A Generic Nearest Point Algorithm for Solving Support Vector Machines

(Tentative)

Jinbo Bi
Department of Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, NY 12180
bij2@rpi.edu

Kristin P. Bennett
Department of Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, NY 12180
bennek@rpi.edu

ABSTRACT
We propose a nearest point algorithm (NPA) for optimization of a family of Support Vector Machines (SVM) methods. Geometrically, optimizing SVM corresponds to finding the nearest points between two polyhedrons. In classification, the hard margin case corresponds to finding the closest points of the convex hull of each class. The soft margin case corresponds to finding the closest points in the reduced convex hulls. A prior nearest point algorithm (NPA) was very effective but limited to the convex hull case for classification. We propose a generic NPA applicable to both the regular and reduced convex hull cases. Our approach is applicable to both classification and regression SVM based on 1-norm and 2-norm error functions. The resulting algorithm is efficient, easy to implement and requires no optimization solvers. We review how SVM regression with \( \epsilon \)-insensitive loss can be regarded as a nearest point problem for two reduced convex hulls. Experimental results for NPA for SVM classification problems indicate the method is extremely promising. We discuss extensions of the approach applicable to very large databases.

Keywords
support vector machines, convex hull, nearest point algorithms,

1. INTRODUCTION
The intuitive geometric interpretation of support vector classification has been developed [1]. Finding the maximum margin between the two classes is equivalent to finding the closest points in the convex hulls containing each class in separable cases, or finding the closest points in the reduced convex hulls containing each class in inseparable cases. The similar interpretation can be achieved for support vector regression by converting regression problems into classification problems where the training data have been shifted along the response dimension up and down respectively by \( \epsilon \) distance. Generating decision function for the two shifted classes of data is equivalent to finding regression function. Hence the regression problem can also be interpreted as finding the closest points in the convex hulls of the shifted classes of data in constructing hard \( \epsilon \) tube or reduced convex hulls in constructing soft \( \epsilon \) tube. Without explicitly point out, in the remaining of this paper, we simply call the support vector classification problem the classification problem.

Keerthi proposed a nearest point algorithm (NPA) to solve the support vector classification for its separable case and inseparable case with quadratic penalty on error points. In this paper, we extend it to solve support vector machines of inseparable case with linear penalty on error points. Based on analyzing the separation of convex hulls and properties of polyhedrons, we present a generic framework of nearest point algorithm which can be applied to each of the three scenarios of support vector classification. The proposed algorithm can be easily adapted to solving support vector regression problem of similarly three cases.

The original NPA can be adopted directly to solve separable classification problem formulated as follows

\[
\min_{\alpha, \alpha^*} \frac{1}{2} \| \sum_{x_i \in C^+} \alpha_i x_i - \sum_{x_j \in C^-} \alpha_j^* x_j \|^2
\]

s.t.
\[
\sum_{i} \alpha_i = 1, \quad \sum_{j} \alpha_j^* = 1, \\
\alpha_i \geq 0, \quad \alpha_j^* \geq 0
\]  

(1)

By performing a simple transformation, inseparable classification with quadratic penalty can be converted into an instance of separable classification. Properly adjusting kernel function can easily absorb the transformed term by setting

\[
\hat{k}(x, z) = k(x, z) + \frac{1}{C} \delta_{xz}.
\]

where \( \delta_{xz} = 1 \) if \( x = z \) and zero otherwise and \( C \) is the regularization parameter. So the exactly same NPA for separable classification formulation Eq.(1) can solve this problem with only difference of calculating a slightly altered kernel function.

Using same regularization parameter \( C \) but 1-norm penalty on violations instead of 2-norm, the inseparable classification
with linear penalty can be formulated as follows
\[
\min_{\alpha, \alpha^*} \frac{1}{2} \sum_{x_i \in C^+} \alpha_i x_i - \sum_{x_j \in C^-} \alpha^*_j x_j
\]
subject to
\[
\sum_{i=1}^{n} \alpha_i = 1, \quad \sum_{j=1}^{n} \alpha^*_j = 1,
\]
\[
0 \leq \alpha_i \leq C, \quad 0 \leq \alpha^*_j \leq C.
\]
(2)

This problem is not straightforward to employ the NPA because it involves the reduced convex hulls and the geometry shape of reduced convex hull is not easily acquired although it is still a convex set, more precisely speaking, it is also a polyhedron. We will analyze it in Section 4 after we introduce the basic separation analysis of convex hulls first.

2. REGRESSION AS CLASSIFICATION

By examining when ɛ-tubes exist, we show that SVR can be regarded as a classification problem in the dual space. Hard and soft ɛ-tubes are constructed by separating the convex or reduced convex hulls respectively of the training data with the response variable shifted up and down by ɛ. A novel SVR model is proposed based on choosing the max-margin plane that separates the response variable shifted up and down by ɛ. In the dual space, regression becomes a classification problem in the dual space. Hard and soft ɛ-tubes constructed about the data in the space of the data attributes and the response variable [8] (see e.g. Figure 1(a)).

The primary contribution of this work is the novel geometric interpretation of SVR from the dual perspective with a mathematically rigorous derivation of the geometric concepts. In Section ??, for a fixed ɛ > 0 we examine the question “When does a “perfect” or “hard” ɛ-tube exist?”. With duality analysis, the existence of a hard ɛ-tube depends on the separability of two sets. The two sets consist of the training data augmented with the response variable shifted up and down by ɛ. In the dual space, regression becomes the classification problem of distinguishing between the two shifted datasets. Maximizing the margin corresponds to shrinking the effective ɛ-tube. In the proposed approach the effects of the choices of all parameters become clear geometrically.

We provide a geometric explanation for SVR with the ɛ-insensitive loss function. From the primal perspective, a linear function with no residuals greater than ɛ corresponds to an ɛ-tube constructed about the data in the space of the data attributes and the response variable [8] (see e.g. Figure 1(a)). The primary contribution of this work is the novel geometric interpretation of SVR from the dual perspective along with a mathematically rigorous derivation of the geometric concepts. In Section ??, for a fixed ɛ > 0 we examine the question “When does a “perfect” or “hard” ɛ-tube exist?”.

