ROBUST SUPPORT VECTOR MACHINES VIA CONIC OPTIMIZATION

VALENTINA CEPEDA[†], ANDRÉS GÓMEZ[‡], SHAONING HAN[‡]

February 2024

Abstract. We consider the problem of learning support vector machines robust to uncertainty. It has been established in the literature that typical loss functions, including the hinge loss, are sensible to data perturbations and outliers, thus performing poorly in the setting considered. In contrast, using the 0-1 loss or a suitable non-convex approximation results in robust estimators, at the expense of large computational costs. In this paper we use mixed-integer optimization techniques to derive a new loss function that better approximates the 0-1 loss compared with existing alternatives, while preserving the convexity of the learning problem. In our computational results, we show that the proposed estimator is competitive with the standard SVMs with the hinge loss in outlier-free regimes and better in the presence of outliers.
Keywords. Support vector machine, robustness, mixed-integer nonlinear optimization, convexification, indicator variables.

1. Introduction

Given labeled data $\{(\boldsymbol{x_i}, y_i)\}_{i=1}^n$, where $\boldsymbol{x_i} \in \mathbb{R}^p$ encodes the features of point i and $y_i \in \{-1, 1\}$ denotes its class, consider the support vector machine (SVM) problem defined in the seminal paper by Cortes and Vapnik [10], given by

$$\min_{\boldsymbol{w}\in\mathbb{R}^p, \boldsymbol{\xi}\in\mathbb{R}^n_+} \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^n \xi_i^\eta \tag{1a}$$

$$\text{s.t. } y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w}\right) \geq 1 - \xi_i \quad \forall i \in \{1, \ldots, n\} \tag{1b}$$

[†]Department of Industrial Engineering, Universidad de los Andes, Bogotá, Colombia. v.cepeda@uniandes.edu.co

[‡]Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA, USA. gomezand@usc.edu

[‡]Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA, USA. shaoning@usc.edu .

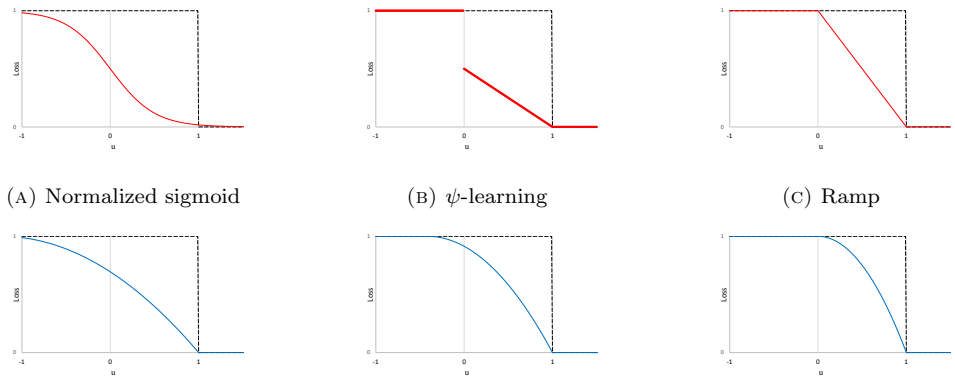
for some regularization parameter $\lambda > 0$ and some sufficiently small constant $\eta \in \mathbb{R}_+$. Cortes and Vapnik [10] argue that if λ is sufficiently large and η is sufficiently small, then an optimal solution of (1) yields a hyperplane that misclassifies the least amount of points possible and has maximum margin among all hyperplanes with minimal misclassification. However, noting that (1) is NP-hard if $\eta < 1$, the authors instead suggest using $\eta = 1$, resulting in the popular SVM with the hinge loss

$$\min_{\boldsymbol{w}\in\mathbb{R}^p,\boldsymbol{\xi}\in\mathbb{R}^n_+} \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^{n} \max\left\{0, 1 - y_i\left(\boldsymbol{x_i}^\top \boldsymbol{w}\right)\right\}. \tag{2}$$

Typical convex surrogates of the 0-1 or misclassification loss –corresponding to setting $\eta = 0$ in (1)– have been shown to perform well in low noise settings [4]. However, these surrogates are known to perform worse than the 0-1 loss in the presence of uncertainty or outliers [25, 16].

In particular, label noise, characterized by label flips in data, affects the performance of classification problems. The standard hinge loss SVM algorithm (2) tends to penalize outliers more and, as a result, these points strongly influence the placement of the separating hyperplane, leading to low classification performance. To improve SVM robustness to label uncertainty, various approaches have been proposed in the literature. Song et al. [33] suggested evaluating an adaptive margin based on the distance of each point to its class center to mitigate the influence of points far from class centroids. Other studies have focused on replacing the hinge loss function with non-convex alternatives. Some popular choices are the ramp loss [35], the rescaled hinge loss [37], the ψ-learning loss [31] or simply using the exact 0-1 loss. Figure 1 (top row) depicts these loss functions, which can be interpreted as non-convex approximations of the 0-1 loss. Unfortunately, while improving robustness, these methods may result in prohibitive increases in computational time due to the non-convex nature of the optimization problems. For example, Brooks [8] propose to use mixed-integer optimization (MIO) solvers to tackle SVMs with the ramp or 0-1 loss using big-M formulations: the resulting approach, which we review in §2, does not scale beyond $n \approx 100$. Thus, heuristics or local minimization methods which do not produce global minimizers are typically used in practice.

In this paper, we derive new non-convex loss functions for robust SVMs. The loss functions, depicted in Figure 1 (bottom), are better approximations of the 0-1 loss function than existing alternatives in the literature, and we also propose a method to tune the parameters of the loss function to ensure that the SVM problem can be solved to optimality via conic optimization. The proposed approach is inspired by convexification techniques from the MIO literature.



