. Society for Artificial Intelligence and Statistics, 2005. (Available electronically at http://www.gatsby.ucl.ac.uk/aistats/).

[26] I. W. Tsang, A. Kocsor, and J. T. Kwok. Simpler core vector machines with enclosing balls. In *Proceedings of the 24th international conference on Machine learning*, pages 911–918, 2007.

[27] I. W. Tsang, J. T. Kwok, and P. M. Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2005.

[28] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[29] V. N. Vapnik and S. Kotz. *Estimation of Dependences Based on Empirical Data: Empirical Inference Science (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[30] P. Wolfe. Convergence theory in nonlinear programming. In *Integer and Nonlinear Programming*, pages 1–36. North-Holland, Amsterdam, 1970.

[31] E. A. Yıldırım. Two algorithms for the minimum enclosing ball problem. *SIAM Journal on Optimization*, 19(3):1368–1391, 2008.