Duality analysis, the existence of a hard ɛ-tube depends on the separability of two sets. The two sets consist of the training data augmented with the response variable shifted up and down by ɛ. In the dual space, regression becomes the classification problem of distinguishing between the two shifted datasets. Maximizing the margin corresponds to shrinking the effective ɛ-tube. In the proposed approach the effects of the choices of all parameters become clear geometrically.

We provide a geometric explanation for SVR with the ɛ-insensitive loss function. From the primal perspective, a linear function with no residuals greater than ɛ corresponds to an ɛ-tube constructed about the data in the space of the data attributes and the response variable [8] (see e.g. Figure 1(a)). The primary contribution of this work is the novel geometric interpretation of SVR from the dual perspective along with a mathematically rigorous derivation of the geometric concepts. In Section ??, for a fixed ɛ > 0 we examine the question “When does a “perfect” or “hard” ɛ-tube exist?”.

With duality analysis, the existence of a hard ɛ-tube depends on the separability of two sets. The two sets consist of the training data augmented with the response variable shifted up and down by ɛ. In the dual space, regression becomes the classification problem of distinguishing between the two shifted datasets. Maximizing the margin corresponds to shrinking the effective ɛ-tube. In the proposed approach the effects of the choices of all parameters become clear geometrically.

We provide a geometric explanation for SVR with the ɛ-insensitive loss function. From the primal perspective, a linear function with no residuals greater than ɛ corresponds to an ɛ-tube constructed about the data in the space of the data attributes and the response variable [8] (see e.g. Figure 1(a)). The primary contribution of this work is the novel geometric interpretation of SVR from the dual perspective along with a mathematically rigorous derivation of the geometric concepts. In Section ??, for a fixed ɛ > 0 we examine the question “When does a “perfect” or “hard” ɛ-tube exist?”.

With duality analysis, the existence of a hard ɛ-tube depends on the separability of two sets. The two sets consist of the training data augmented with the response variable shifted up and down by ɛ. In the dual space, regression becomes the classification problem of distinguishing between the two shifted datasets. Maximizing the margin corresponds to shrinking the effective ɛ-tube. In the proposed approach the effects of the choices of all parameters become clear geometrically.

We provide a geometric explanation for SVR with the ɛ-insensitive loss function. From the primal perspective, a linear function with no residuals greater than ɛ corresponds to an ɛ-tube constructed about the data in the space of the data attributes and the response variable [8] (see e.g. Figure 1(a)). The primary contribution of this work is the novel geometric interpretation of SVR from the dual perspective along with a mathematically rigorous derivation of the geometric concepts. In Section ??, for a fixed ɛ > 0 we examine the question “When does a “perfect” or “hard” ɛ-tube exist?”.

With duality analysis, the existence of a hard ɛ-tube depends on the separability of two sets. The two sets consist of the training data augmented with the response variable shifted up and down by ɛ. In the dual space, regression becomes the classification problem of distinguishing between the two shifted datasets. Maximizing the margin corresponds to shrinking the effective ɛ-tube. In the proposed approach the effects of the choices of all parameters become clear geometrically.

2.1 The hard ɛ-tube case

We now apply the dual convex hull method to constructing the max-margin plane for our augmented sets ɛ+ and ɛ− assuming they are strictly separable, i.e., ɛ > ɛ0. The problem is illustrated in detail in Figure 2. The closest points of ɛ+ and ɛ− can be found by solving the following dual C-SVR quadratic program:

\[
\min_{\alpha, \alpha^*} \frac{1}{2} \sum_{x_i \in C^+} \alpha_i x_i - \sum_{x_j \in C^-} \alpha^*_j x_j
\]

subject to
\[
\sum_{i=1}^{n} \alpha_i = 1, \quad \sum_{j=1}^{n} \alpha^*_j = 1,
\]
\[
0 \leq \alpha_i \leq C, \quad 0 \leq \alpha^*_j \leq C.
\]

(3)

Let the closest points in the convex hulls of ɛ+ and ɛ− be \( c = (x^+ \hat{u}) \hat{u} \) and \( d = (x^- \hat{v}) \hat{v} \) respectively. The max-margin separating plane bisects these two points. The normal \((\hat{w}, \hat{\delta})\) of the plane is the difference between them, i.e., \( \hat{w} = x^+ \hat{u} - x^- \hat{v} \), \( \hat{\delta} = (y^+ - \omega)^T \hat{u} - (y^- - \omega)^T \hat{v} \). The threshold, \( \hat{b} \), is the distance from the origin to the point halfway between the two closest points along the normal:

\[
\hat{b} = \hat{w} \left( \frac{x^+ u + x^- v}{2} \right) + \hat{\delta} \left( \frac{y^+ v + y^- u}{2} \right).
\]

The separating plane has the equation \( \hat{w} x + \hat{\delta} y = \hat{b} = \hat{0} \). Rescaling this plane yields the regression function.

Dual C-SVR (3) is in the dual space. The corresponding Primal C-SVR is:
Then subtract the second inequality from the first inequality. The resulting dual C-SVR (3) can be derived by taking the Wolfe or Lagrangian dual of primal C-SVR (4) and simplifying.

We prove that the optimal plane from C-SVR bisects the \( \hat{\epsilon} \)-tube. The supporting planes for class \( C^+ \) and class \( C^- \) determine the lower and upper edges of the \( \hat{\epsilon} \)-tube respectively. The support vectors from \( C^+ \) and \( C^- \) correspond to the points along the lower and upper edges of the \( \hat{\epsilon} \)-tube. See Figure 2.