(A) Normalized sigmoid (B) ψ-learning (C) Ramp

(D) Conic with $\gamma_i = 0.2$ (E) Conic with $\gamma_i = 0.5$ (F) Conic with $\gamma_i = 1.0$

FIGURE 1. Non-convex robust losses approximating the 0-1 loss, as a function of $u = y_i \boldsymbol{x_i}^\top \boldsymbol{w}$. Top row: loss functions from the literature: the normalized sigmoid loss [26], the ψ-learning loss [31] and the ramp loss [35]. Bottom row: the proposed conic loss in Proposition 1 for different values of hyperparameters γ_i (with $\lambda = 1$) . By solving the conic optimization problem (17), $\boldsymbol{\gamma}$ is chosen automatically to ensure convexity of the ensuing learning problem.

MIO, machine learning and convexification. Several non-convex learning problems can be naturally cast as MIO problems [9]. In the context of SVMs, Brooks [8] proposes a MIO formulation to tackle problems with either the 0-1 or the ramp loss, using big-M reformulations to encode the non-convexities; we review this method in §2. Other MIO approaches for SVMs include problems with the hinge loss but sparsity [18, 24], as well as the approach by Ustun et al. [34] which imposes sparsity and uses the 0-1 loss.

However, simply resorting to off-the-shelf MIO solvers may result in subpar performance. Indeed, these solvers are *general-purpose*, able to solve a large variety of problems but not particularly tailored to any, and certainly not tailored to solve machine learning problems. Nonetheless, in selected learning problems, and in particular in the *best subset selection problem* [27], MIO technology can tackle problems with tens of thousands of decision variables [3, 20]. The key ingredient beyond these algorithms is the exploit of the perspective relaxation [19], that is, a strong convex relaxation of the non-convex learning problem that does not depend on Big-M terms. Interestingly, simply solving the perspective relaxation without the additional enumeration used by MIO methods (possibly with some rounding) results in an estimator in its own right [29, 36, 7, 14], closely related to the MCP estimator in the literature [38]. More sophisticated convex relaxations of MIO

formulations have been successfully deployed in several machine learning problems including regression [2, 17], matrix decomposition and completion [5, 6], sparse principal component analysis [11, 13, 21, 22] and K-means clustering [12, 28].

Contributions and outline. In this paper we use MIO technology to derive strong convex relaxations of SVMs with the 0-1 loss. The resulting formulation can be interpreted as a new non-convex loss function that: *(i)* is separable; *(ii)* underestimates the 0-1 loss; *(iii)* preserves convexity of the SVM problem; and *(iv)* is stronger than alternative loss functions proposed in literature with properties *(i)-(iii)*. Moreover, we show that SVM problems with the derived loss can be implemented as semidefinite programs (SDPs).

The rest of the paper is organized as follows. In §2 we review existing MIO formulations and their connections with the hinge loss. In §3 we derive the proposed loss function, in §4 we discuss its implementation and in §5 we present computational experiments.

Notation. In the paper, we denote vectors and matrices in **bold**. We let $\mathbf{0}$ and $\mathbf{1}$ be vectors of zeros and ones, respectively, whose dimensions can be inferred from the context. Given a number $k \in \mathbb{Z}_+$, define $[k] := \{1, \dots, k\}$. Given $\alpha \in \mathbb{R}$, we let $(\alpha)_+ := \max\{\alpha, 0\}$ and $(\alpha)_- := \max\{-\alpha, 0\}$ denote the positive and negative parts of α, respectively, so that the identity $\alpha = (\alpha)_+ - (\alpha)_-$ holds. Moreover, we use $(\alpha)_\pm^2$ as a shorthand for $((\alpha)_\pm)^2$, i.e., the positive/negative part operator is applied before squaring the term. For any set S, we denote the convex hull of S by cl conv(S). For any $a \in \mathbb{R}$, we adopt the following convention of division by 0: $a^2/0 = 0$ if $a = 0$ and $a^2/0 = +\infty$ otherwise. For any two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, we use $\boldsymbol{A} \succeq \boldsymbol{B}$ to represent that $\boldsymbol{A} - \boldsymbol{B}$ is positive semidefinite, and denote by $\langle \boldsymbol{A}, \boldsymbol{B} \rangle := \sum_i \sum_j A_{ij} B_{ij}$. Given any logical condition "$\cdot$", we use $\mathbb{1}_{\{\cdot\}}$ to denote the function that is equal to 1 whenever the condition is satisfied, and 0 otherwise.

2. The big-M formulation and hinge loss

To solve SVMs with the exact misclassification loss, Brooks [8] proposes to introduce binary variables $z_i = 1$ if point i lies in the margin or is misclassified, and $z_i = 0$ if the point is correctly classified. Then, they reformulate (1) with $\eta = 0$ as

$$\min_{\boldsymbol{w}, \boldsymbol{z}} \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^n z_i \tag{3a}$$

$$\text{s.t. } y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w} \right) \geq 1 - M z_i \quad \forall i \in [n], \tag{3b}$$

$$\boldsymbol{w} \in \mathbb{R}^p, \ \boldsymbol{z} \in \{0,1\}^n \tag{3c}$$

where M is a sufficiently large number. If $z_i = 0$, constraint (3b) forces point i to be correctly classified, while setting $z_i = 1$ incurs a fixed cost of λ but allows point i to be in the margin or in the "incorrect" side of the hyperplane. Brooks [8] also shows that the estimator obtained from solving (3) to optimality is consistent under appropriate conditions.

A good proxy for the effectiveness of branch-and-bound methods is the relaxation quality of the MIO formulation, obtained by relaxing the binary constraints to bound constraints $\mathbf{0} \leq \boldsymbol{z} \leq \mathbf{1}$.

