

OTDK-DOLGOZAT

2023

Budapesti Corvinus Egyetem

CRISS-CROSS ALGORITHM FOR SUPPORT VECTOR MACHINES

CRISS-CROSS ALGORITMUS TÁMASZVEKTOR-GÉPEKRE

Paulovics Péter

Témavezető: Dr. E.-Nagy Marianna

Kézirat lezárása: 2022. április 21.

Contents

1	Introduction	1
2	Statistical classification and the optimal separating hyperplane	2
3	The quadratic optimization problem of the SVM	6
3.1	Primal problem, optimality conditions	6
3.2	Lagrange dual problem and weak duality	8
3.3	The mixed linear complementarity problem	11
3.4	Other specific properties of the problem	14
4	The criss-cross algorithm	15
4.1	Intuitive description of LP algorithm classes	15
4.2	The Klafszky-Terlaky quadratic criss-cross algorithm	16
4.3	The modified criss-cross algorithm for support vector machines	21
4.4	Computational results	28
5	Open problems	32
5.1	Generalizations	32
5.1.1	Soft Margin SVM	32
5.1.2	Nonlinear SVM	33
5.2	Simplification of the modified algorithm	35
6	Summary	35
A	Appendix	I
A.1	Euclidean geometry	I
A.2	Convex optimization	II
A.3	Pivoting	III
A.4	Further numerical examples for the modified algorithm	IV
A.5	Codes	VI

List of Figures

1	Paradigms of the simplex, interior-point and criss-cross algorithms. Source: [19]	16
---	---	----

List of Tables

1	The criss-cross algorithm of Klafszky and Terlaky for quadratic programming	20
2	Modified quadratic criss-cross algorithm	24
3	Description of datasets	28
4	Computational results, $\epsilon = 0$	29
5	Modified quadratic criss-cross algorithm, numerically more stable version .	30
6	Computational results, $\epsilon = 10^{-6}$	31
7	Modified quadratic criss-cross algorithm, simplified version	36

Listings

1	Algorithm for pivoting	VI
2	Criss-cross algorithm of Klafszky and Terlaky (1992)	VII
3	Initialization for the modified criss-cross algorithm – problem (MLCP-SVM)	IX
4	Initialization for the modified criss-cross algorithm – problem (MLCP-SVM-2)	X
5	Modified criss-cross algorithm for SVM	XI

1 Introduction

The primal simplex method, discovered by Dantzig in 1947 was the first algorithm to solve linear programming (LP) problems [15]. The algorithm starts from a primal feasible basis, which is not trivial to find (two-phase simplex method). The birth of the criss-cross algorithm (Zionts, [44]) was motivated by the need to solve a general linear programming problem in one phase. However, Zionts could not prove the finiteness of his algorithm. The first convergent criss-cross algorithm was discovered independently by Terlaky [37] and Wang [40], while Klafszky and Terlaky [24] generalized the algorithm for linearly constrained quadratic programming (QP) problems¹. Today the criss-cross method is a family of algorithms with many variants. For example, Zhang [43] constructed an LP version with new pivoting rules, which was generalized for quadratic problems by Akkeleş, Balogh and Illés [2] and for general linear complementarity problems (LCPs) with sufficient matrices by Csizmadia and Illés [14].

The idea of Support Vector Machines (SVM) can be traced back to the work of Vapnik and Chervonenkis, published in 1964 in Russian [10]. Their framework was concerned with finding an affine hyperplane which separates two sets of observations with the widest margin. Boser, Guyon and Vapnik [6] generalized the model to nonlinear separation boundaries based on results of [1]. The principle of this method is to transform the Euclidean vector space of the observations into a higher dimensional inner product space, and apply the linear SVM there.

Even with a nonlinear boundary, the assumption that the two sets of observations can be separated perfectly holds rarely in real-world applications. Thus the practical use of the SVM was made possible by the seminal paper of Cortes and Vapnik [11], in which they introduced the notion of soft-margin SVM. The idea is that we allow observations to be misclassified, but for every such error a penalizing term diminishes the objective value of the optimization problem. This way the assumption of perfect separability is relaxed and the model tries to find a balance between increasing the width of the margin around the separating boundary and minimizing the number of observations misclassified.

In practice, several popular algorithms are used to solve the optimization problem of the

¹In an unpublished report, Chang [8] discovered a quadratic criss-cross algorithm equivalent to the version of Klafszky and Terlaky. [17]

SVM, for example interior-point methods or stochastic gradient descent (see e.g. [34] or [33, Chapter 10]). To the best of our knowledge, criss-cross type algorithms have not been used to this task. The first part of our paper is concerned with investigating the reason for this: in Section 2, we derive the linearly constrained primal quadratic optimization problem of the linear SVM from the model. Section 3 contains the discussion of the QP from the perspective of mathematical optimization. We present two derivations of the optimality conditions: by noting the convexity of the problem, and applying the Karush-Kuhn-Tucker conditions (Section 3.1); and by formulating the Lagrange dual problem and proving a special version of the weak duality theorem (Section 3.2).

One of the main results of this paper is in Section 3.3, where we show that the optimization problem of the SVM cannot be formulated into a linear complementarity problem (LCP) without considerably increasing its size, thus existing versions of the criss-cross algorithm cannot be applied to the problem in a straightforward manner. Consequently, in Section 4 we present a new algorithm variant which is a generalization of Klafszky and Terlaky’s quadratic criss-cross algorithm in the sense that it works on a mixed linear complementarity problem, which is a generalization of the LCP. Matlab implementations of the classical and our modified algorithms can be found in Appendix A.5, and computational performance of the implementation on real-world datasets are in Section 4.4. Section 5 contains the discussion on some possible paths for future research and Section 6 concludes.

2 Statistical classification and the optimal separating hyperplane

The binary classification problem can be described intuitively by the following example.

Example 2.1 (Bankruptcy prediction).

We would like to predict whether or not firm X will declare bankruptcy within the next quarter by analyzing its public financial indicators. For making our prediction, we collect data from similar companies. Suppose we find that previous year profitability was a perfect predictor of bankruptcy: a firm which was unprofitable for a year would declare bankruptcy within the next quarter, while profitable companies would always remain solvent. Then we could be confident that our prediction regarding firm X based on this observed pattern of past data will be accurate.

Every statistical classification problem is concerned with finding a similar pattern of past data on which prediction of the future could be made. Of course in real-world problems there are no formulae as simple and easily found as in Example 2.1. Thus the need for mathematical modelling of classification problems arises.

The model of binary classification problem is the following. Suppose m observations are given: $\mathbf{a}_j \in \mathcal{J}$, ($j = 1, \dots, m$). Based on their n recorded features, the observation $\mathbf{a}_j$ can be identified with vectors of $\mathbb{R}^n$. Suppose furthermore that a two-element partition $\{\mathcal{J}_1, \mathcal{J}_2\}$ of the set of observations $\mathcal{J}$ is known. The simplest model of the support vector machine is based on a rather crude assumption.

Assumption 2.2 (Linear separability).

Assume that the sets $\mathcal{J}_1$ and $\mathcal{J}_2$ can be strictly separated by an affine hyperplane $\mathcal{H}(\mathbf{u}, \nu) \subseteq \mathbb{R}^n$. That is, $\exists \mathbf{u} \in \mathbb{R}^n \setminus \{\mathbf{0}\}, \nu \in \mathbb{R}$, such that

$$\mathbf{u}^\top \mathbf{a}_j - \nu \geq \varepsilon > 0 \quad \text{if } \mathbf{a}_j \in \mathcal{J}_1 \quad (1a)$$

$$\mathbf{u}^\top \mathbf{a}_j - \nu \leq -\varepsilon < 0 \quad \text{if } \mathbf{a}_j \in \mathcal{J}_2 \quad (1b)$$

Note that separating hyperplanes are not uniquely characterized by the normal vector $\mathbf{u}$ and scalar ν : it is easy to see that if $\mathcal{H}(\mathbf{u}, \nu)$ satisfies inequalities (1), then for any strictly positive scalar $c \in \mathbb{R}_+$ $\mathcal{H}(c\mathbf{u}, c\nu)$ also satisfies (1). This ambiguity motivates the following definition (see e.g. [33, p. 190]):

Definition 2.3 (Canonical form of the hyperplane).

A pair $(\mathbf{u}, \nu)$ satisfying inequalities (1) is called the *canonical form* of the separating hyperplane with respect to $\mathcal{J}$ if it is scaled such that

$$\min_{\mathbf{a}_j \in \mathcal{J}} |\mathbf{u}^\top \mathbf{a}_j - \nu| = 1.$$

Remark 2.4. It is straightforward to prove that every separating hyperplane has a canonical form, and that a canonical form is unique.

We will now show that if a separating hyperplane is given in its canonical form $\mathcal{H}(\mathbf{u}, \nu)$, then the Euclidean distance of the hyperplane and the nearest observation, denoted by δ (see Definitions A.5 and A.6) is inversely proportionate to the Euclidean norm of $\mathbf{u}$.

Lemma 2.5. Suppose a separating hyperplane is given in its canonical form $\mathcal{H}(\mathbf{u}, \nu)$. Then

$$\min_{\mathbf{a}_j \in \mathcal{J}} \{\delta(\mathbf{a}_j, \mathcal{H}(\mathbf{u}, \nu))\} = \frac{1}{\|\mathbf{u}\|}.$$

Proof. Denote by $\mathbf{a}_1 := \arg \min_{\mathbf{a}_j \in \mathcal{J}} \{\delta(\mathbf{a}_j, \mathcal{H}(\mathbf{u}, \nu))\}$ an observation closest to the hyperplane. It can be assumed without loss of generality that $\mathbf{a}_1 \in \mathcal{J}_1$. Then by the definition of canonical form

$$\mathbf{u}^\top \mathbf{a}_1 - \nu = 1. \quad (2)$$

If $\mathbf{y}$ is a point of the hyperplane $\mathcal{H}(\mathbf{u}, \nu)$, that is, $\mathbf{u}^\top \mathbf{y} - \nu = 0$, then the hyperplane can be written as $\mathcal{H}(\mathbf{u}, \nu) = \{\mathbf{x} : \mathbf{u}^\top (\mathbf{x} - \mathbf{y}) = 0\}$ since

$$\mathbf{u}^\top (\mathbf{x} - \mathbf{y}) = \mathbf{u}^\top \mathbf{x} - \mathbf{u}^\top \mathbf{y} = \nu - \nu = 0 \quad \forall \mathbf{x}, \mathbf{y} \in \mathcal{H}(\mathbf{u}, \nu).$$

Of course $\mathcal{H}(\mathbf{u}, \nu)$ is a closed affine subspace, thus by elementary inner product space theory there exists an orthogonal projection of $\mathbf{a}_1$ onto $\mathcal{H}(\mathbf{u}, \nu)$ denoted by $P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1$. The orthogonal projection $P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1$ is characterized by two properties:

1. It is an element of the hyperplane: $P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1 \in \mathcal{H}(\mathbf{u}, \nu)$.
2. The vector $\mathbf{a}_1 - P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1$ is orthogonal to the hyperplane:

$$(\mathbf{a}_1 - P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1)^\top \mathbf{x} = 0 \quad \forall \mathbf{x} \in \mathcal{H}(\mathbf{u}, \nu).$$

Property 1 is just $\mathbf{u}^\top (P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1 - \mathbf{y}) = 0$, while Property 2 can be rewritten as $\mathbf{a}_1 - P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1 = \lambda \mathbf{u}$ for some $\lambda \in \mathbb{R}$ by the definition of the normal vector $\mathbf{u}$. Putting these two together, we obtain

$$\mathbf{u}^\top (\mathbf{a}_1 - \lambda \mathbf{u} - \mathbf{y}) = 0,$$

thus

$$\lambda = \frac{\mathbf{u}^\top (\mathbf{a}_1 - \mathbf{y})}{\mathbf{u}^\top \mathbf{u}}.$$

It is also known that the orthogonal projection has the property

$$\delta(\mathbf{a}_1, \mathcal{H}(\mathbf{u}, \nu)) = \|\mathbf{a}_1 - P_{\mathcal{H}(\mathbf{u}, \nu)} \mathbf{a}_1\| = \|\lambda \mathbf{u}\| = \left\| \frac{\mathbf{u}^\top (\mathbf{a}_1 - \mathbf{y})}{\mathbf{u}^\top \mathbf{u}} \mathbf{u} \right\|$$

by simple substitution. But

$$\mathbf{u}^\top (\mathbf{a}_1 - \mathbf{y}) = \mathbf{u}^\top \mathbf{a}_1 - \mathbf{u}^\top \mathbf{y} = \mathbf{u}^\top \mathbf{a}_1 - \nu = 1$$

by the definition of $\mathbf{y}$ and (2). Thus

$$\delta(\mathbf{a}_1, \mathcal{H}(\mathbf{u}, \nu)) = \left\| \frac{\mathbf{u}^\top (\mathbf{a}_1 - \mathbf{y})}{\mathbf{u}^\top \mathbf{u}} \mathbf{u} \right\| = \left\| \frac{1}{\mathbf{u}^\top \mathbf{u}} \mathbf{u} \right\| = \left\| \frac{1}{\|\mathbf{u}\|} \frac{\mathbf{u}}{\|\mathbf{u}\|} \right\| = \frac{1}{\|\mathbf{u}\|} \left\| \frac{\mathbf{u}}{\|\mathbf{u}\|} \right\| = \frac{1}{\|\mathbf{u}\|},$$

where we used basic properties of the norm induced by the Euclidean inner product. $\square$

If a separating hyperplane is given in its canonical form $\mathcal{H}(\mathbf{u}, \nu)$ (see Definition 2.3), then Assumption 2.2 can be reformulated as

$$\mathbf{u}^\top \mathbf{a}_j - \nu \geq 1, \quad \text{if } \mathbf{a}_j \in \mathcal{J}_1, \quad (3a)$$

$$\mathbf{u}^\top \mathbf{a}_j - \nu \leq -1, \quad \text{if } \mathbf{a}_j \in \mathcal{J}_2. \quad (3b)$$

Let us denote belonging to sets $\mathcal{J}_1, \mathcal{J}_2$ by $d_j = \begin{cases} 1 & \text{if } \mathbf{a}_j \in \mathcal{J}_1 \\ -1 & \text{if } \mathbf{a}_j \in \mathcal{J}_2 \end{cases}$

Then Assumption 2.2 can be written as

$$d_j (\mathbf{u}^\top \mathbf{a}_j - \nu) \geq 1 \quad (j = 1, \dots, m). \quad (4)$$

Now we introduce the following notations:

$$A := \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_m^\top \end{bmatrix}, \quad \mathbf{d} := \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_m \end{bmatrix}, \quad D := \begin{bmatrix} d_1 & 0 & \dots & 0 \\ 0 & d_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & d_m \end{bmatrix}, \quad \mathbf{1} := \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \quad \mu := -\nu$$

that is, $A \in \mathbb{R}^{m \times n}$ is the matrix formulated from the representations of m observations in the n -dimensional Euclidean vector space, $\mathbf{d} \in \mathbb{R}^m$, $[\mathbf{d}]_j \in \{-1, 1\}$ is a vector describing the classification of the observations, $D := \text{diag}(\mathbf{d}) \in \mathbb{R}^{m \times m}$, and $\mathbf{1} \in \mathbb{R}^m$. The variable $\mu := -\nu$ was defined only to get rid of notationally inconvenient subtractions. It is easy to see that with our new notations, Assumption 2.2 is just

$$DA\mathbf{u} + \mathbf{d}\mu \geq \mathbf{1}. \quad (5)$$

Of course many separating hyperplanes can be constructed for a given dataset, even if we require that they be in their canonical forms. One could argue that separating hyperplanes farther away from the nearest observation, that is, hyperplanes having a wider so-called margin around them, are separating the two classes of observations in

a safer and more stable way, thus having more predictive power regarding categorizing future data. Indeed, the optimality of this intuitive requirement can be rigorously proved in statistical learning theory. The support vector machine is concerned with finding the hyperplane which separates the classes of observations with the maximal margin.