**Theorem 1** (C-SVR constructs \( \hat{\epsilon} \)-tube). Let the max-margin plane obtained by C-SVR (3) be \( \hat{w} \hat{x} + \delta \hat{y} - \hat{b} = 0 \) where \( \hat{w} = X\hat{u} - X\hat{v}, \delta = (\hat{y} + \epsilon)\hat{u} - (\hat{y} - \epsilon)\hat{v}, \) and \( \hat{b} = \hat{w}'\left(\frac{X'\alpha + X'\beta}{2}\right) + \delta \left(\frac{X'\alpha + X'\beta}{2}\right). \) If \( \epsilon > \epsilon_0 \), then the plane \( w'x + b \) corresponds to an \( \hat{\epsilon} \)-tube of training data \((X_i, y_i), i = 1, 2, \ldots, m\) where \( w = -\frac{\hat{w}'}{\sqrt{2}}, b = \frac{\hat{b}}{\sqrt{2}} \) and \( \hat{\epsilon} = \epsilon - \frac{\epsilon_0}{\sqrt{2}} < \epsilon \).

**Proof.** First, we show \( \hat{\delta} > 0 \). By the Wolfe duality theorem [4], \( \alpha - \beta > 0 \), since the objective values of (4) and the negative objective value of (3) are equal at optimality. By complementarity, the closest points are right on the margin planes \( \hat{w}'X + \delta y - \hat{\alpha} = 0 \) and \( \hat{w}'X + \delta y - \hat{\beta} = 0 \) respectively, so \( \alpha = \hat{w}X\hat{u} + \delta (y + \epsilon)\hat{u} - (y - \epsilon)\hat{v} \) and \( \beta = \hat{w}X\hat{u} + \delta (y - \epsilon)\hat{u} - (y + \epsilon)\hat{v} \). Hence \( \hat{b} = \hat{w}'\left(\frac{X'\alpha + X'\beta}{2}\right) + \delta \left(\frac{X'\alpha + X'\beta}{2}\right) \). If \( \epsilon > \epsilon_0 \), then subtract the second inequality from the first inequality: \( 2\epsilon - \hat{\alpha} + \hat{\beta} > 0 \), that is, \( \hat{\delta} > \frac{2\epsilon}{2\epsilon_0} > 0 \) because \( \epsilon > \epsilon_0 \). Rescale constraints by \( \hat{\delta} < 0 \), and reverse the signs. Let \( \alpha' = -\frac{\hat{w}'}{\sqrt{2}}, \beta' = \frac{\hat{w}'}{\sqrt{2}} \), then the inequalities become \( \hat{x} \hat{w}' - y \leq \epsilon \) and \( \hat{w}'X + \delta (y + \epsilon)\hat{u} - (y - \epsilon)\hat{v} \). Let \( \hat{b} = \frac{\hat{b}}{\sqrt{2}} \). Then \( \hat{\delta} = \frac{\hat{b} + \frac{2\epsilon - \hat{\alpha}}{2\epsilon}}{2\epsilon} \). Substituting into the previous inequalities yields \( \hat{x} \hat{w}' - y \leq \epsilon \). Hence plane \( y = \hat{w}'x + b \) is in the middle of the \( \hat{\epsilon} < \epsilon \) tube.

**3. Separation of Convex Hulls**

A fundamental separation theorem of convex sets is that let \( S \) be a closed convex set in \( \mathbb{R}^n \), a point \( p \notin S \), then \( x \) is the closest point to \( p \) in \( S \) if and only if \( (p - x)'(x - x) \leq 0 \), \( \forall x \in S \). In our classification problem, we have two convex sets, and we can regard it as one of convex sets in the theorem is degenerated to be only one point.

Let us look at our formulations. Realize that if take \( C = 1 \) in formulation (2), it turns to be equivalent to formulation (1) because if the summation of non-negative \( \alpha \)'s is equal to 1, each of \( \alpha \)'s can not be greater than 1. Hence whether or not the convex hull is reduced and the extent to which it is reduced depend on parameter \( C \). In order to make life easier, we only consider formulation (2). Since no matter convex hulls or reduced convex hulls, they are all convex sets, we distinguish them only by different \( C \) values. Define \( U = \{\sum_{\alpha_i \in C^+} \alpha_i x_i | \sum \alpha_i = 1, 0 < \alpha_i \leq C\} \) and \( V = \{\sum_{\alpha_i \in C^-} \alpha_j x_j | \sum \alpha_j = 1, 0 < \alpha_j \leq C\} \) as the convex sets respectively for the class \( C^+ \) labelled as +1 and the class \( C^- \) labelled as -1. Then the Eq. (2) becomes

\[
\min_{x \in U, x' \in V} \frac{1}{2} ||x' - x||^2 \\
\text{s.t.} \quad x^+ \in U, \; x^- \in V
\]

Since the function \( \frac{1}{2} ||x' - x||^2 \) is a continuous function on set \( U \cup V \), and \( U \cup V \) is a compact set as we defined, then by applying basic calculus theorem, we know \( \exists u \in U, v \in V \), such that the infimum of the objective function is attained at the two points. So formulation (6) is well-defined. We simply assume \( C \) takes an appropriate value so that the two convex sets (actually polyhedrons) are strictly separable because it is always the case when solving SVM classifications. It implies that \( u \notin V \) and \( v \notin U \). The following theorem holds true. Figure 4 illustrates the separation theorem of two convex sets by a group of points in two dimensional space. Clearly the vectors \( u - v \) and \( v - u \) have obtuse angle for any \( x \in V \), \( x \neq v \), which indicates \( (u - v)'(v - x) \leq 0 \), \( \forall x \in V \). Worthy of mention is that forming an obtuse angle is not always the case. Consider the other set \( U \), there are two points \( x_2 \) and \( x_3 \) make the \( (v - u)'(v - x) = 0 \) because the vectors \( x_2 - u \) and \( x_3 - u \) are perpendicular to the vector \( v - u \). Therefore based on the following theorem, the closest point, if it is not the extreme point of the convex set, should be a convex combination of this kind of points, which means the set of support vectors consists of these points.

**Theorem 2** (Separation of \( U \) and \( V \)). Let \( U \) and \( V \) defined as Eq.(5). \((u, v)\) is the solution of problem (6) if and only if \( \min_{x \in U} w'x = \max_{x \in V} w'x \) AND \( \max_{x \in U} w'x = \min_{x \in V} w'x \) where \( w = u - v \).