Relaxation quality. Consider the relaxation of (3):

$$\min_{\boldsymbol{w}\in\mathbb{R}^p, \mathbf{0}\leq\boldsymbol{z}\leq\mathbf{1}} \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^n z_i \tag{4a}$$

$$\text{s.t. } y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w}\right) \geq 1 - M z_i \quad \forall i \in [n]. \tag{4b}$$

A common approach to assess the relaxation quality is through the $\texttt{gap} := (\zeta_{mio} - \zeta_{relax})/\zeta_{mio}$, where ζ_{mio} and ζ_{relax} denote the optimal objective values of (3) and (4), respectively. Unfortunately, *regardless of the data*, (4) results in almost trivial relaxation gaps. Indeed, note that the solution $\boldsymbol{w} = \mathbf{0}$, $\boldsymbol{z} = 1/M$ is always feasible with an objective value of $(\lambda n)/M \xrightarrow{M\to\infty} 0$. Thus, if a large value of M is used, the gap is close to 1. Additionally, the solutions $(\boldsymbol{w}, \boldsymbol{z})$ obtained from solving the relaxation are uninformative, which will severely impair branch-and-bound algorithms. Brooks [8] reports that in the pool of 1,425 instances tested in their paper, while 39% are solved to optimality, the average gap on the remaining 61% instances is over 70% after 10 minutes of branch-and-bound. We performed computations with this formulation using the commercial solver Gurobi, and observed a similar lackluster performance: instances with $n = 100$ and $p = 3$ cannot be solved in 10 minutes, and no instance with $n = 200$ could be solved (typically resulting in gaps above 50%).

Connections to the hinge loss. The relaxation (4) is connected with problem (2). Indeed, note that in optimal solutions of the relaxation (4) we find that $Mz_i^* = \left(1 - y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w}\right)\right)_+$, and thus (if M is big enough), problem (4) is equivalent to

$$\min_{\boldsymbol{w}\in\mathbb{R}^p, \boldsymbol{\xi}\in\mathbb{R}^n_+} \|\boldsymbol{w}\|_2^2 + \frac{\lambda}{M} \sum_{i=1}^n \max\left\{0, 1 - y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w}\right)\right\}.$$

Thus, we see that SVMs with the hinge loss (2) can be interpreted as an approximation[1] of (3) informed by its natural continuous relaxation.

[1]Problem (2) is only a relaxation if the regularization controlling the hinge loss is negligible. Since practical uses of (3) use much larger parameters, i.e., values of M too small

Alternative non-convex reformulations. Observe that we can rewrite the misclassification constraints for point $i \in [n]$ as the quadratic non-convex constraints

$$(1 - y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w} \right))(1 - z_i) \leq 0 \tag{5a}$$

$$(1 - y_i \left(\boldsymbol{x_i}^\top \boldsymbol{w} \right))z_i \geq 0. \tag{5b}$$

Constraints (5a) correspond exactly to (3b); constraints (5b), which force misclassified to be in the "incorrect" side of the hyperplane, are valid (but redundant) for the core SVM problem, and we keep them in the formulation for generality. While reformulation (5) does not offer an advantage in terms of MIO methods, we use these constraints in the rest of the paper as they exactly represent the non-convexities of SVM problems without introducing artificial bounds M.

3. Loss functions via convexification

Consider the problem of designing loss functions $\mathcal{L}_i : \mathbb{R} \to \mathbb{R}_+$ (which may depend on the point $i \in [n]$) such that

$$\nu^* = \min_{\boldsymbol{w} \in \mathbb{R}^p} \ \|\boldsymbol{w}\|_2^2 + \sum_{i=1}^{n} \mathcal{L}_i(y_i \boldsymbol{x_i}^\top \boldsymbol{w}) \tag{6}$$

is a *strong* (i.e., the objective ν^* should be as large as possible) and *convex* relaxation of the SVM problem with 0-1 loss (i.e., $\mathcal{L}_i(y_i \boldsymbol{x_i}^\top \boldsymbol{w}) \leq \lambda \mathbb{1}_{\{y_i \boldsymbol{x_i}^\top \boldsymbol{w} < 1\}}$). Note that these two goals are often at odds with each other. A natural approach to ensure convexity is forcing all loss functions to be convex themselves: however, in that case (and assuming the existence of an upper bound on the maximum margin $y_i \boldsymbol{x_i}^\top \boldsymbol{w} - 1$) the best possible relaxation is the hinge/big-M relaxation which, as discussed in §2, is weak – and reduces to the trivial lower bound $\mathcal{L}_i(\cdot) = 0$ as the upper bound goes to infinity. Alternatively, the non-convex losses depicted in Figure 1 (except the sigmoid loss, which is not an underestimator) are stronger, but in general result in non-convex problems. The key idea is thus to carefully design loss functions that are non-convex, but the induced non-convexities are offset by the strong convexity of the term $\|\boldsymbol{w}\|_2^2$.

Consider the extreme case where function $\mathcal{L}_i$ subsumes all non-convexities of the problem and thus $\mathcal{L}_j = 0$ for all $j \neq i$. In this case, the best loss function $\mathcal{L}_i$ can be obtained from the convex envelope of the function $\psi(\boldsymbol{w}) := \|\boldsymbol{w}\|_2^2 + \lambda \mathbb{1}_{\{y_i \boldsymbol{x_i}^\top \boldsymbol{w} < 1\}}$. In this section, we derive this convex

to guarantee the correctness of the MIO formulation, we prefer the term "approximation" instead.

envelope by studying the more general mixed-integer set

$$W_Q = \Big\{ \boldsymbol{w} \in \mathbb{R}^p,\ z \in \{0,1\},\ t \in \mathbb{R} : t \geq \boldsymbol{w}^\top \boldsymbol{Q} \boldsymbol{w}, \\ \Big(1 - y(\boldsymbol{x}^\top \boldsymbol{w})\Big)(1-z) \leq 0,\ \Big(1 - y(\boldsymbol{x}^\top \boldsymbol{w})\Big) z \geq 0 \Big\},$$

where $\boldsymbol{Q} \succeq 0$ and where we dropped the index i from W_Q for simplicity. Indeed, observe that the epigraph of ψ can be written in terms of W_Q by setting $\boldsymbol{Q} = \boldsymbol{I}$, since

$$\{(\boldsymbol{w}, \tau) : \psi(\boldsymbol{w}) \leq \tau\} \Leftrightarrow \{(\boldsymbol{w}, \tau) : \exists (z,t) \text{ s.t. } t + \lambda z \leq \tau,\ (\boldsymbol{w}, z, t) \in W_I\}.$$

The rest of the section is devoted to studying cl conv(W_Q).