Definition 2.6 (Optimal separating hyperplane).

We say that a separating hyperplane given in its canonical form $\mathcal{H}(\mathbf{u}^*, \mu^*) \subseteq \mathbb{R}^n$ is *optimal* if

$$(\mathbf{u}^*, \mu^*) = \arg \max_{\mathbf{u}, \mu} \left\{ \min_{\mathbf{a}_j \in \mathcal{J}} \{ \delta(\mathbf{a}_j, \mathcal{H}(\mathbf{u}, \mu)) \} \right\}.$$

It can be shown that the optimal separating hyperplane is unique, see e.g. [39, p. 402-403]. That is, if we require canonical representation then the optimality condition coupled with the separating criterion in form (5) yields

$$\left. \begin{aligned} \max_{\mathbf{u}, \mu} \left\{ \min_{\mathbf{a}_j \in \mathcal{J}} \{ \delta(\mathbf{a}_j, \mathcal{H}(\mathbf{u}, \mu)) \} \right\} \\ D\mathbf{A}\mathbf{u} + \mathbf{d}\mu \geq \mathbf{1} \end{aligned} \right\}$$

Remark that the set of optimal solutions to this optimization problem can contain at most one element $(\mathbf{u}, \mu)$, since there is only one optimal separating hyperplane [39, p. 402-403] and its canonical representation is unique (Remark 2.4). Moreover, by applying Lemma 2.5 we get that our optimization problem can be written as

$$\left. \begin{aligned} \max_{\mathbf{u}, \mu} \frac{1}{\|\mathbf{u}\|} \\ D\mathbf{A}\mathbf{u} + \mathbf{d}\mu \geq \mathbf{1} \end{aligned} \right\}$$

which is obviously equivalent to the linearly constrained quadratic programming problem of

$$\left. \begin{aligned} \min_{\mathbf{u}, \mu} \frac{1}{2} \mathbf{u}^\top \mathbf{u} \\ D\mathbf{A}\mathbf{u} + \mathbf{d}\mu \geq \mathbf{1} \end{aligned} \right\} \text{ (SVM-P)}$$

3 The quadratic optimization problem of the SVM

3.1 Primal problem, optimality conditions

In Section 2, we have derived that the simplest form of the support vector machine is the quadratic optimization problem (SVM-P). The Lagrangian function $\mathcal{L} : \mathbb{R}^{n+1} \times \mathbb{R}_{\oplus}^m \rightarrow \mathbb{R}$

of the problem is

$$\mathcal{L}(\mathbf{u}, \mu, \mathbf{v}) = \frac{1}{2} \mathbf{u}^\top \mathbf{u} + \mathbf{v}^\top (\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu)$$

where $\mathbf{v}$ is the Lagrange multiplier vector.

Theorem 3.1. *The optimality conditions of problem (SVM-P) are*

$$\text{Primal feasibility:} \quad \mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu \leq \mathbf{0} \quad (6a)$$

$$\text{Dual feasibility:} \quad \mathbf{v} \geq \mathbf{0} \quad (6b)$$

$$\text{Complementarity:} \quad \mathbf{v}^\top (\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu) = 0 \quad (6c)$$

$$\text{Stationarity } (\mathbf{u}): \quad \mathbf{u} - (D\mathbf{A})^\top \mathbf{v} = \mathbf{0} \quad (6d)$$

$$\text{Stationarity } (\mu): \quad -\mathbf{d}^\top \mathbf{v} = 0 \quad (6e)$$

Proof. It is apparent that the square of the Euclidean norm is convex, since norms are convex functions by definition, the square function id^2 is convex and nondecreasing on $[0, \infty)$, thus the composition $\text{id}^2 \circ \|\cdot\|$ is convex. This in combination with its linear constraints implies that problem (SVM-P) is convex (see Section A.2), therefore the Karush-Kuhn-Tucker conditions (see Theorem A.10) are necessary and sufficient conditions for a solution of our problem to be optimal². Thus we only need to show that the KKT conditions (13) indeed take the form (6) in the special case of problem (SVM-P).

Rearranging the constraints $D\mathbf{A}\mathbf{u} + \mathbf{d}\mu \geq \mathbf{1}$ trivially yields the primal feasibility criteria $\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu \leq \mathbf{0}$. Dual feasibility is also trivial. To prove complementarity, observe that by the definition of matrix multiplication

$$\mathbf{v}^\top (\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu) = \sum_{j=1}^m [\mathbf{v}]_j [\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j$$

which is evidently the left hand side of equation (13c).

Recall that the decision variables are $\mathbf{u} \in \mathbb{R}^n$ and $\mu \in \mathbb{R}$, thus the gradient vectors are of dimension $n + 1$. Directly substituting into (13d) gives

$$\begin{bmatrix} \frac{\partial}{\partial u_1} \left(\frac{1}{2} \mathbf{u}^\top \mathbf{u} + 0\mu \right) \\ \vdots \\ \frac{\partial}{\partial u_n} \left(\frac{1}{2} \mathbf{u}^\top \mathbf{u} + 0\mu \right) \\ \frac{\partial}{\partial \mu} \left(\frac{1}{2} \mathbf{u}^\top \mathbf{u} + 0\mu \right) \end{bmatrix} + \sum_{j=1}^m v_j \begin{bmatrix} \frac{\partial}{\partial u_1} [\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j \\ \vdots \\ \frac{\partial}{\partial u_n} [\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j \\ \frac{\partial}{\partial \mu} [\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad (7)$$

²The constraints of (SVM-P) are linear functions, thus the regularity condition of Theorem A.10 is satisfied (see e.g. [7]).

But $\frac{1}{2} \mathbf{u}^\top \mathbf{u} = \frac{1}{2} \sum_{i=1}^n u_i^2$, thus the gradient vector of the objective function is $[u_1, \dots, u_n, 0]^\top$.

Carrying out the matrix multiplication in the second term:

$$[\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j = 1 - \sum_{i=1}^n u_i d_j a_{ji} - d_j \mu.$$

Therefore the derivatives of the constraint are $\frac{\partial}{\partial u_k} [\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j = -d_j a_{jk}$, and $\frac{\partial}{\partial \mu} [\mathbf{1} - D\mathbf{A}\mathbf{u} - \mathbf{d}\mu]_j = -d_j$.

Consequently, (7) is just

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \\ 0 \end{bmatrix} + \begin{bmatrix} \sum_{j=1}^m -v_j d_j a_{j1} \\ \vdots \\ \sum_{j=1}^m -v_j d_j a_{jn} \\ \sum_{j=1}^m -v_j d_j \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}.$$

The last row can be rewritten as $-\mathbf{d}^\top \mathbf{v} = 0$, which is condition (6e). Thus it remains to show that

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} + \begin{bmatrix} \sum_{j=1}^m -v_j d_j a_{j1} \\ \vdots \\ \sum_{j=1}^m -v_j d_j a_{jn} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \iff \mathbf{u} - (DA)^\top \mathbf{v} = \mathbf{0}.$$

Which follows immediately from the definition of matrix multiplication, since

$$DA = \begin{bmatrix} d_1 a_{11} & \dots & d_1 a_{1n} \\ d_2 a_{21} & \dots & d_2 a_{2n} \\ \vdots & \ddots & \vdots \\ d_m a_{m1} & \dots & d_m a_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}, \quad (DA)^\top = \begin{bmatrix} d_1 a_{11} & d_2 a_{21} & \dots & d_m a_{m1} \\ \vdots & \vdots & \ddots & \vdots \\ d_1 a_{1n} & d_2 a_{2n} & \dots & d_m a_{mn} \end{bmatrix} \in \mathbb{R}^{n \times m}$$

and of course $\mathbf{v} = [v_1, v_2, \dots, v_m]^\top \in \mathbb{R}^m$.

□

3.2 Lagrange dual problem and weak duality

The problem (SVM-P) is so special that we can obtain its optimality conditions without the Karush-Kuhn-Tucker theorem, by formulating its Lagrange dual problem and proving

the weak duality theorem. Following Section A.2 of Appendix, we define

$$\psi(\mathbf{v}) = \inf_{\mathbf{u}, \mu} \mathcal{L}(\mathbf{u}, \mu, \mathbf{v}) = \inf_{\mathbf{u}, \mu} \left\{ \frac{1}{2} \mathbf{u}^\top \mathbf{u} + \mathbf{v}^\top (\mathbf{1} - DA\mathbf{u} - \mathbf{d}\mu) \right\}$$

where μ , $\mathbf{u}$ are primal feasible and $\mathbf{v} \geq \mathbf{0}$. In order to obtain the explicit form of the Lagrange dual problem, we have to solve this extreme value problem. Note that

$$\begin{aligned} \nabla \mathcal{L}(\mathbf{u}, \mu, \mathbf{v}) = \mathbf{0} &\iff \nabla \left(\frac{1}{2} \mathbf{u}^\top \mathbf{u} \right) + \sum_{j=1}^m [\mathbf{v}]_j \nabla [\mathbf{1} - DA\mathbf{u} - \mathbf{d}\mu]_j = \mathbf{0} \iff \\ &\iff \begin{cases} \mathbf{u} - (DA)^\top \mathbf{v} = \mathbf{0} \\ -\mathbf{d}^\top \mathbf{v} = 0 \end{cases} \end{aligned}$$

where the last equivalence is shown at the end of the proof of Theorem 3.1. The Lagrangian function is clearly unbounded if $\mathbf{d}^\top \mathbf{v} \neq 0$. But if $\mathbf{d}^\top \mathbf{v} = 0$, then substituting $\mathbf{u} = (DA)^\top \mathbf{v}$ back into the Lagrangian function gives

$$\begin{aligned} &\frac{1}{2} ((DA)^\top \mathbf{v})^\top (DA)^\top \mathbf{v} + \mathbf{v}^\top (\mathbf{1} - DA(DA)^\top \mathbf{v} - \mathbf{d}\mu) = \\ &= \frac{1}{2} \mathbf{v}^\top DA(DA)^\top \mathbf{v} + \mathbf{v}^\top \mathbf{1} - \mathbf{v}^\top DA(DA)^\top \mathbf{v} = -\frac{1}{2} \mathbf{v}^\top DA(DA)^\top \mathbf{v} + \mathbf{v}^\top \mathbf{1}. \end{aligned}$$

That is,

$$\psi(\mathbf{v}) = \begin{cases} -\frac{1}{2} \mathbf{v}^\top DA(DA)^\top \mathbf{v} + \mathbf{v}^\top \mathbf{1} & \text{if } \mathbf{d}^\top \mathbf{v} = 0 \\ -\infty & \text{otherwise} \end{cases}$$

Thus the explicit form of the Lagrange dual problem of (SVM-P) is

$$\left. \begin{aligned} \max_{\mathbf{v}} \quad & -\frac{1}{2} \mathbf{v}^\top DA(DA)^\top \mathbf{v} + \mathbf{v}^\top \mathbf{1} \\ & \mathbf{v} \geq \mathbf{0} \\ & \mathbf{d}^\top \mathbf{v} = 0 \end{aligned} \right\} \quad (\text{SVM-D})$$

Theorem 3.2 (Weak duality theorem for SVM).

Suppose $(\hat{\mathbf{u}}, \hat{\mu})$ is a feasible solution of problem (SVM-P), and $\hat{\mathbf{v}}$ is a feasible solution of problem (SVM-D). Then

$$\frac{1}{2} \hat{\mathbf{u}}^\top \hat{\mathbf{u}} \geq -\frac{1}{2} \hat{\mathbf{v}}^\top DA(DA)^\top \hat{\mathbf{v}} + \hat{\mathbf{v}}^\top \mathbf{1}$$

Moreover, equality holds if and only if $\hat{\mathbf{v}}^\top (\mathbf{1} - DA\hat{\mathbf{u}} - \mathbf{d}\hat{\mu}) = 0$ and $(DA)^\top \hat{\mathbf{v}} - \hat{\mathbf{u}} = \mathbf{0}$.

Proof. By definition and an elementary property of the Euclidean norm

$$0 \leq \left\| (DA)^\top \hat{\mathbf{v}} - \hat{\mathbf{u}} \right\|^2 = \left\| (DA)^\top \hat{\mathbf{v}} \right\|^2 - 2\hat{\mathbf{v}}^\top DA\hat{\mathbf{u}} + \|\hat{\mathbf{u}}\|^2. \quad (8)$$

Simple rearranging gives

$$\frac{1}{2} \|\hat{\mathbf{u}}\|^2 \geq \hat{\mathbf{v}}^\top DA\hat{\mathbf{u}} - \frac{1}{2} \left\| (DA)^\top \hat{\mathbf{v}} \right\|^2.$$

Since $(\hat{\mathbf{u}}, \hat{\mu})$ and $\hat{\mathbf{v}}$ are feasible solutions, $DA\hat{\mathbf{u}} + \mathbf{d}\hat{\mu} \geq \mathbf{1}$, $\mathbf{d}^\top \hat{\mathbf{v}} = 0$, $\hat{\mathbf{v}} \geq \mathbf{0}$ holds. Then it immediately follows that

$$\hat{\mathbf{v}}^\top DA\hat{\mathbf{u}} + \hat{\mathbf{v}}^\top \mathbf{d}\hat{\mu} \geq \hat{\mathbf{v}}^\top \mathbf{1} \implies \hat{\mathbf{v}}^\top DA\hat{\mathbf{u}} \geq \hat{\mathbf{v}}^\top \mathbf{1}.$$

Thus

$$\frac{1}{2} \|\hat{\mathbf{u}}\|^2 \geq \hat{\mathbf{v}}^\top DA\hat{\mathbf{u}} - \frac{1}{2} \left\| (DA)^\top \hat{\mathbf{v}} \right\|^2 \geq \hat{\mathbf{v}}^\top \mathbf{1} - \frac{1}{2} \left\| (DA)^\top \hat{\mathbf{v}} \right\|^2, \quad (9)$$

which proves the first statement of the theorem.

In the first inequality of (9), equality holds if the inequality (8) is satisfied by equality. This is equivalent to $(DA)^\top \hat{\mathbf{v}} - \hat{\mathbf{u}} = \mathbf{0}$ by the definition of norm.

In the second inequality of (9), equality holds if $\hat{\mathbf{v}}^\top DA\hat{\mathbf{u}} + \hat{\mathbf{v}}^\top \mathbf{d}\hat{\mu} = \hat{\mathbf{v}}^\top \mathbf{1}$. That is, if $\hat{\mathbf{v}}^\top (\mathbf{1} - DA\hat{\mathbf{u}} - \mathbf{d}\hat{\mu}) = 0$. $\square$

Theorem 3.3 (Weak equilibrium theorem for SVM).