**Proof.** Without loss of generality, we only prove one of them. We use the following definition of separation of convex sets. Let \( U \) and \( V \) be nonempty convex sets. A plane \( H = \{x: w'x = \alpha\} \) is said to separate \( U \) and \( V \) if \( w'x \geq \alpha, \forall x \in U \) and \( w'x \leq \alpha, \forall x \in V \). \( H \) is said to strictly separate \( U \) and \( V \) if \( w'x \geq \alpha + \Delta \) for \( x \in U \) and \( w'x \leq \alpha - \Delta \) for each \( x \in V \) where \( \Delta > 0 \).
Figure 2: The numbered data points are training data, \( C^+ \) consists of the points from 1 to 5, and \( C^- \) consists of the points from 6 to 9. The closest points (circled points) are \( u \) in \( U \) and \( v \) in \( V \) respectively.

A convex hull generated by a finite number of points in \( \mathbb{R}^n \) is actually a polyhedron not a general convex set. For example, a sphere is a convex set but not a polyhedron, and a polyhedron can be expressed by linear constraints. A linear optimization problem with linear cost function over a polyhedron such as \( \min \{ \mathbf{w}^T \mathbf{x} \mid \mathbf{x} \in \mathcal{U} \} \) where \( \mathbf{w} \) is fixed is just a linear programming problem. It guarantees that if a linear programming problem has feasible solutions, it must have basic feasible solutions which are the extreme points of the polyhedron. In another words, the minimum value can be attained at certain extreme points. We are not obviously aware of what should be the extreme points of the \( U \) and \( V \), but when \( C = 1 \), we know the extreme point sets of the convex hulls \( E_U \subseteq C^+ \) and \( E_V \subseteq C^- \). So the following corollary can be easily derived. We describe it without proof.

**COROLLARY 3** (Separation of \( U \) and \( V \) with \( C = 1 \)). Let \( U \) and \( V \) defined as Eq. (5), \( (u, v) \) is the solution of problem (6) if and only if \( \min \{ \mathbf{w}^T \mathbf{x} \mid \mathbf{x} \in \mathcal{U} \} \geq \mathbf{w}^T u \) AND \( \max \{ \mathbf{w}^T \mathbf{x} \mid \mathbf{x} \in \mathcal{C}^- \} \leq \mathbf{w}^T v \) where \( \mathbf{w} = u - v \).

Notice that \( C^+ \) or \( C^- \) is a finite set. You can simply loop on each point in the set to check the optimality. It’s not necessary to solve the linear programming problem in the Theorem 2.

**4. OPTIMALITY CONDITIONS**

Now we have criterion (Corollary 3) to evaluate the optimality and it is pretty simple. Unfortunately it’s not ready to be applied into our NPA as the stopping criterion. As a class of kernel learning machines, SVMs use kernel function \( k(\cdot, \cdot) \), which implicitly maps an input vector \( \mathbf{x} \) to \( \Phi(\mathbf{x}) \) in an usually higher dimensional feature space where \( \Phi(\cdot) \) is a function vector of \( d \) dimensions such that \( k(\mathbf{x}, \mathbf{z}) = \Phi(\mathbf{x})^T \Phi(\mathbf{z}) \), and \( d \) could be infinite. This allows to enhance the capability of a linear learning machine to learn an actually non-linear model. The problem of finding closest points of two convex hulls in feature space induced by kernel function becomes

\[
\min_{\alpha, \alpha^*} \frac{1}{2} \sum_{i \in \mathcal{C}^+} \alpha_i \Phi(\mathbf{x}_i) - \sum_{j \in \mathcal{C}^-} \alpha_j^* \Phi(\mathbf{x}_j) \|\|^2
\]

\[
s.t. \quad \sum_{i \in \mathcal{C}^+} \alpha_i = 1, \quad \sum_{j \in \mathcal{C}^-} \alpha_j^* = 1, \quad 0 \leq \alpha_i \leq C, \quad 0 \leq \alpha_j^* \leq C
\]

(7)

Therefore instead of working on the original input points \( \mathbf{x}_i \)'s \( i \in \mathbb{R}^n \), we have to work with the feature space, looking at the vectors \( \Phi(\mathbf{x}_i) \)'s, and the convex hull of \( \Phi(\mathbf{x}_i) \)'s. Moreover, most cases of classification in practical data mining applications are not linearly separable, even not separable in feature space after kernel mapping. Then the convex hulls have to be reduced in an appropriate way so that the reduced convex hulls are (strictly) separable.

In reduced convex hull cases, however, Corollary 3 can not be directly applied because it is hard to determine the extreme points of reduced convex hulls. When \( C < 1 \), the extreme points of \( U \), for instance, are not necessarily included in \( \mathcal{C}^+ \). As we know, \( \mathcal{C} \) controls the robustness of the SVMs. If \( C = 1 \), it implies that the closest points \( u \) and \( v \) each depend on at least \( M \) training points. So let us assume \( C \) takes values of form \( \frac{1}{M} \) where \( M \) is a positive integer. Obviously, \( 1 \leq M \leq \min\{ |\mathcal{C}^+|, |\mathcal{C}^-| \} \). If \( M \) is too large, problem (6) is not feasible; if \( M \) is so small that \( C > 1 \), it doesn’t restrict more, just same as \( C = 1 \). We find that the extreme point of \( U \), for example, \( \mathbf{x} = \sum \alpha_i \mathbf{x}_i \), must have all \( \alpha_i \)'s either equal 0 or \( \frac{1}{\gamma_i} \). The definition of extreme points of convex set we use in this paper is that a point in the convex set \( \mathcal{S} \) is an extreme point if and only if it can NOT be expressed as a middle point of other two points in \( \mathcal{S} \).

**THEOREM 4** (Vertices of reduced convex set). Let \( \mathcal{S} \) be a reduced convex set defined as

\[
\mathcal{S} = \left\{ \sum_{i=1}^{L} \alpha_i \mathbf{x}_i \mid \sum_{i=1}^{L} \alpha_i = 1, \quad 0 \leq \alpha_i \leq \frac{1}{M} \right\}
\]

If \( \mathbf{x} = \sum_{i=1}^{L} \gamma_i \mathbf{x}_i \) is an extreme point of \( \mathcal{S} \), then \( \gamma_i = 0 \), or \( \frac{1}{\gamma_i} \), \( i = 1, \cdots, L \).