3.1. **Derivation of the convex hull.** This subsection is devoted to proving Theorem 1 below.

Theorem 1. *The convex hull of W_Q is described by bound constraints $0 \leq z \leq 1$, and inequality*

$$t \geq \boldsymbol{w}^\top \boldsymbol{Q} \boldsymbol{w} - \gamma \Big(1 - y(\boldsymbol{x}^\top \boldsymbol{w})\Big)^2 + \gamma \frac{\Big(1 - y(\boldsymbol{x}^\top \boldsymbol{w})\Big)_+^2}{z} + \gamma \frac{\Big(1 - y(\boldsymbol{x}^\top \boldsymbol{w})\Big)_-^2}{1-z},$$

where $\gamma \in \mathbb{R}_+$ is the largest number such that $\boldsymbol{w}^\top \boldsymbol{Q} \boldsymbol{w} - \gamma \Big(1 - y(\boldsymbol{x}^\top \boldsymbol{w})\Big)^2$ is convex, given by $\gamma = 1/(\boldsymbol{x}^\top \boldsymbol{Q}^{-1} \boldsymbol{x})$.

The rest of this subsection can be safely skipped by readers not interested in the proof of the main theorem. We first establish the convexification result for a simplified version of W_Q with only three variables. Given $b \in \mathbb{R}$, define

$$\widehat{W} = \big\{(w,z,t) :\ t \geq w^2,\ z \in \{0,1\},\ (w-b)(1-z) \geq 0,\ (w-b)z \leq 0\big\}.$$

Lemma 1. *The convex hull of $\widehat{W}$ described by $0 \leq z \leq 1$ and*

$$t \geq \frac{(w-b)_+^2}{1-z} + \frac{(w-b)_-^2}{z} + 2bw - b^2. \tag{7}$$

Proof. By definition, $(w,z,t) \in \text{cl conv}(\widehat{W})$ if and only if there exists $0 \leq \lambda \leq 1$ and $(t_0, w_0, 0), (t_1, w_1, 1) \in \widehat{W}$, such that $(t, w, z) = (1-\lambda)(t_0, w_0, 0) + \lambda(t_1, w_1, 1)$. Equivalently, we find that $z = \lambda$ and the following inequality system is consistent

$$\begin{aligned} & t = (1-z)t_0 + z t_1,\ w = (1-z)w_0 + z w_1, \\ & t_0 \geq w_0^2,\ w_0 - b \geq 0, \\ & t_1 \geq w_1^2,\ w_1 - b \leq 0. \end{aligned}$$

By substituting out t_i with their lower bounds w_i^2, $i = 1, 2$, we find the equivalent system

$$\begin{aligned} &t \geq (1-z)w_0^2 + zw_1^2,\ w = (1-z)w_0 + zw_1, \\ &w_0 - b \geq 0,\ w_1 - b \leq 0. \end{aligned} \tag{8}$$

It suffices to prove by cases that after eliminating additional variables (w_0, w_1), (8) is equivalent to (7).

- *Case 1:* $z = 0$. In this case, by the convention of division by 0, (7) reduces to $t \geq (w-b)_+^2 + 2bw - b^2$ and $(w-b)_- = 0$, which is equivalent to $w \geq b$ and $t \geq (w-b)_+^2 + 2bw - b^2 = w^2$, or namely, (8).
- *Case 2:* $z = 1$. In this case, by the convention of division by 0, (7) reduces to $t \geq (w-b)_-^2 + 2bw - b^2$ and $(w-b)_+ = 0$, which is equivalent to $w \leq b$ and $t \geq (w-b)_-^2 + 2bw - b^2 = w^2$, or namely, (8).
- *Case 3:* $0 < z < 1$. Since $w = (1-z)w_0 + zw_1 = (1-z)(w_0 - b) + z(w_1 - b) + b$, one has

$$\begin{aligned} &(1-z)w_0^2 + zw_1^2 \\ =&(1-z)(w_0-b)^2 + z(w_1-b)^2 + b^2 + 2b[(1-z)(w_0-b) + z(w_1-b)] \\ =&(1-z)(w_0-b)^2 + z(w_1-b)^2 + b^2 + 2b(w-b) \\ =&(1-z)(w_0-b)^2 + z(w_1-b)^2 + 2bw - b^2. \end{aligned}$$

  By changing variables $p = (w_0 - b)(1-z)$ and $q = -(w_1 - b)z$, eliminating (w_0, w_1) in (8) amounts to

$$\begin{aligned} t \geq \min_{p,q}\ &\frac{p^2}{1-z} + \frac{q^2}{z} + 2bw - b^2 \\ \text{s.t. }\ &w - b = p - q,\ p \geq 0,\ q \geq 0. \end{aligned}$$

  It can be seen easily that at the optimal solution, $p = (w-b)_+$ and $q = (w-b)_-$. The conclusion follows by substituting out the optimal p, q.

This finishes the proof. □

We can use, using an appropriate linear transformation of variables, recover the proof of Theorem 1 from Lemma 1.