Suppose $(\mathbf{u}^*, \mu^*)$ is a feasible solution of problem (SVM-P), $\mathbf{v}^*$ is a feasible solution of problem (SVM-D), and

$$\frac{1}{2} (\mathbf{u}^*)^\top \mathbf{u}^* = -\frac{1}{2} (\mathbf{v}^*)^\top DA (DA)^\top \mathbf{v}^* + (\mathbf{v}^*)^\top \mathbf{1}$$

holds. Then $(\mathbf{u}^*, \mu^*)$ is a primal optimal solution and $\mathbf{v}^*$ is a dual optimal solution.

Proof. Let $(\hat{\mathbf{u}}, \hat{\mu})$ be an arbitrary primal feasible solution. Then applying the weak duality theorem (Theorem 3.2) gives

$$\frac{1}{2} \hat{\mathbf{u}}^\top \hat{\mathbf{u}} \geq -\frac{1}{2} (\mathbf{v}^*)^\top DA (DA)^\top \mathbf{v}^* + (\mathbf{v}^*)^\top \mathbf{1} = \frac{1}{2} (\mathbf{u}^*)^\top \mathbf{u}^*$$

which means exactly that $(\mathbf{u}^*, \mu^*)$ is primal optimal. The optimality of $\mathbf{v}^*$ is proved similarly. $\square$

Corollary 3.4. The optimality conditions of problem (SVM-P) are indeed those of (6).

3.3 The mixed linear complementarity problem

Introducing the slack variable $\mathbf{s} := D\mathbf{A}\mathbf{u} + \mathbf{d}\mu - \mathbf{1}$ makes it possible to rewrite the optimality conditions (6) to

$$\mathbf{s} - \mathbf{d}\mu - D\mathbf{A}\mathbf{u} = -\mathbf{1} \quad (10a)$$

$$\mathbf{d}^\top \mathbf{v} = 0 \quad (10b)$$

$$(D\mathbf{A})^\top \mathbf{v} - \mathbf{u} = \mathbf{0} \quad (10c)$$

$$\mathbf{v}^\top \mathbf{s} = 0 \quad (10d)$$

$$\mathbf{v}, \mathbf{s} \geq \mathbf{0} \quad (10e)$$

Conditions (10a)-(10c) in matrix form are

$$\begin{bmatrix} s_1 \\ \vdots \\ s_m \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & \dots & 0 & | & d_1 & d_1 a_{11} & \dots & d_1 a_{1n} \\ \vdots & \ddots & \vdots & | & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & | & d_m & d_m a_{m1} & \dots & d_m a_{mn} \\ \hline -d_1 & \dots & -d_m & | & 0 & 0 & \dots & 0 \\ -d_1 a_{11} & \dots & -d_m a_{m1} & | & 0 & 1 & \dots & 0 \\ \vdots & \ddots & \vdots & | & \vdots & \vdots & \ddots & \vdots \\ -d_1 a_{1n} & \dots & -d_m a_{mn} & | & 0 & 0 & \dots & 1 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ \vdots \\ v_m \\ \mu \\ u_1 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} -1 \\ \vdots \\ -1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

For notational convenience, we define an augmented matrix $C := [\mathbf{d} \mid D\mathbf{A}] \in \mathbb{R}^{m \times (n+1)}$.

Then the optimality conditions of problem (SVM-P) are

$$\left. \begin{aligned} & \underbrace{\begin{bmatrix} \mathbf{s} \\ \mathbf{f} \end{bmatrix}}_{\mathbf{z}} - \underbrace{\begin{bmatrix} 0 & | & C \\ -C^\top & | & 0 \\ \hline & & I_n \end{bmatrix}}_M \underbrace{\begin{bmatrix} \mathbf{v} \\ \mu \\ \mathbf{u} \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} -\mathbf{1} \\ \mathbf{0} \end{bmatrix}}_{\mathbf{q}} \\ & \mathbf{z}^\top \mathbf{x} = 0 \\ & \mathbf{v}, \mathbf{s} \geq \mathbf{0} \\ & \mathbf{f} = \mathbf{0} \end{aligned} \right\} \quad (\text{MLCP-SVM})$$

where $\mathbf{s} \in \mathbb{R}^m$, $\mathbf{f} \in \mathbb{R}^{n+1}$, $\mathbf{z}, \mathbf{x}, \mathbf{q} \in \mathbb{R}^{m+n+1}$ and $M \in \mathbb{R}^{(m+n+1) \times (m+n+1)}$.

Observe that if vector $\mathbf{x}$ would be required to be nonnegative, then the optimality conditions would result in a linear complementarity problem (LCP). Moreover, it is apparent

that matrix M is bisymmetric: the diagonal blocks are symmetric and positive semidefinite, while the remaining matrix $\begin{bmatrix} 0 & C \\ -C^\top & 0 \end{bmatrix}$ is skew symmetric, namely it is a square matrix whose transpose equals its negative. Klafszky and Terlaky [24] showed that if matrix M is bisymmetric, their quadratic criss-cross algorithm is convergent. However, in our case the vector $\mathbf{x}$ is not necessarily nonnegative, since even if $\mathbf{v} \geq \mathbf{0}$ is required, $\mathbf{u}$ and μ are free variables. Consequently, the optimality conditions (MLCP-SVM) are more general than a linear complementarity problem. Since the classical criss-cross algorithm works on an LCP, our aim for the rest of this section is to find out whether our problem (MLCP-SVM) could be converted into a linear complementarity problem.

A generalization of the LCP is the mixed linear complementarity problem (MLCP), which is an LCP mixed with a system of linear equations. Following Cottle, Pang and Stone [13], the MLCP is concerned with finding vectors $\boldsymbol{\xi} \in \mathbb{R}^k$, $\boldsymbol{\eta} \in \mathbb{R}^l$ satisfying

$$\left. \begin{aligned} \boldsymbol{\alpha} + P\boldsymbol{\xi} + R\boldsymbol{\eta} &= \mathbf{0} \\ \boldsymbol{\beta} + S\boldsymbol{\xi} + Q\boldsymbol{\eta} &\geq \mathbf{0} \\ \boldsymbol{\eta} &\geq \mathbf{0} \\ \boldsymbol{\eta}^\top (\boldsymbol{\beta} + S\boldsymbol{\xi} + Q\boldsymbol{\eta}) &= 0 \end{aligned} \right\} \quad (\text{MLCP})$$

where $P \in \mathbb{R}^{k \times k}$, $Q \in \mathbb{R}^{l \times l}$, $R \in \mathbb{R}^{k \times l}$, $S \in \mathbb{R}^{l \times k}$, $\boldsymbol{\alpha} \in \mathbb{R}^k$, $\boldsymbol{\beta} \in \mathbb{R}^l$ are given. The novelty of this problem class is that the vector $\boldsymbol{\xi}$ is not required to be nonnegative.

Remark 3.5. The problem (MLCP-SVM) is a mixed linear complementarity problem. Indeed, $\boldsymbol{\xi} := \begin{bmatrix} \mu \\ \mathbf{u} \end{bmatrix}$, $\boldsymbol{\eta} := \mathbf{v}$, $\boldsymbol{\alpha} := \mathbf{0}$, $P := \begin{bmatrix} 0 & \\ & -I_n \end{bmatrix}$, $R := C^\top$, $\boldsymbol{\beta} := -\mathbf{1}$, $S := C$, $Q := 0$.

In linear programming, one could note that a problem with free variables can always be converted into a problem with nonnegative ones, because any vector can be written as the difference of two nonnegative vectors by the identity

$$[\mathbf{x}]_i = \max \{0, [\mathbf{x}]_i\} + \min \{0, [\mathbf{x}]_i\}.$$

The reason for this is that classical versions of the simplex method can only operate on problems with nonnegative variables. However, this conversion results in doubling the size of the problem, which can be really unfavourable from a computational point of view. This motivated researchers to develop variants capable of handling all types of variables.

A particular algorithm is the General Primal Simplex Method of István Maros [27], the principles of which motivated our modified criss-cross algorithm.

Given the aforementioned conversion of free variables in a linear programming problem, it is reasonable to ask whether a mixed linear complementarity problem could be transformed into an LCP. The following result can be found in [13].

Theorem 3.6. *If the matrix P is nonsingular, the mixed linear complementarity problem (MLCP) can be converted into an LCP.*

Proof. If P is nonsingular, then the equation

$$\alpha + P\xi + R\eta = 0$$

can be solved for ξ :

$$\xi = P^{-1}(-\alpha - R\eta).$$

Plugging this solution of ξ into the remaining conditions of (MLCP) yields

$$\begin{aligned} \beta - SP^{-1}\alpha + (Q - SP^{-1}R)\eta &\geq 0 \\ \eta &\geq 0 \\ \eta^\top (\beta - SP^{-1}\alpha + (Q - SP^{-1}R)\eta) &= 0 \end{aligned}$$

which is indeed a linear complementarity problem with coefficient matrix $Q - SP^{-1}R$ and right hand side vector $\beta - SP^{-1}\alpha$. $\square$

Observe that this theorem cannot be applied to our problem (MLCP-SVM), since the matrix $\begin{bmatrix} 0 & \\ & -I_n \end{bmatrix}$ is singular. Cottle, Stone and Pang [13] writes that in the case where the matrix P is singular, the conversion of the problem (MLCP) into a standard LCP is ‘not an entirely straightforward matter’. Thus for our problem, the algorithm of Klafszky and Terlaky is not applicable, since it operates on an LCP, therefore solving the quadratic optimization problem of the support vector machine with a criss-cross-type algorithm is a nontrivial task.

To end this section, we remark that even if there was an easy way to convert our problem into an LCP, the possibly vast increase in the size of the problem still inclines us to instead generalize the criss-cross algorithm to the (MLCP-SVM).

3.4 Other specific properties of the problem

The derivation of the MLCP in Section 3.3 is applicable for all quadratic optimization problems. However, given the particularity of problem (SVM-P), an alternative formulation can be obtained, to which this section pertains.

Consider optimality conditions (10) and observe that its matrix form contains an identity as a submatrix. This motivates us to eliminate the corresponding variables from the system. Indeed, equation (10c) can be solved for $\mathbf{u}$:

$$\mathbf{u} = (DA)^\top \mathbf{v}. \quad (11)$$

Substituting this into equation (10a) results in the following formulation of optimality conditions for problem (SVM-P):

$$\mathbf{s} - \mathbf{d}\mu - DA(DA)^\top \mathbf{v} = -\mathbf{1} \quad (12a)$$

$$\mathbf{d}^\top \mathbf{v} = 0 \quad (12b)$$

$$\mathbf{v}^\top \mathbf{s} = 0 \quad (12c)$$

$$\mathbf{v}, \mathbf{s} \geq \mathbf{0} \quad (12d)$$

That is, our alternative formulation of the optimality conditions of problem (SVM-P) is

$$\left. \begin{array}{l} \underbrace{\begin{bmatrix} \mathbf{s} \\ f \end{bmatrix}}_{\widehat{\mathbf{z}}} - \underbrace{\begin{bmatrix} DA(DA)^\top & \mathbf{d} \\ -\mathbf{d}^\top & 0 \end{bmatrix}}_{\widehat{M}} \underbrace{\begin{bmatrix} \mathbf{v} \\ \mu \end{bmatrix}}_{\widehat{\mathbf{x}}} = \underbrace{\begin{bmatrix} -\mathbf{1} \\ 0 \end{bmatrix}}_{\widehat{\mathbf{q}}} \\ \widehat{\mathbf{z}}^\top \widehat{\mathbf{x}} = 0 \\ \mathbf{v}, \mathbf{s} \geq \mathbf{0} \\ f = 0 \end{array} \right\} \quad (\text{MLCP-SVM-2})$$

where $f \in \mathbb{R}$, $\widehat{\mathbf{z}}, \widehat{\mathbf{x}}, \widehat{\mathbf{q}} \in \mathbb{R}^{m+1}$ and $\widehat{M} \in \mathbb{R}^{(m+1) \times (m+1)}$ is bisymmetric. It is easy to check that this formulation is also a mixed linear complementarity problem, and that Theorem 3.6 is not applicable now either, since the zero matrix is not invertible.

Remark that our new formulation is considerably smaller in size than problem (MLCP-SVM). In addition, there are significantly fewer zeros in the diagonal of matrix $\widehat{M}$ than in the diagonal of matrix M , which may result in the modified criss-cross algorithm performing more diagonal and fewer exchange pivots when we run it on $\widehat{M}$ instead of M . In

the light of these remarks, one could expect the new formulation (MLCP-SVM-2) to be computationally favorable. This hypothesis is empirically investigated in Section 4.4.

4 The criss-cross algorithm

In this section we present our new criss-cross-type algorithm variant adapted to the special structure of problem (SVM-P). Before that, Klafszky and Terlaky’s standard QP variant is examined, since the criss-cross method is somewhat less known compared to other algorithms such as simplex or interior-point methods.

4.1 Intuitive description of LP algorithm classes

Real-world linear programming problems are usually solved by simplex or interior-point algorithms. Denote the set of feasible solutions to an LP problem by $\mathcal{F}$. It is well-known that $\mathcal{F}$ is a convex polyhedron, and that its extremal points are exactly basic feasible solutions to the LP in standard form. To put it simply, simplex algorithms move along the extremal points of this polyhedron in a way that feasibility is always preserved and the value of the objective function improves or remains the same when moving to an adjacent extremal point. As their name suggests, interior-point methods reach optimal solution by traversing the interior of the polyhedron $\mathcal{F} \neq \emptyset$, thus they also preserve feasibility of the current solution.

On the sharp contrary, criss-cross algorithms do not try to maintain feasibility and they do not even take into account the objective function. The former indicates that these algorithms could move onto points outside of $\mathcal{F}$, while the latter means that they often traverse along seemingly counterintuitive, ‘crisscrossing’ paths to optimum. The different principles of the three classes of algorithms are visualized in Figure 1.

The criss-cross method has exponential worst-case complexity [29] similar to simplex algorithms [25], while interior-point methods are polynomial [23]. However, in practice simplex algorithms are remarkably efficient, while criss-cross methods to date are not [22]. Despite this, Komei Fukuda conjectures that the criss-cross method is an expected polynomial-time algorithm, that is, the expected number of pivots performed by it with random permutation of variables is bounded above by a polynomial function [19].