**PROOF.** We use proof by contradiction. Suppose not all \( \gamma_i \)'s be either 0 or \( \frac{1}{\gamma_i} \), i.e., \( \exists \gamma_i \) such that \( 0 < \gamma_i < \frac{1}{\gamma_i} \), we show that there has to exist another \( \gamma_i \neq \gamma_i \), and \( 0 < \gamma_i < \frac{1}{\gamma_i} \). Otherwise, all coefficients \( \gamma_i = 0 \), or \( \frac{1}{\gamma_i} \), \( \forall i \neq t \). Let the number of coefficients being \( \frac{1}{\gamma_i} \) is \( \Gamma \).

Case 1: if \( \Gamma \geq M \), then \( \sum_{i \neq t} \gamma_i + \gamma_i = \sum_{i \neq t} \gamma_i + \gamma_i + 1 \), \( \Gamma > 1 \); Case 2: if \( \Gamma < M - 1 \), then \( \sum_{i \neq t} \gamma_i + \gamma_i \leq \sum_{i \neq t} \gamma_i + \gamma_i + 1 \leq (M - 1) \frac{1}{\gamma_i} + \gamma_i < 1 \).

This is a contradiction with the condition of \( \sum_{i=1}^{L} \gamma_i = 1 \).
Hence \( \exists \gamma_s, \gamma_l \), such that \( \gamma_s \neq \gamma_l, \ 0 < \gamma_l, \ \gamma_s < \frac{1}{2} \). Let \( \Delta > 0 \) is small enough such that \( 0 \leq \gamma_1 \leq \frac{1}{2} \) where \( \gamma_1 = \frac{1}{2} - \Delta, \ \gamma_2 = \frac{1}{2}, \ \gamma_3 = \frac{1}{2} + \Delta \), and \( \gamma_4 = \frac{1}{2} - \Delta, \ \gamma_5 = \frac{1}{2} \). Then the points \( x_s = \sum_{i \in S} \gamma_i x_i + \frac{1}{2} x_s \in S, \) and \( x_t = \sum_{i \notin S} \gamma_i x_i + \frac{1}{2} x_t \in S \). The \( x = \frac{1}{2} x_s + \frac{1}{2} x_t \), which means \( x \) is not an extreme point of \( S \), it is a contradiction. So all \( \gamma_i \)'s are either 0 or \( \frac{1}{2} \).

Denote the set containing all convex combinations of \( x_i \)'s with coefficients either 0 or \( \frac{1}{2} \) be \( E_S \). Each point in \( E_S \) will have exactly \( M \) non-zero coefficients of value \( \frac{1}{2} \). From Theorem 4, the set of extreme points of \( S, E_S \subseteq S \). Similar to how we obtain Corollary 3, it is not necessary to solve linear programming problems in Theorem 2 even for reduced convex hull cases, we check the optimality by looping on the points in \( E_S \) that is a finite set. However, the cardinality of \( E_S \) will be \( \binom{2M}{M} = \frac{(2M)!}{M!M!} \) which is going to be a huge number when \( |S| \) and \( M \) are large. Correspondingly, our algorithm will be in exponential order of computation with respect to the number of data points in \( S \). Fortunately, a further analysis shows that we don’t have to evaluate all \( \binom{2M}{M} \) points. For instance, consider \( U \) in Theorem 2, we calculate the \( w'x_i, \forall x_i \in C^+ \), and sort them by increasing order, then pick out the first \( M \) points to form a point in \( U_E \) by assigning these \( M \) points with coefficients \( \frac{1}{M} \). Only this extreme point has to be evaluated.

**Corollary 5** (Separation of \( U \) and \( V \) with \( C = \frac{1}{M} \)). Let \( U \) and \( V \) defined as Eq.(5). \( (u, v) \) is the solution of problem (6) (and \( w = u - v \)) if and only if \( \frac{1}{M} \sum_{i=1}^{M} w'x_i \geq w'u \) where \( w'x_1, \ldots, w'x_M \) are the first \( M \) smallest values of \( w'x_i, \forall x_i \in C^+ \) and \( \frac{1}{M} \sum_{i=1}^{M} w'x_i \leq w'v \) where \( w'x_1, \ldots, w'x_M \) are the first \( M \) largest values of \( w'x_j, \forall x_j \in C^- \).

Notice that Corollary 3 is a special case of the Corollary 5 when \( M = 1 \). So Corollary 5 is a more general result, and it can be applied to either convex hulls or reduced convex hulls. If we use kernel in the formulation of SVMs, it implies that we are evaluating the separation of the two convex hulls or reduced convex hulls of mapped data points \( \Phi(x_i) \) in feature space. All theoretical analysis above can be easily extended to feature space. Let us denote \( \Phi(U), \Phi(V) \) respectively as the convex hulls of \( C^+ \) and \( C^- \) in feature space.

\[
\Phi(U) = \left\{ \sum_{x_i \in C^+} \alpha_i \Phi(x_i) \mid \sum \alpha_i = 1, \ 0 \leq \alpha_i \leq C \right\}
\]

\[
\Phi(V) = \left\{ \sum_{x_j \in C^-} \alpha_j \Phi(x_j) \mid \sum \alpha_j = 1, \ 0 \leq \alpha_j \leq C \right\}
\]

Then similarly \( u \) and \( v \) in previous theorems can be expressed as \( u = \sum_{x_i \in C^+} \gamma_i \Phi(x_i), \ v = \sum_{x_j \in C^-} \gamma_j \Phi(x_j), \) and still \( w = u - v \). Correspondingly, the inner product such as \( w'x_i \) in the theorems should be calculated in feature space.

\[
w'\Phi(x_i) = \left( \sum_j \gamma_j \Phi(x_j) \right)' \Phi(x_i) = \sum_j \gamma_j \Phi(x_j) \Phi(x_i) = \sum_j \gamma_j k(x_j, x_i)
\]

The kernel version of Theorem 2 becomes

**Theorem 6** (Separation of \( \Phi(U) \) and \( \Phi(V) \)). Let \( \Phi(U) \) and \( \Phi(V) \) defined as Eq.(8). \( (u, v) \) is the solution of problem (6) if and only if \( \min \{ w'x \mid x \in \Phi(U) \} \geq w'u \) AND \( \max \{ w'x \mid x \in \Phi(V) \} \leq w'v \) where \( u = \sum_{x_i \in C^+} \gamma_i \Phi(x_i), \ v = \sum_{x_j \in C^-} \gamma_j \Phi(x_j), \) and \( w = u - v \).