Proof of Theorem 1. Without loss of generality, we assume $y = 1$; otherwise one can change variables $x \leftarrow yx$. Denote $\boldsymbol{U}$ as an orthonormal matrix whose first column is $\boldsymbol{Q}^{-1/2}\boldsymbol{x} / \left\|\boldsymbol{Q}^{-1/2}\boldsymbol{x}\right\|_2$. Note that $\boldsymbol{U}^{-1} = \boldsymbol{U}^\top$. Let $\boldsymbol{u} = \boldsymbol{U}^\top \boldsymbol{Q}^{1/2}\boldsymbol{w}$. By the construction, one has $\boldsymbol{w}^\top \boldsymbol{Q}\boldsymbol{w} = \|\boldsymbol{U}\boldsymbol{u}\|_2^2 = \|\boldsymbol{u}\|_2^2$, Moreover, $\boldsymbol{x}^\top \boldsymbol{w} = \boldsymbol{x}^\top \boldsymbol{Q}^{-1/2}\boldsymbol{U}\boldsymbol{u} = \left(\boldsymbol{U}^\top \boldsymbol{Q}^{-1/2}\boldsymbol{x}\right)^\top \boldsymbol{u} = \left\|\boldsymbol{Q}^{-1/2}\boldsymbol{x}\right\|_2 u_1$. Let $b = 1/\left\|\boldsymbol{Q}^{-1/2}\boldsymbol{x}\right\|_2$. The change of variables leads to the following bijective

affine transformation of W_Q, which is denoted by $\widetilde{W}$

$$\widetilde{W} = \Big\{(\boldsymbol{u}, z, t) : t \geq \|\boldsymbol{u}\|_2^2,\ z \in \{0,1\}, (b-u_1)(1-z) \leq 0,\ (b-u_1)z \geq 0\Big\}.$$

Since $\|\boldsymbol{u}\|_2^2 = \sum_{i=1}^p u_i^2$ is separable in u_i, and u_i is only involved in the constraint $t \geq \|\boldsymbol{u}\|_2^2$ for all $i \geq 2$, one can deduce that

$$\begin{aligned}\text{cl conv}(\widetilde{W}) &= \left\{(\boldsymbol{u}, z, t) : \left(u_1, z, t - \sum_{i=2}^2 u_i^2\right) \in \text{cl conv}(\widehat{W})\right\} \\ &= \left\{(\boldsymbol{u}, z, t) :\ 0 \leq z \leq 1,\ t \geq \sum_{i=2}^p u_i^2 + \frac{(b-u_1)_-^2}{1-z} + \frac{(b-u_1)_+^2}{z} + 2bu_1 - b^2\right\}.\end{aligned}$$

The formula for the convex hull of W can be recovered by substituting out $\|\boldsymbol{u}\|_2^2 = \boldsymbol{w}^\top \boldsymbol{Q}\boldsymbol{w}$, $b - u_1 = 1/\left\|\boldsymbol{Q}^{-1/2}\boldsymbol{x}\right\|_2 - \boldsymbol{x}^\top\boldsymbol{w}/\left\|\boldsymbol{Q}^{-1/2}\boldsymbol{x}\right\|_2 = \sqrt{\gamma}\left(1 - \boldsymbol{x}^\top\boldsymbol{w}\right)$:

$$\begin{aligned}t \geq& \sum_{i=2}^p u_i^2 + \frac{(b-u_1)_-^2}{1-z} + \frac{(b-u_1)_+^2}{z} + 2bu_1 - b^2 \\ =& \|\boldsymbol{u}\|_2^2 + \frac{(b-u_1)_-^2}{1-z} + \frac{(b-u_1)_+^2}{z} - (b-u_1)^2 \\ =& \boldsymbol{w}^\top\boldsymbol{Q}\boldsymbol{w} - \gamma\Big(1 - (\boldsymbol{x}^\top\boldsymbol{w})\Big)^2 + \gamma\frac{\Big(1 - \boldsymbol{x}^\top\boldsymbol{w}\Big)_+^2}{z} + \gamma\frac{\Big(1 - \boldsymbol{x}^\top\boldsymbol{w}\Big)_-^2}{1-z},\end{aligned}$$

thus concluding the proof. □

3.2. **Interpretation as regularization.** From Theorem 1, we found a representation (with an additional variable z_i) of the best loss function based on a single margin $y_i\boldsymbol{x_i}^\top\boldsymbol{w}$. To retrieve a representation in the original space of variables, it suffices to project out the additional variable:

$$\mathcal{L}^*(y_i\boldsymbol{x_i}^\top\boldsymbol{w}; \gamma_i) := \min_{0\leq z_i\leq 1}\ \lambda z_i - \gamma_i\Big(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\Big)^2 \tag{9}$$

$$+ \gamma_i\frac{\Big(1 - y_i(\boldsymbol{x_i}^\top\boldsymbol{w})\Big)_+^2}{z_i} + \gamma_i\frac{\Big(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\Big)_-^2}{1-z_i} \tag{10}$$

for some properly chosen $\gamma_i > 0$. Observe that we remove the subscript from $\mathcal{L}$ since the dependence on the datapoint is captured by γ_i. The resulting function is given in Proposition 1 below.

Proposition 1. *Assuming $\gamma_i, \lambda > 0$, the optimal value of problem* (10) *is the non-convex loss function*

$$\mathcal{L}^*(u; \gamma_i) = \begin{cases} 0 & \text{if } \ 1 - u \leq 0 \\ 2\sqrt{\lambda\gamma_i}(1-u) - \gamma_i(1-u)^2 & \text{if } \ 0 < 1 - u \leq \sqrt{\lambda/\gamma_i} \\ \lambda & \text{if } \ 1 - u > \sqrt{\lambda/\gamma_i} \end{cases}$$

where $u = y_i \boldsymbol{x_i}^\top \boldsymbol{w}$.

Proof. To determine the optimal value of z_i in (10), we consider two cases depending on the value of $y_i(\boldsymbol{x_i}^\top \boldsymbol{w})$.