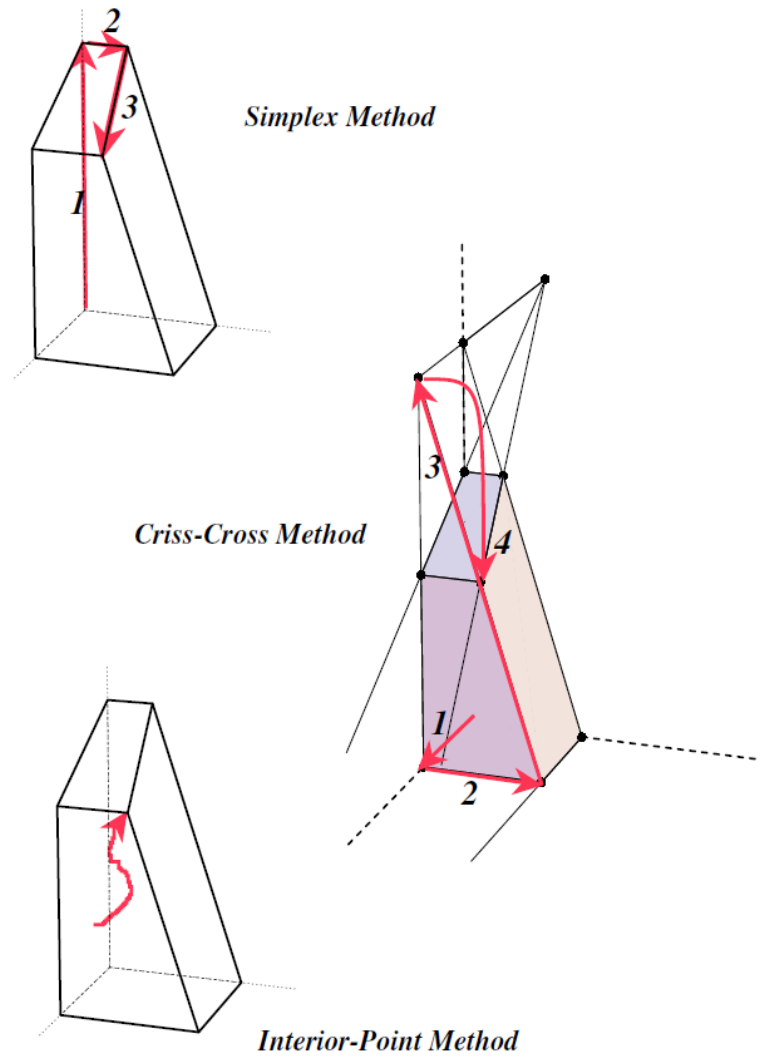


Figure 1: Paradigms of the simplex, interior-point and criss-cross algorithms. Source: [19]

4.2 The Klafszky-Terlaky quadratic criss-cross algorithm

Consider the standard linearly constrained, nonnegative primal-dual quadratic programming problem

$$\left. \begin{aligned} \min_{\boldsymbol{\xi}, \boldsymbol{\zeta}} \quad & \boldsymbol{\gamma}^\top \boldsymbol{\xi} + \frac{1}{2} \boldsymbol{\xi}^\top G^\top G \boldsymbol{\xi} + \frac{1}{2} \boldsymbol{\zeta}^\top \boldsymbol{\zeta} \\ & E \boldsymbol{\xi} + F \boldsymbol{\zeta} \geq \boldsymbol{\beta} \\ & \boldsymbol{\xi} \geq \mathbf{0} \end{aligned} \right\} \text{(QP-P)} \quad \left. \begin{aligned} \max_{\boldsymbol{\eta}, \boldsymbol{\omega}} \quad & \boldsymbol{\beta}^\top \boldsymbol{\eta} - \frac{1}{2} \boldsymbol{\eta}^\top F F^\top \boldsymbol{\eta} - \frac{1}{2} \boldsymbol{\omega}^\top \boldsymbol{\omega} \\ & E^\top \boldsymbol{\eta} - G^\top \boldsymbol{\omega} \leq \boldsymbol{\gamma} \\ & \boldsymbol{\eta} \geq \mathbf{0} \end{aligned} \right\} \text{(QP-D)}$$

where $E \in \mathbb{R}^{m \times n}$, $F \in \mathbb{R}^{m \times k}$, $G \in \mathbb{R}^{l \times n}$ are arbitrary matrices and $\boldsymbol{\xi}, \boldsymbol{\gamma} \in \mathbb{R}^n$, $\boldsymbol{\eta}, \boldsymbol{\beta} \in \mathbb{R}^m$, $\boldsymbol{\zeta} \in \mathbb{R}^k$, $\boldsymbol{\omega} \in \mathbb{R}^l$. Note that the optimization problem of the SVM does not fit into this framework, because here the nonnegativity of the primal variable $\boldsymbol{\xi}$ is required and heavily exploited in the following derivation, while in problem (SVM-P) the variables $\mathbf{u}, \mu$ are free.

The optimality conditions of problems (QP-P)-(QP-D) can be derived similarly to what we have seen regarding problems (SVM-P) and (SVM-D) (weak duality theorem or Karush-Kuhn-Tucker theorem). The conditions are the following:

$$\left. \begin{aligned} -F F^\top \boldsymbol{\eta} - E \boldsymbol{\xi} + \bar{\boldsymbol{\eta}} &= -\boldsymbol{\beta} \\ E^\top \boldsymbol{\eta} - G^\top G \boldsymbol{\xi} + \bar{\boldsymbol{\xi}} &= \boldsymbol{\gamma} \\ \boldsymbol{\xi}, \quad \boldsymbol{\eta}, \quad \bar{\boldsymbol{\xi}}, \quad \bar{\boldsymbol{\eta}} &\geq \mathbf{0} \\ \boldsymbol{\xi}^\top \bar{\boldsymbol{\xi}} = 0, \quad \boldsymbol{\eta}^\top \bar{\boldsymbol{\eta}} &= 0 \end{aligned} \right\}$$

where $\bar{\boldsymbol{\xi}} := E \boldsymbol{\xi} + F \boldsymbol{\zeta} - \boldsymbol{\beta}$, $\bar{\boldsymbol{\eta}} := \boldsymbol{\gamma} - \boldsymbol{\eta}^\top E + \boldsymbol{\omega}^\top G$ are primal and dual slack variables. Clearly, for an arbitrary vector $\mathbf{a}$ of appropriate dimension we have

$$\mathbf{a}^\top G^\top G \mathbf{a} = (G \mathbf{a})^\top (G \mathbf{a}) \geq 0,$$

therefore matrix $G^\top G$ is positive semidefinite. Similarly, $F F^\top$ is positive semidefinite as well. It is well-known (see e.g. [12]) that the optimality conditions are equivalent to a linear complementarity problem³

$$\left. \begin{aligned} -T \mathbf{y} + \mathbf{w} &= \mathbf{p} \\ \mathbf{y}, \mathbf{w} &\geq \mathbf{0} \\ \mathbf{y}^\top \mathbf{w} &= 0 \end{aligned} \right\} \text{(LCP-QP)}$$

³Another common formulation is writing $\mathbf{y} \odot \mathbf{w} = 0$ instead of $\mathbf{y}^\top \mathbf{w} = 0$, where the $\odot : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}^N$, $[\mathbf{y} \odot \mathbf{w}]_i := [\mathbf{y}]_i [\mathbf{w}]_i$ operator is the Hadamard product. Of course, the requirement $\mathbf{y}, \mathbf{w} \geq \mathbf{0}$ makes this formulation equivalent to our inner product one.

where $T := \begin{bmatrix} FF^\top & E \\ -E^\top & G^\top G \end{bmatrix} \in \mathbb{R}^{(m+n) \times (m+n)}$ $\mathbf{y} := \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\xi} \end{bmatrix} \in \mathbb{R}^{m+n}$, $\mathbf{w} := \begin{bmatrix} \bar{\boldsymbol{\eta}} \\ \bar{\boldsymbol{\xi}} \end{bmatrix} \in \mathbb{R}^{m+n}$, and $\mathbf{p} := \begin{bmatrix} -\boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix} \in \mathbb{R}^{m+n}$. That is, problems (QP-P) and (QP-D) have an optimal solution if and only if problem (LCP-QP) have an optimal solution, i.e. there exists a decision variable $\mathbf{y}$ which satisfies the three conditions of the linear complementarity problem. We have seen that $FF^\top$ and $G^\top G$ are positive semidefinite, and the remaining matrix $\begin{bmatrix} 0 & E \\ -E^\top & 0 \end{bmatrix}$ is skew-symmetric, which means that matrix T is bisymmetric by definition.

If we let $\mathbf{y} = \mathbf{0}$ and $\mathbf{w} = \mathbf{p}$, then two conditions of the problem (LCP-QP) will be fulfilled. Namely, $-T\mathbf{y} + \mathbf{w} = \mathbf{p}$ and the complementarity condition $\mathbf{y}^\top \mathbf{w} = 0$ are satisfied; only $\mathbf{w} \geq \mathbf{0}$ could be unfulfilled⁴. Starting from this trivial basic solution, our aim is to take steps such that conditions $-T\mathbf{y} + \mathbf{w} = \mathbf{p}$ and $\mathbf{y}^\top \mathbf{w} = 0$ remain satisfied, and after finitely many of such steps, the nonnegativity condition is either fulfilled or a proof that problem (LCP-QP) has no solution is found.

The linear system $-T\mathbf{y} + \mathbf{w} = \mathbf{p}$ can be represented as the basic tableau

	$\mathbf{y}^\top$	$\mathbf{w}^\top$	
$\mathbf{w}$	$-T$	I	$\mathbf{p}$

where the variables in the left column form the current basis, which is $\mathbf{w}$ with our starting choice of $\mathbf{y} = \mathbf{0}$. It is well-known that a change of basis does not alter the solution of a linear system. Therefore by performing pivot steps condition $-T\mathbf{y} + \mathbf{w} = \mathbf{p}$ will remain satisfied. However, we also have to preserve complementarity, that is, $\mathbf{y}^\top \mathbf{w} = 0$ must remain satisfied. Consequently, only special pivot steps are allowed: if a set of variables $([\mathbf{w}]_\gamma)_{\gamma \in \Gamma}$ leave the basis, then they can only be replaced by their complementary variables $([\mathbf{y}]_\gamma)_{\gamma \in \Gamma}$ and vice versa. Such basic transformations are called principal pivot transformations and were discovered by Tucker [38].

We are concerned with two types of principal pivot transformations. If only one variable leaves the basis and its complementary variable replaces it, then we perform a so-called

⁴Clearly if the problem is very special, that is, $\mathbf{p} \geq \mathbf{0}$, then this trivial choice of $\mathbf{y}$ and $\mathbf{w}$ is a solution to the linear complementarity problem (LCP-QP). Of course in practice this fortunate case is extremely rare, and we have seen that the optimization problem of the support vector machine never has a nonnegative right hand side vector $\mathbf{p}$.

diagonal pivot step. Of course if the corresponding diagonal element of the matrix M is 0, then a diagonal pivot cannot be performed. If two basic variables are replaced with their complementary pair, that is, $[\mathbf{w}]_i$ is replaced with $[\mathbf{y}]_j$ and $[\mathbf{w}]_j$ is replaced by $[\mathbf{y}]_i$, then we call the basic transformation an exchange pivot.

In order to present the minimal index quadratic criss-cross algorithm of Klafszky and Terlaky, we need to state some lemmas.

Lemma 4.1. *A bisymmetric real matrix is positive semidefinite.*

Proof. Let $T = \begin{bmatrix} F & E \\ -E^\top & G \end{bmatrix}$ be a bisymmetric real matrix, that is, F and G are positive semidefinite. Then by the definition of matrix multiplication $\mathbf{a}^\top T \mathbf{a} = \mathbf{a}^\top \begin{bmatrix} F & 0 \\ 0 & 0 \end{bmatrix} \mathbf{a} - \mathbf{a}^\top \begin{bmatrix} 0 & 0 \\ E^\top & 0 \end{bmatrix} \mathbf{a} + \mathbf{a}^\top \begin{bmatrix} 0 & E \\ 0 & 0 \end{bmatrix} \mathbf{a} + \mathbf{a}^\top \begin{bmatrix} 0 & 0 \\ 0 & G \end{bmatrix} \mathbf{a} = \mathbf{a}^\top \begin{bmatrix} F & 0 \\ 0 & 0 \end{bmatrix} \mathbf{a} + \mathbf{a}^\top \begin{bmatrix} 0 & 0 \\ 0 & G \end{bmatrix} \mathbf{a} \geq 0$ where we used the fact that a matrix and its transpose has the same quadratic form, and that F and G are positive semidefinite. $\square$

Lemma 4.2. *All diagonal elements of a real positive semidefinite matrix are nonnegative.*

Proof. The assertion is trivial: suppose on the contrary that for a matrix $A \in \mathbb{R}^{n \times n}$ there exists an $i \in \{1, \dots, n\}$ such that $a_{ii} := [A]_{ii} < 0$ holds. Then by denoting the i th unit vector with $\mathbf{e}_i$ we have that $\mathbf{e}_i^\top A \mathbf{e}_i = a_{ii} < 0$, that is, A is not positive semidefinite. $\square$

Lemma 4.3 (Tucker, 1963). *Principal pivot transformations preserve the bisymmetry of the nonbasic part of the basic tableau and the complementarity of the basic tableau.*

This assertion was first stated by Tucker [38, p. 305].

Lemma 4.4. *The terminal tableaus of problem (LCP-QP) are the following complementary tableaus:*

$\oplus$	$\dots$	$\oplus$	$-$
----------	---------	----------	-----

No solution exists

	$\oplus$
	$\vdots$
	$\oplus$
	$\oplus$

A solution has been found

Proof. For any vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}_{\oplus}^N$, the inequality $\mathbf{a}^\top \mathbf{b} \geq 0$ holds. Thus according to the first tableau, the vectors $\mathbf{y}$ and $\mathbf{w}$ satisfying conditions $\mathbf{y}^\top \mathbf{w} = 0$ (complementarity) and $-\mathbf{T}\mathbf{y} + \mathbf{w} = \mathbf{p}$ cannot be nonnegative. This verifies that vectors $\mathbf{y}, \mathbf{w}$ satisfying all three conditions of problem (LCP-QP) do not exist, therefore the problem has no solution.

The second tableau says that all elements of $\mathbf{y}$ and $\mathbf{w}$ in the basis are nonnegative. Since nonbasic variables have to be nonnegative by definition, the nonnegativity criterion of problem (LCP-QP) is satisfied, thus a solution has been found. $\square$

Let $\tilde{\mathbf{p}}$ and $\tilde{T}$ denote the right hand side and the coefficient matrix of the nonbasic part of the current tableau, respectively. Let $\mathcal{I} := \left\{i : \left[\tilde{\mathbf{p}}\right]_i < 0\right\}$. The quadratic criss-cross method of Klafszky and Terlaky is presented in Table 1.

Minimal index QP criss-cross algorithm [24]
Initialization: Let $\mathbf{y} = \mathbf{0}, \mathbf{w} = \mathbf{p}$, thus the algorithm starts from a complementary, but not necessarily feasible basis $\mathbf{w}$. $\mathcal{I} := \left\{i : \left[\tilde{\mathbf{p}}\right]_i < 0\right\}$
While $\mathcal{I} \neq \emptyset$ Do $i := \min \{\mathcal{I}\}$
If $\left[\tilde{T}\right]_{ii} < 0$ then diagonal pivot on position $\left[\tilde{T}\right]_{ii}$.
Else $\mathcal{J} := \left\{j : \left[\tilde{T}\right]_{ij} < 0\right\}$.
If $\mathcal{J} = \emptyset$ then STOP: the problem has no solution.
Else $j := \min \{\mathcal{J}\}$. Exchange pivot on position $\left[\tilde{T}\right]_{ij}$.
Endif
Endif
$\mathcal{I} := \left\{i : \left[\tilde{\mathbf{p}}\right]_i < 0\right\}$
Endwhile
STOP: a solution to the problem has been found.

Table 1: The criss-cross algorithm of Klafszky and Terlaky for quadratic programming

We implemented the algorithm in Matlab, see code section 2. The pseudocode shows us that the algorithm prefers diagonal pivot steps: exchange pivots are performed only if diagonal pivots are not admissible because of a zero in the diagonal. Remark that the algorithm is justified by Lemmas 4.1-4.4.

Theorem 4.5 (Klafszky, Terlaky 1992). *The minimal index quadratic criss-cross algorithm is finite.*

For a proof we refer to [24], or alternatively to Chapter 1.3 of [30].