One of the significant advantages of kernel learning machines is that the mapped data \( \Phi(x) \) are never explicitly formed, and the mapping is implicitly done when kernel is used to replace inner product. So notice that the calculation of \( w'x_i \) uses kernel expression not \( \Phi(x) \). In the solution \( w = u - v \), we only care about the values of those coefficients \( \gamma_i \) and \( \gamma_j \) because eventually our classification or regression function is \( f(x) = \sum_i \gamma_i k(x_i, x) - \sum_j \gamma_j k(x_j, x) + b \).

The following corollary is the kernel version of Corollary 5. The points \( x_i \)'s in \( C^+ \) are sorted by the value of \( w'\Phi(x_i) \) increasing. We denote them as \( \{ x_{i_1}, x_{i_2}, \ldots \} \). The points in \( C^- \) are sorted by \( w'\Phi(x_j) \) value decreasing, denoted as points \( \{ x_{j_1}, x_{j_2}, \ldots \} \).

**Corollary 7** (Separation of \( \Phi(U) \) and \( \Phi(V) \)). Let \( \Phi(U) \) and \( \Phi(V) \) defined as Eq.(8). \( (\gamma, \gamma^*) \) is the solution of problem (7) (\( \gamma \) and \( \gamma^* \) are coefficient vectors for \( C^+ \) and \( C^- \) respectively) if and only if

\[
\frac{1}{M} \sum_{i=1}^{M} w'\Phi(x_{i_l}) \geq w'u
\]

AND

\[
\frac{1}{M} \sum_{i=1}^{M} w'\Phi(x_{j_l}) \leq w'v
\]

where \( w, u, v, x_{i_l} \) and \( x_{j_l}, \ l = 1, \ldots, M \) are defined as above.

This result is more general than any previous corollaries because it includes any cases where \( C = 1 \) or \( C < 1 \), and if
inner product in $\mathbb{R}^n$ has been taken as kernel, then $f(x,z)$ is exactly same as $x'z$, and then Corollary 7 is equivalent to Corollary 5.

5. THE NPA ALGORITHM

In this section, we will describe the framework of nearest point algorithms. The basic idea is that each iteration the algorithm uses the current $i^{th}$ iterates $(u^i, v^i)$ to go through all training points to check optimality of $(u^i, v^i)$ as we analyzed in the previous section. Once a point is found not satisfying the optimality, then the point will be a clue to updating the current iterates. So a small problem that serves as a subroutine is to find the closest point from a line segment to a point not in the segment. We include the analysis about basic ideas and basic procedure of NPAs in this section.

Keerthi et. al. proposed a nearest point algorithm which is a hybrid approach by combining Gilbert’s algorithm about minimizing quadratic form on convex set [2] and Mitchell-Dem’yanov-Malozenov algorithm [5]. The second algorithm is a way to speed up convergence of NPA. Based on their observation and analysis, they suggested in their paper that there could be other ways to speed up the NPA, such as instead of finding the closest point from a segment to a point, finding the closest point from certain triangles to a point. To keep our generic description of the NPA simple, we wouldn’t include those discussions in this section. Related discussion will be incorporated in Section 7.

Let us first make a real stopping criterion for the NPA. In the implementation of NPA, the optimality is satisfied within a tolerance $\epsilon > 0$ which means Eq.(9) and Eq.(10) become

\[
\frac{1}{M} \sum_{i=1}^{M} w'(x_i) \geq w'u - \epsilon
\]  
\[
\frac{1}{M} \sum_{i=1}^{M} w'(x_i) \leq w'v + \epsilon
\]

If these optimality conditions are not satisfied, for example, there are $M$ points $x_{i1}, \ldots, x_{IM}$ in $C'$ such that Eq.(11) is not met, then the point $\hat{x} = \sum_{i=1}^{M} \frac{1}{M} \Phi(x_{i1})$ is in $\Phi(I)$ (because the summation of coefficients is 1 and each coefficient is no larger than $\frac{1}{M}$), and $w'\hat{x} < w'u - \epsilon$. We then show there must exist a point $u^{new}$ in the segment from $u$ to $\hat{x}$ such that the objective value is strictly dragged down, i.e., $||u^{new} - v||^2 < ||u - v||^2$. A point in a line segment with end points $u$ and $\hat{x}$ can be expressed as $\lambda u + (1 - \lambda)\hat{x}$ where $0 \leq \lambda \leq 1$. Correspondingly, for set $\Phi(Vc)$, we can also update $v$ using similar method. The following theorem is general, suitable to apply to both sets.

Theorem 8 (Closest Point of Line Segment). Let $S$ be a reduced convex hull, $p \notin S$, and $x \in S$. Let $w = p - x$. If $x \in S$ such that $w'x > w'x$, then $\exists u^{new} = \lambda x + (1 - \lambda)\hat{x}$, and $u^{new} \neq x$, such that $||u^{new} - p||^2 < ||x - p||^2$ where

\[
\lambda = \begin{cases} 
0 & \text{if } (x - \hat{x})(p - \hat{x}) \leq 0 \\
\frac{|x - \hat{x}|(p - x)}{|x - \hat{x}|^2} & \text{if } (x - \hat{x})(p - \hat{x}) > 0.
\end{cases}
\]