• *Case 1:* $1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w}) \leq 0$. In this case, the point is classified on the correct side of the margin. We find that the optimal value is $z_i^* = 0$ as the objective is minimized to zero.

• *Case 2:* $1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w}) \geq 0$. In this case, we find by setting the derivative with respect to z_i to zero (10) that the optimal solution is

$$z_i^* = \min\left\{\sqrt{\frac{\gamma_i}{\lambda}}\Big(1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w})\Big), 1\right\},$$

with objective value $2\sqrt{\lambda\gamma_i}(1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w})) - (1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w}))^2$ if $0 < 1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w}) \leq \sqrt{\lambda/\gamma_i}$, and λ if $1 - y_i(\boldsymbol{x_i}^\top \boldsymbol{w}) > \sqrt{\lambda/\gamma_i}$. □

Figure 1 depicts the loss function $\mathcal{L}^*$ along with alternatives from the literature. The parameter γ_i controls the tradeoff between non-convexity and approximation to the 0-1 loss. Note that for $\gamma_i = 1$, the derived loss is a better approximation of the misclassification loss than the normalized sigmoid, ψ-learning, and ramp losses. In problems with uncertainty and outliers, the concave downward shape of the loss function redistributes the influence from outliers to points near the boundary. This reduces the sensitivity to noise when compared to hinge and other non-convex losses.

Note that using loss function $\mathcal{L}^*$ in practice is not trivial. First, it requires a careful selection of parameters $\boldsymbol{\gamma}$ to guarantee an ideal tradeoff between convexity and strength. While Theorem 1 gives the best value in the context of a single observation $(\boldsymbol{x_i}, y_i)$, it is not immediately clear how to generalize the result. Indeed, if all γ_i are simultaneously set according to Theorem 1, i.e. $\gamma_i = 1/\|\boldsymbol{x_i}\|_2^2$, then the resulting problem is non-convex, posing difficulties for optimization. Second, even if $\boldsymbol{\gamma}$ is given, optimization with the loss function as described in Proposition 1 can be difficult, as it is non-smooth and defined by pieces (thus not directly compatible with off-the-shelf solvers). In the next section we discuss how to resolve these issues.

4. Implementation via conic optimization

Consider now problem (6) where each loss function corresponds to (10) for suitable values of γ_i, that is,

$$\nu^* = \min_{\substack{\boldsymbol{w}\in\mathbb{R}^p \\ \boldsymbol{z}\in[0,1]^n}} \|\boldsymbol{w}\|_2^2 + \lambda\sum_{i=1}^n z_i - \sum_{i=1}^n \gamma_i \left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\right)^2 \tag{11}$$

$$+ \sum_{i=1}^n \gamma_i \left(\frac{\left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\right)_+^2}{z_i} + \frac{\left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w} - 1\right)_-^2}{1-z_i}\right). \tag{12}$$

Note that if constraints $\boldsymbol{z}\in\{0,1\}^n$ were additionally imposed, (12) is a valid formulation for problem (3) , provided that $\boldsymbol{\gamma} > \boldsymbol{0}$; moreover, the relaxation is stronger than the big-M/hinge relaxation.

Since convex quadratic terms divided by nonnegative linear terms are convex and second-order cone representable [1, 23], it follows that the terms in the last summation of (12) are convex. Thus, to ensure convexity of the objective in (12), it suffices to ensure the convexity of the function $\|\boldsymbol{w}\|_2^2 - \sum_{i=1}^n \gamma_i \left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\right)^2$, which is equivalent to

$$\boldsymbol{I} - \sum_{i=1}^n \gamma_i \boldsymbol{x_i}\boldsymbol{x_i}^\top \succeq 0. \tag{13}$$

Therefore, the relaxation of SVM we propose can be formulated as the max/min optimization problem

$$\max_{\boldsymbol{\gamma}\in\mathbb{R}^n_+:(13)} \min_{\substack{\boldsymbol{w}\in\mathbb{R}^p \\ \boldsymbol{z}\in[0,1]^n}} \|\boldsymbol{w}\|_2^2 + \lambda\sum_{i=1}^n z_i - \sum_{i=1}^n \gamma_i \left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\right)^2 \tag{14}$$

$$+ \sum_{i=1}^n \gamma_i \left(\frac{\left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\right)_+^2}{z_i} + \frac{\left(1 - y_i\boldsymbol{x_i}^\top\boldsymbol{w}\right)_-^2}{1-z_i}\right), \tag{15}$$

where the inner minimization solves the convex relaxation of the MIO problem, and the outer maximization seeks to find vector $\boldsymbol{\gamma}$ resulting in the strongest convex relaxation.

We start by denoting $h_i(\boldsymbol{w}, z_i) \geq 0$ as the strengthening associated with point i

$$h_i(\boldsymbol{w}, z_i) = \left(\frac{(1 - y_i(\boldsymbol{x_i}^\top\boldsymbol{w}))_+^2}{z_i} + \frac{(1 - y_i(\boldsymbol{x_i}^\top\boldsymbol{w}))_-^2}{1-z_i}\right) - (1 - y_i(\boldsymbol{x_i}^\top\boldsymbol{w}))^2. \tag{16}$$

Observe that function h_i corresponds to $\mathcal{L}_i^*$ with $\gamma_i = 1$, but we make explicit the dependence on the additional variable z_i. In addition of variables $\boldsymbol{0} \leq \boldsymbol{z} \leq \boldsymbol{1}$ and $\boldsymbol{w}\in\mathbb{R}^p$, we will introduce a matrix variable $\boldsymbol{W}\in\mathbb{R}^{p\times p}$ which can be interpreted as representing the outer product $\boldsymbol{W} \approx \boldsymbol{w}\boldsymbol{w}^\top$.