4.3 The modified criss-cross algorithm for support vector machines

Consider now the primal-dual quadratic optimization problem (SVM-P)-(SVM-D) of the support vector machine. We have seen that their optimality conditions can be rewritten as (MLCP-SVM).⁵ This problem differs from (LCP-QP) in that there is no nonnegativity condition for the entire vector $\mathbf{x}$, just for its certain elements. As usual, we denote the linear system $-M\mathbf{x} + \mathbf{z} = \mathbf{q}$ by

	$\mathbf{x}^\top$	$\mathbf{z}^\top$	
$\mathbf{z}$	$-M$	I	$\mathbf{q}$

that is, the trivial starting basis is $\mathbf{x} = \mathbf{0}$, $\mathbf{z} = \mathbf{q}$. Of course this could not be a solution to problem (MLCP-SVM), since $\mathbf{s} = -\mathbf{1}$ which violates the nonnegativity condition. However, the other two conditions of the problem are satisfied. Thus as we have seen in section 4.2, we would perform principal pivot steps to preserve complementarity, and would study the sign structure of the current basic tableaus to prove that either a solution has been found or problem (MLCP-SVM) has no solution. The difference is that now our terminal tableaus are different from those of Lemma 4.4. To understand the new terminal tableaus, first we write the starting tableau in more detail:

	$\mathbf{v}^\top$	$\mu, \mathbf{u}^\top$	$\mathbf{s}^\top$	$\mathbf{f}^\top$	
$\mathbf{s}$	$-M_{11}$	$-M_{12}$	I_{11}	0	$-\mathbf{1}$
$\mathbf{f}$	$-M_{21}$	$-M_{22}$	0	I_{22}	$\mathbf{0}$
	$\underbrace{\hspace{1.5cm}}_m$	$\underbrace{\hspace{1.5cm}}_{n+1}$	$\underbrace{\hspace{1.5cm}}_m$	$\underbrace{\hspace{1.5cm}}_{n+1}$	

It is apparent that problem (MLCP-SVM) has a solution if

1. nonnegative variables $(\mathbf{s}, \mathbf{v})$ in the basis are nonnegative
2. free variables $(\mu, \mathbf{u})$ take any values
3. fixed variables $\mathbf{f}$ in the basis are equal to zero.

⁵In this section, we choose (MLCP-SVM) over formulation (MLCP-SVM-2) because of its generality: every linearly constrained QP can be reformulated into a problem similar to (MLCP-SVM), while the trick which allowed us to derive (MLCP-SVM-2) in Section 3.4 is particular to the special structure of the optimization problem of the SVM.

Based on this remark, the following theorem characterizes the terminal tableaux of our mixed linear complementarity problem.

Theorem 4.6. *For the simplicity of depiction, let us assume that the first m elements of the basis are of $\mathbf{s}$ or $\mathbf{v}$, while the last $n + 1$ elements are of $\mu, \mathbf{u}$ or $\mathbf{f}$ with variables of $\mu, \mathbf{u}$ preceding variables of $\mathbf{f}$.⁶ Then the terminal tableaux of problem (MLCP-SVM) are the complementary tableaux with the following nonbasic part:*

	$\mathbf{s}, \mathbf{v}, \mathbf{f}$	$\mu, \mathbf{u}$	
$\mathbf{s}, \mathbf{v}$			$\oplus$ $\vdots$ $\oplus$
$\mu, \mathbf{u}$			
$\mathbf{f}$			0 $\vdots$ 0

A solution has been found

	$\mathbf{s}, \mathbf{v}, \mathbf{f}$	$\mu, \mathbf{u}$	
$\mathbf{s}, \mathbf{v}$	$\oplus \quad \dots \quad \oplus$	$0 \quad \dots \quad 0$	$-$
$\mu, \mathbf{u}$			
$\mathbf{f}$			

No solution exists, case a)

	$\mathbf{s}, \mathbf{v}, \mathbf{f}$	$\mu, \mathbf{u}$	
$\mathbf{s}, \mathbf{v}$			
$\mu, \mathbf{u}$			
$\mathbf{f}$	$\oplus \quad \dots \quad \oplus$	$0 \quad \dots \quad 0$	$-$

No solution exists, case b)

	$\mathbf{s}, \mathbf{v}, \mathbf{f}$	$\mu, \mathbf{u}$	
$\mathbf{s}, \mathbf{v}$			
$\mu, \mathbf{u}$			
$\mathbf{f}$	$\ominus \quad \dots \quad \ominus$	$0 \quad \dots \quad 0$	$+$

No solution exists, case c)

⁶Obviously this order of basic variables is artificial and not required to be fulfilled, since the algorithm tracks the index sets of free, nonnegative and fixed variables.

Proof. The tableau characterizing the existence of a solution is an immediate consequence of the different types of variables, while the other tableaus are slight modifications of the no solution case of Lemma 4.4 due to the free variable status of μ and $\mathbf{u}$ and the fixed status of $\mathbf{f}$. $\square$

Remark that the new terminal tableaus imply new pivoting rules for our algorithm. For example, while the original criss-cross algorithm only pivots on negative coefficients, now we may pivot on a positive coefficient, if it is in the column of a free variable.

Let $\tilde{\mathbf{q}}$ and $\tilde{M}$ denote the right hand side and the coefficient matrix of the nonbasic part of the current tableau, respectively. In order to present our modified algorithm, we introduce index sets to keep track of our variables of different types. We denote by $\mathcal{B}$ the index set of basic variables, $\mathcal{L}_f$ the index set of free variables, $\mathcal{L}_n$ the index set of nonnegative variables, and $\mathcal{L}_0$ the index set of fixed variables. $\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < 0, i \in \mathcal{L}_n \cap \mathcal{B} \right\}$ and $\mathcal{I}_0 := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i \neq 0, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}$. The pseudocode of our new algorithm can be found in Table 2.

Let us see how the new algorithm works on the following simple numeric example.

Example 4.7. Suppose that two observations are given: $\mathbf{a}_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top$, $\mathbf{a}_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}^\top$, belonging to different classes, that is, $\mathbf{d} = \begin{bmatrix} -1 & 1 \end{bmatrix}^\top$. Then the optimal separating hyperplane is obviously a line of slope 1 crossing the origin. Thus we expect the following solution: $\mu = 0$, $\mathbf{u} = \begin{bmatrix} -1 & 1 \end{bmatrix}^\top$. Using the notations introduced in Sections 2 and 3.3, we have that

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad D = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad DA = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad M = \left[\begin{array}{cc|ccc} 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ \hline 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \end{array} \right]$$

Thus the starting basic tableau of formulation (MLCP-SVM) is

Modified minimal index criss-cross algorithm for SVM

Initialization: Let $\mathbf{x} = \mathbf{0}$, $\mathbf{z} = \mathbf{q}$, thus the algorithm starts from a complementary, but not necessarily feasible basis $\mathbf{z}$.

$$\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < 0, i \in \mathcal{L}_n \cap \mathcal{B} \right\}, \quad \mathcal{I}_0 := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i \neq 0, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}.$$

While $\mathcal{I}_n \cup \mathcal{I}_0 \neq \emptyset$ **Do** $i := \min \{ \mathcal{I}_n \cup \mathcal{I}_0 \}$

$$\text{If } i \in \mathcal{I}_n \text{ then } \mathcal{J} := \left\{ j : \left[\tilde{M} \right]_{ij} < 0, j \in \mathcal{L}_n \right\} \cup \left\{ j : \left[\tilde{M} \right]_{ij} \neq 0, j \in \mathcal{L}_f \right\}.$$

If $\mathcal{J} = \emptyset$ **then** STOP: the problem has no solution.

Else

If $i \in \mathcal{J}$ **then** diagonal pivot on position $\left[\tilde{M} \right]_{ii}$.

Else $j := \min \{ \mathcal{J} \}$, exchange pivot on position $\left[\tilde{M} \right]_{ij}$.

Endif

Endif

ElseIf $i \in \mathcal{I}_0$ **then**

$$\mathcal{J} := \left\{ j : \left| \left[\tilde{\mathbf{q}} \right]_i + \left[\tilde{M} \right]_{ij} \right| > \left| \left[\tilde{\mathbf{q}} \right]_i \right|, j \in \mathcal{L}_n \right\} \cup \left\{ j : \left[\tilde{M} \right]_{ij} \neq 0, j \in \mathcal{L}_f \right\}.$$

If $\mathcal{J} = \emptyset$ **then** STOP: the problem has no solution.

Else

If $i \in \mathcal{J}$ **then** diagonal pivot on position $\left[\tilde{M} \right]_{ii}$.

Else $j := \min \{ \mathcal{J} \}$, exchange pivot on position $\left[\tilde{M} \right]_{ij}$.

Endif

Endif

Endif

$$\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < 0, i \in \mathcal{L}_n \cap \mathcal{B} \right\}, \quad \mathcal{I}_0 := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i \neq 0, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}.$$

Endwhile

STOP: a solution to the problem has been found.

Table 2: Modified quadratic criss-cross algorithm

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
s_1	0	0	1	1	0	1	0	0	0	0	-1
s_2	0	0	-1	0	-1	0	1	0	0	0	-1
f_1	-1	1	0	0	0	0	0	1	0	0	0
f_2	-1	0	0	-1	0	0	0	0	1	0	0
f_3	0	1	0	0	-1	0	0	0	0	1	0

The minimal index rule says that we have to pivot in the row of s_1 . Note that there are only nonnegative elements in this row, which would implicate a terminal tableau in the case of the original criss-cross algorithm. However, some positive elements in the row correspond to free variables (μ and u_1), therefore by Theorem 4.6 our modified algorithm does not terminate here.

The first element of the row is a zero, thus the preferred diagonal pivot is not admissible. Hence, we perform an exchange pivot in the column of μ due to the minimal index rule. The first step of the exchange pivot gives us the following basic tableau.

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	1	0	1	0	0	0	0	-1
s_2	0	0	0	1	-1	1	1	0	0	0	-2
f_1	-1	1	0	0	0	0	0	1	0	0	0
f_2	-1	0	0	-1	0	0	0	0	1	0	0
f_3	0	1	0	0	-1	0	0	0	0	1	0

This basic tableau is not complementary, since μ entered the basis, but s_1 left it instead of f_1 , the complementary pair of μ . In order to restore complementarity, we have to execute the second step of the exchange pivot, that is, bringing the variable v_1 into the basic in place of f_1 . The resulting tableau is

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	1	0	1	0	0	0	0	-1
s_2	0	0	0	1	-1	1	1	0	0	0	-2
v_1	1	-1	0	0	0	0	0	-1	0	0	0
f_2	0	-1	0	-1	0	0	0	-1	1	0	0
f_3	0	1	0	0	-1	0	0	0	0	1	0

Complementarity is restored. Now the first row corresponds to the free variable μ , thus the -1 on the right hand side does not contradict the conditions of problem (MLCP-SVM), therefore further pivot steps are not needed in this row. However, the nonnegative variable s_2 is still negative, hence the tableau is not terminal, so we pivot in the second row. The corresponding diagonal element of the nonbasic part of the current tableau is 0, thus the diagonal pivot step is again infeasible. Therefore, exchange pivot is performed. After the first step we get

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	0	1	0	-1	0	0	0	1
u_1	0	0	0	1	-1	1	1	0	0	0	-2
v_1	1	-1	0	0	0	0	0	-1	0	0	0
f_2	0	-1	0	0	-1	1	1	-1	1	0	-2
f_3	0	1	0	0	-1	0	0	0	0	1	0

and the second step yields

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	0	1	0	-1	0	0	0	1
u_1	0	0	0	1	-1	1	1	0	0	0	-2
v_1	1	0	0	0	1	-1	-1	0	-1	0	2
v_2	0	1	0	0	1	-1	-1	1	-1	0	2
f_3	0	0	0	0	-2	1	1	-1	1	1	-2

This basic tableau is complementary after the second step of the exchange pivot. Theorem 4.6 tells us that this tableau is not terminal, because the fixed variable f_3 does not equal to 0. Thus we pivot in the last row. Fortunately, now the preferred diagonal pivot is permitted.

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	0	0	1/2	-1/2	-1/2	1/2	1/2	0
u_1	0	0	0	1	0	1/2	1/2	1/2	-1/2	-1/2	-1
v_1	1	0	0	0	0	-1/2	-1/2	-1/2	-1/2	1/2	1
v_2	0	1	0	0	0	-1/2	-1/2	1/2	-1/2	1/2	1
u_2	0	0	0	0	1	-1/2	-1/2	1/2	-1/2	-1/2	1

This is a terminal tableau: every nonnegative variable admits a nonnegative value, and every fixed variable is equal to zero, in this case because none of them are in the basis. The optimal solution is therefore $\mu = 0$, $\mathbf{u} = \begin{bmatrix} -1 & 1 \end{bmatrix}^\top$, which corresponds to our expectations.

The following example depicts us formulation (MLCP-SVM-2), while a numerical example of a problem with no solution can be found in Appendix A.4.

Example 4.8. Consider the observations of Example 4.7. Following the results in Section 3.4, the coefficient matrix of formulation (MLCP-SVM-2) is

$$\widehat{M} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

which yields the starting basic tableau

	v_1	v_2	μ	s_1	s_2	f	
s_1	-1	0	1	1	0	0	-1
s_2	0	-1	-1	0	1	0	-1
f	-1	1	0	0	0	1	0

Diagonal pivot in the first row gives

	v_1	v_2	μ	s_1	s_2	f	
v_1	1	0	-1	-1	0	0	1
s_2	0	-1	-1	0	1	0	-1
f	0	1	-1	-1	0	1	1

Now we have to perform a diagonal pivot in the second row, which yields

	v_1	v_2	μ	s_1	s_2	f	
v_1	1	0	-1	-1	0	0	1
v_2	0	1	1	0	-1	0	1
f	0	0	-2	-1	1	1	0

This is a terminal tableau showing that a solution has been found. Because μ is not in the basis, it is zero by definition. To obtain $\mathbf{u}$, we have to read the optimal values of the Lagrange multiplier vector $\mathbf{v} = \begin{bmatrix} 1 & 1 \end{bmatrix}^\top$ from the tableau, and substitute it into formula (11) along with $DA = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$:

$$\mathbf{u} = (DA)^\top \mathbf{v} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

which is what we got in Example 4.7.

Examples 4.7 and 4.8 indicate that with slight modifications, our algorithm can solve general linearly constrained quadratic optimization problems (formulation (MLCP-SVM)),

while simultaneously it can be made considerably more efficient if we exploit the special structure of the optimization problem of the SVM (formulation (MLCP-SVM-2)).

4.4 Computational results

In this section, we apply the Matlab implementation of our new algorithm (see Table 2 and code sections 3, 4 and 5 in Appendix A.5) to real-world data and make some remarks on the numerical stability of our implementation. The datasets, described in Table 3 were downloaded from the UCI Machine Learning Repository [18], and computations were carried out on an Intel Core i5-6500 3.20GHz processor and Matlab version 2019b.

Name	Description	Source
wisconsin ⁷	Breast Cancer Dataset from Wisconsin	LINK , [42], [26], [41], [4]
hepatitis	Hepatitis Dataset	LINK
cleveland ⁸	Heart Disease Dataset from Cleveland	LINK
hungarian ⁹	Heart Disease Dataset from Hungary	LINK
sonar ¹⁰	Connectionist Bench (Sonar, Mines vs. Rocks) Dataset	LINK

Table 3: Description of datasets

The results of the naïve implementation of Algorithm 2 can be found in Table 4. We report the dimensions of the datasets, the runtime averaged over 100 runs, and the number of diagonal and exchange pivot steps made in a run by the algorithm for formulations (MLCP-SVM) and (MLCP-SVM-2) separately. Note that an exchange pivot always requires two transformations, thus the number of exchange pivot steps performed have to be even.