6. PSEUDO-CODE

By Theorem 8, we know $x^{new} \in S$ since $x^{new}$ is in the segment from $z$ to $\hat{z}$, and $z$, $\hat{z}$ are in the convex set $S$. Hence even consider the reduced convex hull, we still have new iteration point within reduced convex hull. Moreover, with the duality analysis, we know the normal of the maximum margin plane, denoted as $Vc$, is exactly the difference of the closest points $u - v$, and the intercept term $b$ as how we define it in Theorem ?? and ??, is the distance from the origin to the point $u - v$. We can rewrite the objective function as

\[
\min_{\lambda \geq 0} ||\lambda x + (1 - \lambda)\hat{x} - p||^2.
\]

Notice that the objective function is just a univariate function of $\lambda$ since $x, \hat{x}$ and $p$ are fixed here. Rewrite objective function as

\[
g(\lambda) = ||\lambda x + (1 - \lambda)\hat{x} - p||^2 = \lambda^2(||x - \hat{x}||^2) - 2\lambda((x - \hat{x})(p - \hat{x})) + ||p - \hat{x}||^2.
\]

Obviously, $x \neq \hat{x}$, so $||x - \hat{x}||^2 > 0$, then function $g(\lambda)$ is a strictly convex function. The solution $\lambda$ to a strictly convex optimization problem is unique and also $g(\lambda) > g(\lambda)$, if $\lambda \neq \hat{\lambda}$. Therefore all we need to show is $\hat{\lambda} \neq 1$. Solve the problem $\min g(\lambda)$ without restriction to $\lambda$. We obtain $\hat{\lambda} = \frac{|x - \hat{x}|(p - \hat{x})}{||x - \hat{x}||^2}$. Then under constraints, if $\hat{\lambda} \leq 0$, just set $\hat{\lambda} = 0$, and we show $\hat{\lambda} < 1$.

\[
||x - \hat{x}||^2 = (x - \hat{x})'(x - \hat{x}) = (x - \hat{x})'(x - p) + (p - \hat{x}) = (x - \hat{x})'(x - p) + (x - \hat{x})'(p - \hat{x}) > (x - \hat{x})'(p - \hat{x})
\]

by the condition of $w'\hat{x} < w'u$. 

With the duality analysis, we know the normal of the maximum margin plane, if we denote it as $w$, is exactly the difference of the closest points $w = u - v$, and we need to know the intercept term $b$. As in [1], $b$ is the distance from the origin to the point halfway between the two closest points $u$ and $v$ along the normal $w$, so $b = w'\frac{u + v}{2} = \frac{||u||^2 + ||v||^2}{2}$.

**Figure 4:** Find the point $u^{new}$ on segment from $u$ to $\hat{x}$ in $u$ closest to $v$ in $Vc$. 

**Proof.** Let us solve the problem

\[
\min_{0 \leq \lambda \leq 1} ||\lambda x + (1 - \lambda)\hat{x} - p||^2.
\]
halfway between the two closest points \( \mathbf{u} \) and \( \mathbf{v} \) along the normal \( \langle \mathbf{w}, \mathbf{x} \rangle \) (in the feature space if using kernel), so \( b = \left( \langle \mathbf{w}, \mathbf{u} \rangle + \langle \mathbf{w}, \mathbf{v} \rangle \right) / 2 = \frac{1}{2} (\langle \mathbf{u} - \mathbf{v} \rangle (\mathbf{u} + \mathbf{v}) = \frac{||\mathbf{u}||^2 + ||\mathbf{v}||^2}{2} \). The algorithm is described in Algorithm 2 where \( \mathbf{u} = \sum_{i \in C^+} u_i \Phi(z_i), \mathbf{v} = \sum_{i \in C^-} v_i \Phi(z_i) \). The content in parentheses is arguments for each specific function. In order to use one subroutine to assess optimality for both sets, we put a minus sign to Eq 12 so that evaluating optimality for \( \mathbf{v} \) converts to be a minimization problem too.

Calculating kernel entry for each pair of training points is probably computationally expensive, so efforts have to be done to prevent from evaluating optimality conditions by repeatedly computing Eq 12, or computing \( \lambda \) exactly by the formula in Theorem 8. Since assessing optimality using Eq. 11 and 12 is crucial for NPA, so loop on each point to calculate the left hand side of Eq 11 or 12 may not be able to be avoided in each iteration because the current \( \mathbf{u} \) or \( \mathbf{v} \) varies from previous iterations. Therefore caching the information obtained in optimality evaluation step such as caching the values of \( \sum_{i \in C^+} u_i k(z_i, z_i) \), \( j = 1, \ldots, M \) seems wise in order not to repeat same calculation in later step when updating \( \mathbf{u} \) or \( \mathbf{v} \), or other variables. Meanwhile the right hand sides of Eq 11 and 12 can consume much less computation effort by adopting certain updating scheme instead of Eq. ?? by caching the current values of \( \mathbf{u}' \mathbf{u}, \mathbf{u}' \mathbf{v} \) and \( \mathbf{v}' \mathbf{v} \).

Realize that the step of updating variables takes place after optimality evaluation if certain point \( \hat{\mathbf{z}} \) has been captured not satisfying Eq 11 or 12. Let us consider the current working set \( \mathcal{U} \). Then in Theorem 8, \( \lambda \) is computed as

\[
\lambda = \frac{(\mathbf{u} - \hat{\mathbf{z}})'(\mathbf{v} - \hat{\mathbf{z}})}{||\mathbf{v} - \hat{\mathbf{z}}||^2} = \frac{\mathbf{u}' \mathbf{u} - \mathbf{u}' \hat{\mathbf{z}} - \mathbf{v}' \hat{\mathbf{z}} + \hat{\mathbf{z}}' \hat{\mathbf{z}}}{\mathbf{u}' \mathbf{u} - 2 \mathbf{u}' \hat{\mathbf{z}} + \hat{\mathbf{z}}' \hat{\mathbf{z}}}.
\]