Theorem 2. *Problem* (15) *is equivalent to the SDP*

$$\min_{\boldsymbol{w},\,\boldsymbol{z},\,\boldsymbol{W}} \ \mathrm{Tr}(\boldsymbol{W}) + \lambda \sum_{i=1}^{n} z_i \tag{17a}$$

$$s.t.\ \langle \boldsymbol{x_i x_i^\top}, \boldsymbol{W}\rangle - 2y_i(\boldsymbol{x_i^\top w}) + 1 \geq \frac{(1-y_i(\boldsymbol{x_i^\top w}))_+^2}{z_i} \tag{17b}$$

$$+ \frac{(1-y_i(\boldsymbol{x_i^\top w}))_-^2}{1-z_i} \quad \forall i \in [n] \tag{17c}$$

$$\begin{pmatrix} 1 & \boldsymbol{w}^\top \\ \boldsymbol{w} & \boldsymbol{W} \end{pmatrix} \succeq 0,\ \boldsymbol{w} \in \mathbb{R}^p,\ \boldsymbol{W} \in \mathbb{R}^{p\times p} \tag{17d}$$

$$\boldsymbol{z} \in [0,1]^n. \tag{17e}$$

Proof. We first interchange max and min in (15). Note that the objective is convex in $\boldsymbol{w}$ and $\boldsymbol{z}$, and linear in $\boldsymbol{\gamma}$. Moreover, the feasible region of $\boldsymbol{\gamma}$ is compact and convex as it is bounded by the positive semidefinite (13) and linear constraints. Thus, using Sion's Minimax Theorem [32], we switch the order of the operators, and (15) reduces to

$$\min_{\boldsymbol{w},\,\boldsymbol{0}\leq\boldsymbol{z}\leq\boldsymbol{1}} \ \max_{\boldsymbol{\gamma},\,\boldsymbol{A}} \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^{n} z_i + \sum_{i=1}^{n} \gamma_i h_i(\boldsymbol{w}, z_i)$$

$$\text{s.t. } \boldsymbol{I} - \sum_{i=1}^{n} \gamma_i \boldsymbol{x_i x_i^\top} = \boldsymbol{A} \tag{$\boldsymbol{V}$}$$

$$\boldsymbol{\gamma} \geq \boldsymbol{0} \tag{$\boldsymbol{s}$}$$

$$\boldsymbol{A} \succeq \boldsymbol{0}. \tag{$\boldsymbol{S}$}$$

Next, we dualize the inner maximization problem. The dual variables associated with each constraint are introduced in the above formulation. We obtain the conic dual

$$\min_{\substack{\boldsymbol{w}\in\mathbb{R}^p,\,\boldsymbol{0}\leq\boldsymbol{z}\leq\boldsymbol{1}\\ \boldsymbol{s}\geq 0,\,\boldsymbol{S}\succeq 0}} \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^{n} z_i + \max_{\boldsymbol{\gamma},\,\boldsymbol{A}} \left\{ \sum_{i=1}^{n} \gamma_i h_i(\boldsymbol{w}, z_i) + \boldsymbol{s}^\top \boldsymbol{\gamma} + \langle \boldsymbol{S}, \boldsymbol{A}\rangle \right.$$

$$\left. + \left\langle \boldsymbol{V}, \boldsymbol{A} + \sum_{i=1}^{n} \gamma_i \boldsymbol{x_i x_i^\top} - \boldsymbol{I} \right\rangle \right\}.$$

Observe that the inner maximization problem is unbounded unless $-s_i = h_i(\boldsymbol{w}, z_i) + \langle \boldsymbol{V}, \boldsymbol{x_i x_i^\top}\rangle$ for all points, and $-\boldsymbol{V} = \boldsymbol{A}$. Thus, the problem

reduces to

$$\begin{aligned}\min_{\boldsymbol{w},\boldsymbol{z},\boldsymbol{V}}\ & \|\boldsymbol{w}\|_2^2+\lambda\sum_{i=1}^n z_i-\langle \boldsymbol{V},\boldsymbol{I}\rangle\\ \text{s.t. }& h_i(\boldsymbol{w},z_i)+\boldsymbol{x_i}^\top \boldsymbol{V}\boldsymbol{x_i}\le 0\quad \forall i\in[n]\\ & -\boldsymbol{V}\succeq 0,\ \boldsymbol{V}\in\mathbb{R}^{p\times p}\\ & \boldsymbol{w}\in\mathbb{R}^p,\ \boldsymbol{0}\le\boldsymbol{z}\le\boldsymbol{1}.\end{aligned}$$

Finally, we perform the change of variables $\boldsymbol{W}=\boldsymbol{w}\boldsymbol{w}^\top-\boldsymbol{V}$, we obtain the formulation

$$\begin{aligned}\min_{\boldsymbol{w},\boldsymbol{z},\boldsymbol{W}}\ & \sum_{j=1}^n W_{jj}+\lambda\sum_{i=1}^n z_i\\ \text{s.t. }& h_i(\boldsymbol{w},z_i)+(\boldsymbol{x_i}^\top\boldsymbol{w})^2-\langle \boldsymbol{x_i}\boldsymbol{x_i}^\top,\boldsymbol{W}\rangle\le 0\quad \forall i\in[n]\\ & \boldsymbol{W}\succeq\boldsymbol{w}\boldsymbol{w}^\top,\ \boldsymbol{w}\in\mathbb{R}^p,\ \boldsymbol{W}\in\mathbb{R}^{p\times p}\\ & \boldsymbol{0}\le\boldsymbol{z}\le\boldsymbol{1}.\end{aligned}$$

We recover (17) by replacing h_i with its definition and using the Schur complement for constraints $\boldsymbol{W}-\boldsymbol{w}\boldsymbol{w}^\top\succeq 0$. □

Note that problem (17) is an SDP with $n+p+p^2$ variables, n constraints and positive definite constraints involving cones of order $p+1$. We point out that while SDPs are infamous for being difficult to solve, the main factor affecting runtimes with leading conic solvers –in our case, Mosek– is the order of the positive semidefinite cones. In particular, in our computations presented in the next section, we observe that runtimes scale almost linearly with the number of datapoints n, but polynomially with the number of features p. Thus, the proposed approach can be directly used is datasets with n in the thousands, provided that the number of features is relatively small – these types of datasets are common in high-stakes domains such as policy-making and healthcare, e.g., see [30] and the references therein.