⁷This breast cancer database was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

⁸This database was obtained from the V.A. Medical Center, Long Beach and Cleveland Clinic Foundation from Robert Detrano, M.D., Ph.D.

⁹This database was obtained from the Hungarian Institute of Cardiology, Budapest, from Andras Janosi, M.D.

¹⁰The dataset is from Terry Sejnowski, now at the Salk Institute and the University of California at San Deigo. The dataset was developed in collaboration with R. Paul Gorman of Allied-Signal Aerospace Technology Center.

It is often recommended for applied researchers to standardize¹¹ variables in order to avoid numerical difficulties during the calculation (see e.g. [20]). Because of this, we compare the runtime of our algorithm on standardized and nonstandardized data for each dataset and each formulation, and report whether or not standardizing made calculations faster. Consequently, the reported runtimes and number of pivot steps correspond to the quicker method.

	wisconsin	hepatitis	cleveland	hungarian	sonar
observations (m)	683	80	297	294	208
variables (n)	9	19	13	13	60
Formulation (MLCP-SVM)					
runtime (10^{-2} sec)	211.45	—	31.03	—	0.25
diagonal pivot steps	0	—	0	—	0
exchange pivot steps	268	—	354	—	2
faster method	n.	—	n.	—	n.
Formulation (MLCP-SVM-2)					
runtime (10^{-2} sec)	—	5.02	—	—	3.56
diagonal pivot steps	—	182	—	—	64
exchange pivot steps	—	0	—	—	0
faster method	—	n.	—	—	n.

s. is short for standardized; n. stands for nonstandardized.

Runtime was averaged over 100 runs.

Table 4: Computational results, $\epsilon = 0$

The hyphens in Table 4 indicate that the algorithm did not converge in those cases, that is, cycling or pivoting on a zero occurred. The reason for these was that pivoting could result in the nonbasic part of the current tableau ($\tilde{M}$) losing its bisymmetry numerically. If the matrix M is not bisymmetric, then of course the exchange pivot could force its second step to pivot on a zero. Similarly, bisymmetry plays a heavy role in the proof of the finiteness of Klafszky and Terlaky’s algorithm [24].

To combat this numerical instability, we used a numerical trick and ‘gave the algorithm an epsilon of room’. That is, a positive decision threshold ϵ can be defined by the user,

¹¹Standardization is subtracting the empirical mean and dividing the result by the empirical standard deviation for each value of each variable.

and the algorithm ignores differences smaller than ϵ when comparing numbers. Because of this method, numerical trash is less likely to influence pivoting rules. Of course the naïve implementation, of which results are reported in Table 4, would correspond to the case $\epsilon = 0$. The numerically more stable variant of our new algorithm is presented in Table 5.

Modified minimal index criss-cross algorithm for SVM, numerical

Initialization: Let $\mathbf{x} = \mathbf{0}$, $\mathbf{z} = \mathbf{q}$, thus the algorithm starts from a complementary, but not necessarily feasible basis $\mathbf{z}$. Let $\epsilon > 0$.

$$\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < -\epsilon, i \in \mathcal{L}_n \cap \mathcal{B} \right\}, \quad \mathcal{I}_0 := \left\{ i : \left| \left[\tilde{\mathbf{q}} \right]_i \right| > \epsilon, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}.$$

While $\mathcal{I}_n \cup \mathcal{I}_0 \neq \emptyset$ **Do** $i := \min \{ \mathcal{I}_n \cup \mathcal{I}_0 \}$

If $i \in \mathcal{I}_n$ **then** $\mathcal{J} := \left\{ j : \left[\tilde{M} \right]_{ij} < -\epsilon, j \in \mathcal{L}_n \right\} \cup \left\{ j : \left| \left[\tilde{M} \right]_{ij} \right| > \epsilon, j \in \mathcal{L}_f \right\}$.

If $\mathcal{J} = \emptyset$ **then** STOP: the problem has no solution.

Else

If $i \in \mathcal{J}$ **then** diagonal pivot on position $\left[\tilde{M} \right]_{ii}$.

Else $j := \min \{ \mathcal{J} \}$, exchange pivot on position $\left[\tilde{M} \right]_{ij}$.

Endif

Endif

ElseIf $i \in \mathcal{I}_0$ **then**

$$\mathcal{J} := \left\{ j : \left| \left[\tilde{\mathbf{q}} \right]_i + \left[\tilde{M} \right]_{ij} \right| > \left| \left[\tilde{\mathbf{q}} \right]_i \right| + \epsilon, j \in \mathcal{L}_n \right\} \cup \left\{ j : \left| \left[\tilde{M} \right]_{ij} \right| > \epsilon, j \in \mathcal{L}_f \right\}.$$

If $\mathcal{J} = \emptyset$ **then** STOP: the problem has no solution.

Else

If $i \in \mathcal{J}$ **then** diagonal pivot on position $\left[\tilde{M} \right]_{ii}$.

Else $j := \min \{ \mathcal{J} \}$, exchange pivot on position $\left[\tilde{M} \right]_{ij}$.

Endif

Endif

Endif

$$\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < -\epsilon, i \in \mathcal{L}_n \cap \mathcal{B} \right\}, \quad \mathcal{I}_0 := \left\{ i : \left| \left[\tilde{\mathbf{q}} \right]_i \right| > \epsilon, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}.$$

Endwhile

STOP: a solution to the problem has been found.

Table 5: Modified quadratic criss-cross algorithm, numerically more stable version

The computational results corresponding to the numerically more stable version of our modified algorithm with $\epsilon = 10^{-6}$ can be found in Table 6. We remark that while a considerable improvement can be observed compared to the results in Table 4, even this more sophisticated implementation failed to converge in all cases. It should also be noted that out of the 4 cases where the algorithm did converge on both formulations, 3 resulted in the problem (MLCP-SVM-2) being superior in terms of runtime and the number of pivot steps performed. This might lead us to think that formulation (MLCP-SVM-2) is indeed computationally favorable – provided it converges.

	wisconsin	hepatitis	cleveland	hungarian	sonar
observations (m)	683	80	297	294	208
variables (n)	9	19	13	13	60
Formulation (MLCP-SVM)					
runtime (10^{-2} sec)	181.22	28.75	55.05	12.46	0.23
diagonal pivot steps	0	365	0	0	0
exchange pivot steps	266	638	674	170	2
faster method	s.	n.	s.	s.	s.
Formulation (MLCP-SVM-2)					
runtime (10^{-2} sec)	115.22	5.34	–	5.38	3.23
diagonal pivot steps	85	184	–	46	64
exchange pivot steps	42	0	–	12	0
faster method	s.	n.	–	s.	n.

s. is short for standardized; n. stands for nonstandardized.

Runtime was averaged over 100 runs.

Table 6: Computational results, $\epsilon = 10^{-6}$

We believe that further study on the computational properties of the algorithm on the two formulations along with developing a numerically even more stable implementation could serve as an interesting path for future research.

5 Open problems

This section is concerned with some open problems related to the topic of the paper, and as such, suggests possible paths for future research. First we present some well-known generalizations of the linear support vector machine, then state a conjecture which, if true, would allow a simpler version of our new criss-cross algorithm to be formulated.

5.1 Generalizations

5.1.1 Soft Margin SVM

As it was mentioned in Section 1, the premise that sets of data can be perfectly separated is a rather crude assumption almost never met in practice. Therefore real-world applications require an extension of the framework presented in Section 2, that is, a relaxation of Assumption 2.2. The most widely accepted solution is the so-called Soft Margin Hyperplane method, which was introduced by Cortes and Vapnik [11], although Smola and Schölkopf ([35] and [33, p. 204]) note that the idea can be traced back to Bennett and Mangasarian [4]. The rest of this section formulates the Soft Margin Hyperplane framework.

We have seen that the linear separability assumption 2.2 can be written as (3) if the separating hyperplane is given in its canonical form. Cortes and Vapnik [11] define slack variables $\kappa_j \geq 0$, $j = 1, \dots, m$ such that the separability inequality (3) would change to

$$\begin{aligned} \mathbf{u}^\top \mathbf{a}_j - \nu &\geq 1 - \kappa_j && \text{if } \mathbf{a}_j \in \mathcal{J}_1 \\ \mathbf{u}^\top \mathbf{a}_j - \nu &\leq -1 + \kappa_j && \text{if } \mathbf{a}_j \in \mathcal{J}_2 \end{aligned}$$

and accordingly, (4) would be replaced by

$$d_j (\mathbf{u}^\top \mathbf{a}_j - \nu) \geq 1 - \kappa_j \quad (j = 1, \dots, m).$$

Remark 5.1. Note that if κ_j is strictly positive, then $\lim_{\sigma \rightarrow 0} \kappa_j^\sigma = 1$, thus for a sufficiently small $\sigma > 0$, the functional $\sum_{j=1}^m \kappa_j^\sigma$ describes the number of observations misclassified.

Let us now consider a monotone convex function $\Lambda : \mathbb{R}_\oplus \rightarrow \mathbb{R}$ and a scalar $\rho \in \mathbb{R}$. Then the optimization problem ($j = 1, \dots, m$)

$$\left. \begin{aligned} \min_{\mathbf{u}, \mu} \quad & \frac{1}{2} \mathbf{u}^\top \mathbf{u} + \rho \Lambda \left(\sum_{j=1}^m \varkappa_j^\sigma \right) \\ & d_j (\mathbf{u}^\top \mathbf{a}_j - \nu) \geq 1 - \varkappa_j \\ & \varkappa_j \geq 0 \end{aligned} \right\} \quad (\text{SVM-NP})$$

for an appropriate choice of hyperparameters ρ and σ determines a separating hyperplane which has the minimal number of observations misclassified and has the widest margin around it with respect to the correctly classified vectors.

Problem (SVM-NP), however impressive, never made it into practice, as Cortes and Vapnik [11] note that it is NP-complete. However, if $\sigma = 1$, then the problem

$$\left. \begin{aligned} \min_{\mathbf{u}, \mu} \quad & \frac{1}{2} \mathbf{u}^\top \mathbf{u} + \rho \Lambda \left(\sum_{j=1}^m \varkappa_j \right) \\ & d_j (\mathbf{u}^\top \mathbf{a}_j - \nu) \geq 1 - \varkappa_j \\ & \varkappa_j \geq 0 \end{aligned} \right\} \quad (\text{SVM-SM})$$

with the additional condition $\Lambda(0) = 0$ has an efficient method for finding its solution. This problem seeks a separating hyperplane which minimizes the so-called sum of deviations instead of the number of misclassified observations. Cortes and Vapnik [11] term the solution to problem (SVM-SM) the soft margin hyperplane. Note that this problem is convex by a similar argument to that of the proof of Theorem 3.1, though it is not quadratic except for some special cases, like if the function Λ is the identity or the square function.

The literature on the Soft Margin Hyperplane model and on other generalizations of the linear SVM is immense – for a recent review, consult [9]. Despite this, to the best of our knowledge, mixed linear complementarity problems possibly arising from special cases of (SVM-SM) are yet to be studied, along with the generalization of our modified criss-cross algorithm to these problems.

5.1.2 Nonlinear SVM

By utilizing the powerful and elegant theory of reproducing kernel Hilbert spaces and other topics of functional analysis, the nonlinear generalization of the Support Vector Machine

made it possible for the method to be incredibly effective in a number of applications, such as handwriting recognition (see e.g. [16]). In this section, we follow the structure of [36] to establish the landmark idea of Boser, Guyon and Vapnik [6].

We have seen in Section 2 that our observations $\mathbf{a}_j$ ($j = 1, \dots, m$) can be identified with elements of the Euclidean vector space $\mathbb{R}^n$. Let us define a so-called feature map $\Phi : \mathbb{R}^n \rightarrow H$ where the feature space $(H, \langle \cdot, \cdot \rangle)$ is a Hilbert space over $\mathbb{K}$, where $\mathbb{K}$ is either $\mathbb{R}$ or $\mathbb{C}$. Then the linear SVM is applied to the space H , i.e. we search for the normal vector $\mathbf{u} \in H$ and scalar $\nu \in \mathbb{R}$ such that the canonical hyperplane $\mathcal{H}(\mathbf{u}, \nu) \subseteq H$ linearly separates the vectors $\Phi(\mathbf{a}_j)$ with the widest¹² margin. The possible nonlinear nature of the feature map Φ makes it possible for the separating boundary in $\mathbb{R}^n$ corresponding to the optimal separating hyperplane in H to be nonlinear.

It can be argued that computation is quite inconvenient in possibly infinite dimensional Hilbert spaces. The fundamental concept of kernels allows us to represent inner products of abstract spaces as efficiently computable functions. The following definition formalizes the notion of kernels. We denote the complex conjugate of a complex number $z \in \mathbb{C}$ by $\bar{z}$.

Definition 5.2 (kernel).

Let X be a nonempty set. If there exists a Hilbert space $(H, \langle \cdot, \cdot \rangle)$ over $\mathbb{K}$ and a map $\Phi : X \rightarrow H$ such that for all $x, y \in X$ we have

$$\kappa(x, y) = \overline{\langle \Phi(x), \Phi(y) \rangle}$$

then the function $\kappa : X \times X \rightarrow \mathbb{K}$ is called a *kernel* on X with respect to the feature map Φ and feature space H .

Properties of kernels are now well understood. For example, positive definite kernels form a convex cone closed in the topology of pointwise convergence (e.g. [5, Property 1.11, p. 69], or alternatively, [36, Lemma 4.5 and Corollary 4.17] for the case $\mathbb{K} = \mathbb{R}$) and positive semidefiniteness and symmetricity characterizes kernels mapping into $\mathbb{R}$ [36, Theorem 4.16]. Another important result is Mercer's Theorem (see e.g. [28, p. 245-246]), which allows us to construct explicit series representations for continuous kernels on compact domains [36]. These theoretical results, among others, enabled the creation and study of

¹²Of course with respect to the metric induced by $\langle \cdot, \cdot \rangle$, the inner product of H .

particular kernel functions which became astonishingly useful and effective in practical applications.

It remains to show how kernel theory helps tackle the problem of finding a separating hyperplane in possibly infinite dimensional Hilbert spaces. The key remark is noting that in the dual problem (SVM-D), the observations appear only in the form of inner products. Indeed, it is easy to see that the dual of the SVM in the feature space H only depends on the feature map Φ via the inner product $\langle \Phi(\cdot), \Phi(\cdot) \rangle$. Thus kernels, which represent such inner products, can be used to compute the optimal separating boundary in $\mathbb{R}^n$ without knowing the function Φ itself. This observation, popularly known as the ‘kernel trick’, was first stated explicitly by Schölkopf, Smola and Müller in [31] and [32], although the idea can be traced back to [1]. Investigating whether our new criss-cross algorithm could be generalized to seek nonlinear boundaries would be an interesting path for future research.

5.2 Simplification of the modified algorithm

We hypothesize that in Theorem 4.6 the tableaux $b)$ and $c)$ indicating that no solution exists never occur. This surmise is formalized in the following conjecture.

Conjecture 5.3. *Rows of a basic tableau corresponding to fixed variables cannot verify the inconsistency of problem (MLCP-SVM).*

We believe that a proof of this assertion contains many cases, and while we showed that the statement follows in some cases, covering all possibilities is left for future research. This conjecture is interesting, because if proved, then a considerably simpler version of our modified algorithm can be justified. The pseudocode of this potential algorithm is presented in Table 7.