(13)

where \( \mathbf{v}' \mathbf{u} \) and \( \mathbf{u}' \mathbf{v} \) are always in cache, and \( \mathbf{z}' \mathbf{u}, \mathbf{v}' \mathbf{z} \) can be easily computed by the cached quantities from optimality evaluation step. So only \( \hat{\mathbf{z}}' \mathbf{z} \) needs to be evaluated in this step. After the \( \lambda \) is obtained, the closest point \( \mathbf{u} \) is updated as \( \mathbf{u} = \lambda \mathbf{u} + (1 - \lambda) \hat{\mathbf{z}} \). We have to remember that we never use the real closest point \( \mathbf{u} \), instead we store the vector of its convex combination coefficients, which is \( \mathbf{u} \). So the updating step is actually carried out by updating \( \mathbf{u} = \lambda \mathbf{u} + (1 - \lambda) \gamma \mathbf{z} \) where \( \gamma \) is the coefficient vector of \( \mathbf{z} \). The values of \( \mathbf{u}' \mathbf{u} \) and \( \mathbf{u}' \mathbf{v} \) need to be updated consistently as follows

\[
\mathbf{u}' \mathbf{u} = \lambda^2 \mathbf{u}' \mathbf{u} + 2 \lambda (1 - \lambda) \mathbf{u}' \hat{\mathbf{z}} + (1 - \lambda)^2 \hat{\mathbf{z}}' \hat{\mathbf{z}}
\]

\[
\mathbf{u}' \mathbf{v} = \lambda \mathbf{u}' \mathbf{v} + (1 - \lambda) \mathbf{v}' \hat{\mathbf{z}}
\]

(14)

where all the values of \( \mathbf{u}' \mathbf{u}, \mathbf{u}' \mathbf{v}, \mathbf{v}' \mathbf{z} \) and \( \hat{\mathbf{z}}' \hat{\mathbf{z}} \) have been stored, so updating \( \mathbf{u}' \mathbf{u} \) and \( \mathbf{u}' \mathbf{v} \) do not consume any further computation of calculating kernel entries.

7. Speed up and scale NPA

Inspired by the heuristic idea of Platt’s SMO [6], we can adopt the following hierarchy of heuristics: (1) NPA starts with iterating through support vectors to detect the \( \hat{\mathbf{z}} \); (2) if all support vectors satisfy the optimality conditions, then NPA starts iterating through the entire training set until optimality is satisfied. The heuristic behind this is that if we start with the situation that all points are support vectors, then after several iterations, some points do not show up any more in expressions of \( \mathbf{u} \) or \( \mathbf{v} \), which implies they are more likely far from the optimal boundary and hence would not

\[\text{Algorithm 1 All Subroutines in NPA.}\]

function Initialize

Set \( u_i = \frac{1}{\gamma}, v_i = \frac{1}{\gamma}, i = 1, \ldots, \ell \)
Compute \( \mathbf{u}' \mathbf{u}, \mathbf{v}' \mathbf{v} \) by Eq ??
end

function Sort-point

Repeat on points \( \mathbf{z}_i \) in the current working set
Compute \( p_i = \sum_{j \in C^+} u_j k(z_i, z_j) \) and \( q_i = \sum_{j \in C^-} v_j k(z_i, z_j) \)
Cache \( p_i \) and \( q_i \)’s
end
Sort points \( \mathbf{z}_i \) with increasing order of \( p_i - q_i \) if the working set is \( C^+ \) or of \( q_i - p_i \) when the working set is \( C^- \)
Let the sorted set is \( \{ \mathbf{z}_{i_1}, \ldots, \mathbf{z}_{i_M} \} \)
end

function Check-optimality

If the working set is \( C^+ \), Compute the left hand side of Eq 11, i.e. \( \sum_{i=1}^M p_i - q_i \) if the working set is \( C^- \), Compute the left hand side of Eq 12, i.e. \( \sum_{i=1}^M q_i - p_i \)
Check Eq 11 or 12
If it is not satisfied,
Store the indices \( i_l, l = 1, \ldots, M \) in an array \( \gamma \) in proper way
end

function Update

Denote the current working vector be \( \mathbf{q} \), which is either \( \mathbf{u} \) or \( \mathbf{v} \), and the other one be \( \mathbf{p} \)
Compute \( \lambda = \frac{\mathbf{q}' \mathbf{p} - \mathbf{q}' \mathbf{z} - \mathbf{p}' \mathbf{z} + \mathbf{z}' \mathbf{z}}{\mathbf{q}' \mathbf{q} - 2 \mathbf{q}' \mathbf{z} + \mathbf{z}' \mathbf{z}} \)
If \( \lambda < 0 \),
Set \( \lambda = 0 \)
Update the current closest point by \( \mathbf{q} = \lambda \mathbf{q} + (1 - \lambda) \hat{\mathbf{z}} \)
Update \( \mathbf{q}' \mathbf{p} = \lambda \mathbf{q}' \mathbf{p} + (1 - \lambda) \mathbf{p}' \hat{\mathbf{z}} \)
end

be support vectors. The difference from SMO is that we do not iterate only through non-bound support vectors, in contrast, we would like to look at the support vectors at bound, because support vectors at bound are intuitively closer to boundary or even on the other side of boundary, and then closer to the closest point in the other set. So bound support vectors would be more likely to violate optimality than non-bound support vectors according to the basis of NPA. Notice that we can not have more than \( M \) support vectors at bound. We expect that the \( \hat{\mathbf{z}} \) is generated by partly non-bound support vectors and partly support vectors at bound.

8. References

Algorithm 2  The Nearest Point Algorithm.

main routine
Initialize $u$, $v$, $M$, $u'u$, $u'v$, $v'v$
repeat
Sort points in $C^+$ by increasing order of $(u - v)'\Phi(z_i)$ values
computed according to Eq ??, return $\hat{z}$
Check the optimality of $\hat{z} \in U (u, v, \hat{z}, u'u - u'v)$
If not optimal, then update $u$, and $u'u$, $u'v$ using $\hat{z}$

Sort points in $C^-$ by increasing order of $(v - u)'\Phi(z_j)$,
return $\hat{z}$
Check the optimality of $\hat{z} \in V (u, v, \hat{z}, vv - uv)$
If not optimal, then update $v$, and $vv$, $uv$ using $\hat{z}$
until $(u, v)$ is optimal for both $U$ and $V$
Compute $b$
Return obtained model
end

algorithm for support vector machine classifier design.