6 Summary

The Support Vector Machine is one of the most popular statistical learning methods, serving as an industry standard for solving many real-world problems. On the other hand, criss-cross methods are theoretically important algorithms as they stimulated research on many subfields of optimization. Our paper is concerned with the intersection of these two methods, and it is both theoretical and practical in nature.

Modified minimal index criss-cross algorithm for SVM, simplified

Initialization: Let $\mathbf{x} = \mathbf{0}$, $\mathbf{z} = \mathbf{q}$, thus the algorithm starts from a complementary, but not necessarily feasible basis $\mathbf{z}$.

$$\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < 0, i \in \mathcal{L}_n \cap \mathcal{B} \right\}, \quad \mathcal{I}_0 := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i \neq 0, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}.$$

While $\mathcal{I}_n \cup \mathcal{I}_0 \neq \emptyset$ **Do** $i := \min \{ \mathcal{I}_n \cup \mathcal{I}_0 \}$

If $\left[\tilde{M} \right]_{ii} < 0$ **then** diagonal pivot on position $\left[\tilde{M} \right]_{ii}$.

ElseIf $i \in \mathcal{I}_n$ **then** $\mathcal{J} := \left\{ j : \left[\tilde{M} \right]_{ij} < 0, j \in \mathcal{L}_n \right\} \cup \left\{ j : \left[\tilde{M} \right]_{ij} \neq 0, j \in \mathcal{L}_f \right\}$.

If $\mathcal{J} = \emptyset$ **then** STOP: the problem has no solution.

Else $j := \min \{ \mathcal{J} \}$, exchange pivot on position $\left[\tilde{M} \right]_{ij}$.

Endif

ElseIf $i \in \mathcal{I}_0$ **then** $\mathcal{J} := \left\{ j : \left[\tilde{M} \right]_{ij} \neq 0 \right\}$, $j := \min \{ \mathcal{J} \}$.

Exchange pivot on position $\left[\tilde{M} \right]_{ij}$.

Endif

$$\mathcal{I}_n := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i < 0, i \in \mathcal{L}_n \cap \mathcal{B} \right\}, \quad \mathcal{I}_0 := \left\{ i : \left[\tilde{\mathbf{q}} \right]_i \neq 0, i \in \mathcal{L}_0 \cap \mathcal{B} \right\}.$$

Endwhile

STOP: a solution to the problem has been found.

Table 7: Modified quadratic criss-cross algorithm, simplified version

Our theoretical results include two distinct derivations of the mixed linear complementarity problem of the SVM and presenting an alternative formulation by noting the special structure of the problem. The main contribution of this paper is arguably our new algorithm variant, which is a generalization of the quadratic criss-cross algorithm of Klafszky and Terlaky in the sense that it operates on mixed linear complementarity problems and thus can handle free and fixed variables beside nonnegative ones. In terms of practical applications, we implemented our algorithm and tested its numerical performance on datasets of real-world classification problems.

There are many interesting paths for future research, for example extending our algorithm to existing generalizations of the linear SVM, improving the numerical stability of our implementation, investigating in a thorough simulation study the computational properties

of the two distinct MLCP problems of the support vector machine, and generalizing our modified algorithm to other index selection rules with freedom on the pivot position selection. The last direction is particularly compelling as it gives the possibility to avoid numerically instable pivots compared to the minimal index rule.

References

- [1] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer. Theoretical foundation of potential functions method in pattern recognition. *Avtomat. i Telemekh*, 25(6):917–936, 1964.
- [2] A. A. Akkeleş, L. Balogh, and T. Illés. New variants of the criss-cross method for linearly constrained convex quadratic programming. *European Journal of Operational Research*, 157(1):74–86, 2004.
- [3] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.
- [4] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization methods and software*, 1(1):23–34, 1992.
- [5] C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic analysis on semigroups: theory of positive definite and related functions*, volume 100. Springer, 1984.
- [6] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92*, page 144–152, New York, NY, USA, 1992. Association for Computing Machinery.
- [7] C. J. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.
- [8] Y.-Y. Chang. Least-index resolution of degeneracy in linear complementarity problems. Technical report, Department of Operations Research, Stanford University, 1979.
- [9] V. K. Chauhan, K. Dahiya, and A. Sharma. Problem formulations and solvers in linear svm: a review. *Artificial Intelligence Review*, 52(2):803–855, 2019.
- [10] A. Y. Chervonenkis. *Early History of Support Vector Machines*, pages 13–20. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [11] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

- [12] R. W. Cottle, G. B. Dantzig, et al. Complementary pivot theory of mathematical programming. *Mathematics of the decision sciences, part*, 1(11), 1968.
- [13] R. W. Cottle, J.-S. Pang, and R. E. Stone. *The linear complementarity problem*. SIAM, 2009.
- [14] Z. Csizmadia and T. Illés. New criss-cross type algorithms for linear complementarity problems with sufficient matrices. *Optimization Methods and Software*, 21(2):247–266, 2006.
- [15] G. B. Dantzig. Origins of the simplex method. In *A history of scientific computing*, pages 141–151. 1990.
- [16] D. DeCoste and B. Schölkopf. Training invariant support vector machines. *Machine learning*, 46(1):161–190, 2002.
- [17] D. den Hertog, C. Roos, and T. Terlaky. The linear complementarity problem, sufficient matrices, and the criss-cross method. *Linear Algebra and Its Applications*, 187:1–14, 1993.
- [18] D. Dua and C. Graff. UCI machine learning repository, 2017.
- [19] K. Fukuda. Introduction to optimization, 2020.
- [20] C.-W. Hsu, C.-C. Chang, C.-J. Lin, et al. A practical guide to support vector classification, 2003.
- [21] T. Illés. Lineáris optimalizálás elmélete és pivot algoritmusai. 2013.
- [22] T. Illés and T. Terlaky. Pivot versus interior point methods: Pros and cons. *European Journal of Operational Research*, 140(2):170–190, 2002.
- [23] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311, 1984.
- [24] E. Klafszky and T. Terlaky. Some generalizations of the criss-cross method for quadratic programming. *Optimization*, 24(1-2):127–139, 1992.
- [25] V. Klee and G. J. Minty. How good is the simplex algorithm. *Inequalities*, 3(3):159–175, 1972.

- [26] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1990.
- [27] I. Maros. *Computational techniques of the simplex method*, volume 61. Springer Science & Business Media, 2002.
- [28] F. Riesz and B. S. Nagy. *Functional analysis*. Dover Publications, 1990.
- [29] C. Roos. An exponential example for terlaky’s pivoting rule for the criss-cross simplex method. *Mathematical Programming*, 46(1):79–84, 1990.
- [30] C. Roos, T. Terlaky, and E. Klerk. *Nonlinear Optimization (WI3 031)*. 2002.
- [31] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In *International conference on artificial neural networks*, pages 583–588. Springer, 1997.
- [32] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.
- [33] B. Schölkopf, A. J. Smola, F. Bach, et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.
- [34] J. Shawe-Taylor and S. Sun. A review of optimization methodologies in support vector machines. *Neurocomputing*, 74(17):3609–3618, 2011.
- [35] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.
- [36] I. Steinwart and A. Christmann. *Support vector machines*. Springer Science & Business Media, 2008.
- [37] T. Terlaky. A convergent criss-cross method. *Optimization*, 16(5):683–690, 1985.
- [38] A. W. Tucker. Principal pivotal transforms of square matrices. In *SIAM Review*, volume 5, page 305. SIAM Publications, 1963.
- [39] V. N. Vapnik. *Statistical Learning Theory*. Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control. Wiley, 1998.
- [40] Z. Wang. A finite conformal-elimination free algorithm over oriented matroid programming. *Chin. Ann. of Math.*, 8:120–125, 1987.

- [41] W. Wolberg, O. Mangasarian, T. Coleman, and Y. Li. Pattern recognition via linear programming: Theory and application to medical diagnosis. In *Large-scale numerical optimization*, pages 22–30. SIAM Publications, Citeseer, 1990.
- [42] W. H. Wolberg and O. L. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the National Academy of Sciences*, 87(23):9193–9196, 1990.
- [43] S. Zhang. New variants of finite criss-cross pivot algorithms for linear programming. *European Journal of Operational Research*, 116(3):607–614, 1999.
- [44] S. Zionts. The criss-cross method for solving linear programming problems. *Management Science*, 15(7):426–445, 1969.

A Appendix

A.1 Euclidean geometry

Definition A.1 (Hyperplane, Open halfspace, Normal vector).

Suppose that $\mathbf{u} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ and $\mu \in \mathbb{R}$ are given. An (*affine*) *hyperplane* is the set

$$\mathcal{H}(\mathbf{u}, \mu) := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{u}^\top \mathbf{x} - \mu = 0\}$$

and *open (affine) half-spaces* are sets given by

$$\mathcal{F}_>(\mathbf{u}, \mu) := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{u}^\top \mathbf{x} - \mu > 0\} \quad \text{or} \quad \mathcal{F}_<(\mathbf{u}, \mu) := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{u}^\top \mathbf{x} - \mu < 0\}$$

We call $\mathbf{u}$ the *normal vector* to the hyperplane $\mathcal{H}(\mathbf{u}, \mu)$.

Remark A.2. A hyperplane $\mathcal{H}(\mathbf{u}, \mu) \subseteq \mathbb{R}^n$ defines a partition $\{\mathcal{H}(\mathbf{u}, \mu), \mathcal{F}_>(\mathbf{u}, \mu), \mathcal{F}_<(\mathbf{u}, \mu)\}$ of the vector space $\mathbb{R}^n$.

Definition A.3 (Separation).

Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{R}^n$ be nonempty sets and consider the partition $\{\mathcal{H}(\mathbf{u}, \mu), \mathcal{F}_>(\mathbf{u}, \mu), \mathcal{F}_<(\mathbf{u}, \mu)\}$ defined by $\mathcal{H}(\mathbf{u}, \mu) \subseteq \mathbb{R}^n$. If

$$\mathcal{F}_<(\mathbf{u}, \mu) \cap \mathcal{C}_1 = \mathcal{F}_>(\mathbf{u}, \mu) \cap \mathcal{C}_2 = \emptyset \quad \text{or} \quad \mathcal{F}_>(\mathbf{u}, \mu) \cap \mathcal{C}_1 = \mathcal{F}_<(\mathbf{u}, \mu) \cap \mathcal{C}_2 = \emptyset$$

then we say that the hyperplane $\mathcal{H}(\mathbf{u}, \mu)$ *separates* the sets $\mathcal{C}_1$ and $\mathcal{C}_2$.

Definition A.4 (Strict separation).

The hyperplane $\mathcal{H}(\mathbf{u}, \mu) \subseteq \mathbb{R}^n$ *strictly separates* the sets $\mathcal{C}_1 \subseteq \mathbb{R}^n$ and $\mathcal{C}_2 \subseteq \mathbb{R}^n$ if it separates them and

$$\mathcal{H}(\mathbf{u}, \mu) \cap \mathcal{C}_1 = \mathcal{H}(\mathbf{u}, \mu) \cap \mathcal{C}_2 = \emptyset$$

Definition A.5 (Metric induced by Euclidean norm).

The function $\delta : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is called a *metric* or *distance function* if

$$\delta(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$$

Definition A.6 (distance between point and a set). Let $\mathcal{C} \subset \mathbb{R}^n$ be a nonempty set and $\mathbf{x} \in \mathbb{R}^n$. The distance of point $\mathbf{x}$ from the set $\mathcal{C}$ is

$$\delta(\mathbf{x}, \mathcal{C}) := \inf_{\mathbf{y} \in \mathcal{C}} \delta(\mathbf{x}, \mathbf{y})$$

A.2 Convex optimization

Consider the following convex nonlinear optimization problem

$$\left. \begin{array}{l} \min_{\mathbf{x}} f(\mathbf{x}) \\ g_j(\mathbf{x}) \leq 0 \\ \mathbf{x} \in \mathcal{A} \end{array} \right\} \quad (\text{CO})$$

where $\mathcal{A} \subseteq \mathbb{R}^n$ is a convex set and $f, g_j : \mathcal{A} \rightarrow \mathbb{R}$ ($j = 1, \dots, m$) are convex and continuously differentiable functions.

Definition A.7 (Lagrangian function, Lagrange multiplier).

The *Lagrangian function* of the problem (CO) is a function $\mathcal{L} : \mathcal{A} \times \mathbb{R}_{\oplus}^m \rightarrow \mathbb{R}$ such that $\forall \mathbf{x} \in \mathcal{A}, \forall \boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_m]^\top \in \mathbb{R}^m$

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) := f(\mathbf{x}) + \sum_{j=1}^m \lambda_j g_j(\mathbf{x})$$

The vector $\boldsymbol{\lambda}$ is the *Lagrange multiplier*.

Definition A.8 (Lagrange dual problem). We denote $\psi(\boldsymbol{\lambda}) := \inf_{\mathbf{x} \in \mathcal{A}} \{\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})\}$. The optimization problem

$$\left. \begin{array}{l} \sup_{\boldsymbol{\lambda}} \psi(\boldsymbol{\lambda}) \\ \boldsymbol{\lambda} \geq \mathbf{0} \end{array} \right\} \quad (\text{LD})$$

is the *Lagrange dual* of the convex optimization problem (CO).

Definition A.9 (Gradient).

The gradient of a continuously differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a vector-valued function $\nabla f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ such that for all $\mathbf{x} \in \mathbb{R}^n$ $\nabla f(\mathbf{x}) = \left[\frac{\partial}{\partial x_1} f(\mathbf{x}), \dots, \frac{\partial}{\partial x_n} f(\mathbf{x}) \right]^\top$.

Theorem A.10 (Karush-Kuhn-Tucker).

Assume that $\mathcal{A} = \mathbb{R}^n$, and g_j ($j = 1, \dots, m$) are linear functions¹³. The vector $\mathbf{x}^*$ is an optimal solution of the problem (CO) if and only if the following conditions are satisfied.

¹³We chose to state the theorem with this regularity condition because the optimization problem of the SVM trivially satisfies it. For other regularity conditions and their relations, see e.g. [3]

$$\text{Primal feasibility:} \quad g_j(\mathbf{x}^*) \leq 0, \quad j = 1, \dots, m \quad (13a)$$

$$\text{Dual feasibility:} \quad \lambda_j \geq 0, \quad j = 1, \dots, m \quad (13b)$$

$$\text{Complementarity:} \quad \sum_{j=1}^m \lambda_j g_j(\mathbf{x}^*) = 0 \quad (13c)$$

$$\text{Stationarity:} \quad \nabla f(\mathbf{x}^*) + \sum_{j=1}^m \lambda_j \nabla g_j(\mathbf{x}^*) = \mathbf{0} \quad (13d)$$

Proof. The theorem is a version of the famous Karush-Kuhn-Tucker theorem, see e.g. [30, Corollaries 2.73-2.75]. $\square$

A.3 Pivoting

Let X be a vector space over $\mathbb{R}$. Consider the finite set of vectors $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n\} \subseteq X$ and denote its index set by $\mathcal{J} = \{1, 2, \dots, n\}$. Let $\mathcal{B}$ denote the index set of a basis and $\mathcal{N} := \mathcal{J} \setminus \mathcal{B}$. Then of course

$$\mathbf{a}_j = \sum_{i \in \mathcal{B}} t_{ij} \mathbf{a}_i \quad \forall j \in \mathcal{N}$$

Theorem A.11. *The basic vector $\mathbf{a}_r$, $r \in \mathcal{B}$ can be replaced with the nonbasic vector $\mathbf{a}_s$, $s \in \mathcal{N}$, if $t_{rs} \neq 0$. The change of basis is given by the following formula*

$$\begin{aligned} t'_{ij} &= t_{ij} - \frac{t_{rj} t_{is}}{t_{rs}} & i \in \mathcal{B}', \quad i \neq s, \quad j \in \mathcal{N}', \quad j \neq r \\ t'_{sj} &= \frac{t_{rj}}{t_{rs}} & j \in \mathcal{N}', \quad j \neq r \\ t'_{ir} &= -\frac{t_{is}}{t_{rs}} & i \in \mathcal{B}', \quad i \neq s \\ t'_{rs} &= \frac{1}{t_{rs}}, \end{aligned}$$

where $\mathcal{B}' = \mathcal{B} \setminus \{r\} \cup \{s\}$ the index set of the new basis, $\mathcal{N}' = \mathcal{N} \setminus \{s\} \cup \{r\}$ and t'_{ij} are the new coefficients of the representation.

Proof. For a proof, consult e.g. [21]. $\square$

The Matlab implementation of this theorem can be found in code section 1.

A.4 Further numerical examples for the modified algorithm

Example A.12 (infeasibility).

Suppose that two observations are given: $\mathbf{a}_1 = \begin{bmatrix} 0 & 0 \end{bmatrix}^\top$, $\mathbf{a}_2 = \begin{bmatrix} 0 & 0 \end{bmatrix}^\top$. The observations are in different classes, hence $\mathbf{d} = \begin{bmatrix} 1 & -1 \end{bmatrix}^\top$. Of course there exists no hyperplane which strictly separates these two observations. Then

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad DA = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad M = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and the starting basic tableau formulation ([MLCP-SVM](#)) is

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
s_1	0	0	-1	0	0	1	0	0	0	0	-1
s_2	0	0	1	0	0	0	1	0	0	0	-1
f_1	1	-1	0	0	0	0	0	1	0	0	0
f_2	0	0	0	-1	0	0	0	0	1	0	0
f_3	0	0	0	0	-1	0	0	0	0	1	0

Exchange pivot is performed:

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	0	0	-1	0	0	0	0	1
s_2	0	0	0	0	0	1	1	0	0	0	-2
f_1	1	-1	0	0	0	0	0	1	0	0	0
f_2	0	0	0	-1	0	0	0	0	1	0	0
f_3	0	0	0	0	-1	0	0	0	0	1	0

	v_1	v_2	μ	u_1	u_2	s_1	s_2	f_1	f_2	f_3	
μ	0	0	1	0	0	-1	0	0	0	0	1
s_2	0	0	0	0	0	1	1	0	0	0	-2
v_1	1	-1	0	0	0	0	0	1	0	0	0
f_2	0	0	0	-1	0	0	0	0	1	0	0
f_3	0	0	0	0	-1	0	0	0	0	1	0

The row of s_2 tells us that this is a terminal tableau indicating the inconsistency of the mixed linear complementarity problem.

Now we solve the problem with formulation ([MLCP-SVM-2](#)). The coefficient matrix is

$$\widehat{M} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & -1 \\ -1 & 1 & 0 \end{bmatrix}$$

and the starting basic tableau is

	v_1	v_2	μ	s_1	s_2	f	
s_1	0	0	-1	1	0	0	-1
s_2	0	0	1	0	1	0	-1
f	1	-1	0	0	0	1	0

Exchange pivot yields

	v_1	v_2	μ	s_1	s_2	f	
μ	0	0	1	-1	0	0	1
s_2	0	0	0	1	1	0	-2
f	1	-1	0	0	0	1	0

	v_1	v_2	μ	s_1	s_2	f	
μ	0	0	1	-1	0	0	1
s_2	0	0	0	1	1	0	-2
v_1	1	-1	0	0	0	1	0

The row of s_2 serves as evidence that no solution exists.

A.5 Codes

Listing 1: Algorithm for pivoting

```
1 function [tableau,B,N] = pivot(tableau,i,j,B,N)
2 %Description: function for pivoting on position (i,j) of a basic tableau.
3 %B: index set of basic variables. N: index set of nonbasic variables.
4 [m,n]=size(tableau); %We save the dimensions of the pivot tableau. m is
   the number of rows, n is the number of columns.
5 %Errors:
6 if ((i<1) || (j<1) || (i>m) || (j>n-1))%Throwing error if the pivot
   position (i,j) is outside of the tableau.
7     error('Error. Pivot position is outside of tableau');
8     return
9 end
10 if (tableau(i,j)==0) %Throwing error if trying to pivot on a 0.
11     error('Error. Cannot pivot on a 0');
12     return
13 end
14 %The actual algorithm begins here.
15 tableau(:,n+1)=zeros(m,1);
16 tableau(i,n+1)=1; %Creating the would-be pivot column as a temporary
   column.
17 tableau(i,:)=tableau(i,:)/tableau(i,j); %Pivot row.
18 for k=1:m %Other rows.
19     if ((k ~= i) && (tableau(k,j) ~= 0))
20         tableau(k,:)=tableau(k,:)-tableau(k,j)*tableau(i,:);
21     end
22 end
23 tableau(:,j)=tableau(:,n+1); %Copying the pivot column to its place.
24 tableau(:,n+1)=[]; %Deleting the temporary column.
25 %Updating the index sets of basic and nonbasic variables, i.e. B and N.
26 l=B(i); B(i)=N(j); N(j)=l;
27 return
28 end
```

Listing 2: Criss-cross algorithm of Klafszky and Terlaky (1992)

```

1 function [y,w] = crisscross(T,p)
2 %Minimal index criss-cross algorithm of Klafszky and Terlaky for solving
3 %LCPs with bisymmetric T matrices. (e.g. QP problems).
4 %Input: coefficient matrix T and right hand side vector p.
5 %Output: solution vector y and slack vector w.
6
7 %Setting up the index sets:
8 n=length(p); N=1:n; %N: index set of nonbasic variables.
9 B=(n+1):2*n; %B: index set of basic variables.
10 I=find(p<0); %set of rowindices of currently negative basic variables. If
    empty, the algorithm terminates.
11
12 while(~isempty(I))
13     i=I(1); %Least index rule.
14     if (-T(i,i)<0) %Diagonal pivot if possible.
15         [tableau,B,N]=pivot([-T p],i,i,B,N);
16         p=tableau(:,n+1);
17         T=-tableau(:,1:n); %Updating the matrix T and RHS p.
18
19     else %If diagonal pivot is impossible, exchange pivot is attempted.
20         J=find(-T(i,:)<0); %Column index set of possible exchange pivot
            positions.
21         if(isempty(J)) %If there are none, a terminal tableau is reached.
22             fprintf('The LCP has no possible solution')
23             y=[]; w=[];
24             return
25         else
26             j=J(1); %Least index rule for exchange pivot position.
27             [tableau,B,N]=pivot([-T p],i,j,B,N); %First step of the
                exchange pivot.
28             p=tableau(:,n+1);
29             T=-tableau(:,1:n); %Updating the matrix T and RHS p.
30             [tableau,B,N]=pivot([-T p],j,i,B,N); %Second step of the
                exchange pivot.

```

```

31         p=tableau(:,n+1);
32         T=-tableau(:,1:n); %Updating the matrix T and RHS p.
33     end
34 end
35 I=find(p<0); %Updating the index set of negative basic variables.
36 end %The while loop ends when every basic variable is nonnegative.
37 fprintf('A solution of the LCP has been found.') %That is, a solution has
    been found.
38
39 %Extracting the output vectors.
40 y=zeros(n,1);
41 w=zeros(n,1);
42 for i=1:n %Sorting basic variables into the primal and dual solution
    vectors.
43     if(B(i) > n)
44         w(i)=p(i);
45     else
46         y(i)=p(i);
47     end %Nonbasic variables remain at 0.
48 end
49 return
50 end

```

Listing 3: Initialization for the modified criss-cross algorithm – problem ([MLCP-SVM](#))

```

1 function [M,q,L0,Lf,Ln,B,N] = initialization(A,d)
2 %Function creating the starting pivot tableau of the modified criss-cross
3 %algorithm from the data. Formulation MCLP-SVM.
4 %Input: m-by-n data matrix A (m observations, n variables) and
5 %classification vector d of dimension m.
6 %Coefficient matrix M, right hand side vector q, index sets B, N, L0, Lf,
   Ln
7 [m,n]=size(A);
8 %Throwing error if dimensions do not match
9 if (size(d)~= m)
10     error('Error. Dimensions do not match');
11     return
12 end
13 D=diag(d); DA=D*A;
14 M=[zeros(m,m), d, DA ; -transpose(d), zeros(1,n+1) ; -transpose(DA),
   zeros(n,1), eye(n)];
15 q=[-ones(m,1);zeros(n+1,1)];
16 N=1:(n+m+1); B=(n+m+1)+1:2*(n+m+1);
17 Lf=m+1:(n+m+1); L0=(n+m+1)+m+1:2*(n+m+1);
18 Ln=[1:m, (n+m+1)+1:(n+m+1)+m];
19 end

```

Listing 4: Initialization for the modified criss-cross algorithm – problem (MLCP-SVM-2)

```

1 function [M,q,L0,Lf,Ln,B,N,DA] = initialization2(A,d)
2 %Function creating the starting pivot tableau of the modified criss-cross
3 %algorithm from the data. Formulation MCLP-SVM-2.
4 %Input: m-by-n data matrix A (m observations, n variables) and
5 %classification vector d of dimension m.
6 %Coefficient matrix M, right hand side vector q, index sets B, N, L0, Lf,
7 %Ln, and matrix DA.
8 [m,n]=size(A);
9 %Throwing error if dimensions do not match
10 if (size(d)~= m)
11     error('Error. Dimensions do not match');
12     return
13 end
14 D=diag(d); DA=D*A;
15 M=[DA*transpose(DA),d;-transpose(d),0];
16 q=[-ones(m,1);0];
17 N=1:(m+1); B=(m+1)+1:2*(m+1);
18 Lf=(m+1); L0=2*(m+1);
19 Ln=[1:m,(m+1)+1:2*(m+1)-1];
20 end

```

Listing 5: Modified criss-cross algorithm for SVM

```

1 function [mu,u] = svmcrisscrossepsilon(M,q,L0,Lf,Ln,B,N,epsilon,DA)
2 %Minimal index criss-cross algorithm for solving
3 %mixed LCPs arising from SVM optimization problems.
4 %Input: epsilon is the numerical tolerance threshold.
5 %For the other input parameters, see initialization.m and initialization2
   .m
6 %Output: optimal separating hyperplane parameters (u,mu).
7
8 n=length(q);
9
10 qtech=[q,transpose(B)];
11 Mtech=[M;N];
12 In=find(qtech(:,1)<0-epsilon & ismember(qtech(:,2),intersect(B,Ln))); %
    set of rowindices of nonnegative basic variables currently taking
    negative values.
13 IO=find(abs(qtech(:,1))>0+epsilon & ismember(qtech(:,2),intersect(B,L0)))
    ; %set of rowindices of fixed basic variables currently not on their
    required value (0).
14 I=union(In,IO); %indices of all rows we currently want to pivot in.
15
16 while(~isempty(I))
17     i=I(1); %Least index rule.
18     if (ismember(i,In))
19         J=union(find(-Mtech(i,:)<0-epsilon & ismember(Mtech(end,:),
                intersect(N,Ln))),find(abs(-Mtech(i,:))>0+epsilon & ismember(
                Mtech(end,:),intersect(N,Lf)))) );
20         if(isempty(J)) %If there are none, a terminal tableau is reached.
21             fprintf('The LCP has no possible solution')
22             mu=[]; u=[];
23             return
24         else
25             if (ismember(i,J)) %Diagonal pivot if possible.
26                 [tableau,B,N]=pivot([-M q],i,i,B,N);
27                 q=tableau(:,n+1);

```

```

28         M=-tableau(:,1:n); %Updating the matrix M and RHS q.
29         diagnum=diagnum+1;
30     else
31         j=J(1); %Least index rule for exchange pivot position.
32         [tableau,B,N]=pivot([-M q],i,j,B,N); %First step of the
           exchange pivot.
33         q=tableau(:,n+1);
34         M=-tableau(:,1:n); %Updating the matrix M and RHS q.
35         [tableau,B,N]=pivot([-M q],j,i,B,N); %Second step of the
           exchange pivot.
36         q=tableau(:,n+1);
37         M=-tableau(:,1:n); %Updating the matrix M and RHS q.
38     end
39 end
40 else
41     J=union(find(abs(-Mtech(i,:)+qtech(i,1))>abs(qtech(i,1))+epsilon &
           ismember(Mtech(end,:),intersect(N,Ln))) ,find(abs(-Mtech(i,:))
           >0+epsilon & ismember(Mtech(end,:),intersect(N,Lf))));
42     if(isempty(J)) %If there are none, a terminal tableau is reached.
43         fprintf('The LCP has no possible solution')
44         mu=[]; u=[];
45         return
46     else
47         if (ismember(i,J)) %Diagonal pivot if possible.
48             [tableau,B,N]=pivot([-M q],i,i,B,N);
49             q=tableau(:,n+1);
50             M=-tableau(:,1:n); %Updating the matrix M and RHS q.
51         else
52             j=J(1); %Least index rule for exchange pivot position.
53             [tableau,B,N]=pivot([-M q],i,j,B,N); %First step of the
           exchange pivot.
54             q=tableau(:,n+1);
55             M=-tableau(:,1:n); %Updating the matrix M and RHS q.
56             [tableau,B,N]=pivot([-M q],j,i,B,N); %Second step of the
           exchange pivot.

```

```

57         q=tableau(:,n+1);
58         M=-tableau(:,1:n); %Updating the matrix M and RHS q.
59     end
60 end
61 end
62 %updating the arrays defined before the while loop started
63 qtech=[q,transpose(B)];
64 Mtech=[M;N];
65 In=find(qtech(:,1)<0-epsilon & ismember(qtech(:,2),intersect(B,Ln))); %
    set of rowindices of nonnegative basic variables currently taking
    negative values.
66 IO=find(abs(qtech(:,1))>0+epsilon & ismember(qtech(:,2),intersect(B,L0)))
    ; %set of rowindices of fixed basic variables currently not on their
    required value (0).
67 I=union(In,IO); %indices of all rows we currently want to pivot in.
68 end %ending the while loop
69 fprintf('A solution of the LCP has been found.')
70
71 %Extracting the output.
72 if (length(Lf)>1) %Formulation MLCP-SVM
73 mu=0;
74 u=zeros(length(Lf)-1,1);
75     for i=1:n
76         if (B(i)== Lf(1))
77             mu=q(i);
78         else
79             if(ismember(B(i),Lf))
80                 j=find(Lf==B(i))-1;
81                 u(j)=q(i);
82             end
83         end %Nonbasic variables remain at 0.
84     end
85 else %Formulation MLCP-SVM-2
86     if(ismissing(DA)) %Throwing error if matrix DA is missing.
87         error('Error. Formulation MLCP-SVM-2 is used but matrix DA is

```

```

88         missing.')
89     return
90     mu=0;
91     v=zeros(n-1,1);
92     j=0;
93     for i=1:n
94         if (B(i)== Lf(1))
95             mu=q(i);
96             j=1;
97         end
98         if (B(i)< Lf(1))
99             v(i-j)=q(i);
100        end
101    end
102    u=transpose(DA)*v;
103 end
104 return
105 end

```