Remark 1. In practice, Kernel transformations are often used to produce classifiers that are nonlinear. The proposed approach extends to Kernel formulations, and we discuss this extension in Appendix A.

5. Computations

We now discuss computations with the proposed methods.

5.1. **Synthetic instances.** We describe in §5.1.1 the instance generation process, in §5.1.2 the methods used and present in §5.1.3 results.

5.1.1. *Instance generation.* The instances considered are inspired by the instances in [8]: the data from each class is generated from a different Gaussian distribution, and the presence of different types of outliers may obfuscate the data. We now describe in detail the data generation process.

Specifically, let $n, p \in \mathbb{Z}_+$ be dimension parameters, and $\sigma \in \mathbb{R}_+$ represent the standard deviation of the Gaussian distributions. We consider three outlier classes (none, clustered, spread). We first generate a direction $\boldsymbol{d} \in \mathbb{R}^p$ where each entry is generated independently from a uniform distribution in $[-1, 1]$. We then set two centroids $\boldsymbol{\chi_1} = 0.5\boldsymbol{d}/\|\boldsymbol{d}\|_2$ and $\boldsymbol{\chi_{-1}} = -0.5\boldsymbol{d}/\|\boldsymbol{d}\|_2$, and note that they are always one unit apart. Then we generate n points $\boldsymbol{x_i} \in \mathbb{R}^{p+1}$ where $(x_i)_1 = 1$ and the remaining coordinates depend on the outlier class, and are generated as follows:

• **none** With 0.5 probability the remainder coordinates are generated from $\mathcal{N}(\chi_1, \sigma^2\boldsymbol{I})$ and set the label $y_i = 1$, and with 0.5 probability the remainder coordinates are generated from $\mathcal{N}(\chi_{-1}, \sigma^2\boldsymbol{I})$ and set the label $y_i = -1$.

• **clustered** With 0.45 probability the remainder coordinates are generated from $\mathcal{N}(\chi_1, \sigma^2\boldsymbol{I})$ and set the label $y_i = 1$, with 0.45 probability the remainder coordinates are generated from $\mathcal{N}(\chi_{-1}, \sigma^2\boldsymbol{I})$ and set the label $y_i = -1$, and with 0.1 probability the remainder coordinates are generated from $\mathcal{N}(10\chi_{-1}, 0.001\sigma^2\boldsymbol{I})$ and set the label $y_i = 1$.

• **spread** An expected 45% of the points are generated from $\mathcal{N}(\chi_1, \sigma^2\boldsymbol{I})$ and set the label $y_i = 1$, 45% are generated from $\mathcal{N}(\chi_{-1}, \sigma^2\boldsymbol{I})$ and set the label $y_i = -1$, 5% of the points are generated from $\mathcal{N}(\chi_1, 100\sigma^2\boldsymbol{I})$ and set the label $y_i = 1$ and 5% of the points are generated from $\mathcal{N}(\chi_{-1}, 100\sigma^2\boldsymbol{I})$ and set the label $y_i = -1$. In all cases, the *ideal* Bayes classifier is the line perpendicular to $\boldsymbol{\chi_1} - \boldsymbol{\chi_{-1}}$ going through the origin, $\hat{\boldsymbol{w}}^\top = (0\ \boldsymbol{d}^\top)$. We point out that we performed additional experiments where the data is perfectly separable but some labels are flipped: the conclusions are similar to the ones presented here (and also our results with real data), hence we defer the presentation of these computations to Appendix B.

5.1.2. *Methods, metrics and implementation.* We compare the following three methods: SVMs with the hinge loss, SVMs with the conic loss, and the Bayes classifier described in the previous subsubsection (corresponding to the best possible performance). In each experiment, we generate a training and validation set (both contaminated by outliers for classes "clustered" and "spread"), each with n observations, and a testing set without outliers and $100,000$ observations. For the SVM approaches, we solve the training problem on the training set for 100 different values of the hyperparameter, choose the solution that results in fewer misclassified points in validation, and report the (out-of-sample) misclassification rate on the testing set, as

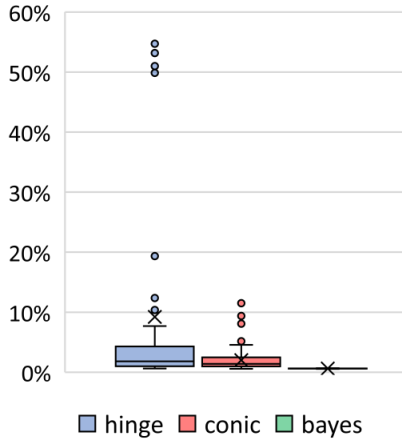
well as the total time required to perform cross-validation (i.e., solving 100 training problems). For each combination of instance class and dimensions, we repeat this process 20 times and report averages and standard deviations across all replications. For SVMs with the hinge loss, we set $\lambda = \beta/(1-\beta)$ where the 100 values of β are selected uniformly in the interval $(0, 1)$; for SVMs with the conic loss, we use the method described in Appendix C.

To train SVMs with the conic loss, we use Mosek 10.0 solver (with default parameters) on a laptop with a 12th Gen Intel Core i7-1280P (20 CPUs) processor and 32 GM RAM. For the hinge loss, we solve the training problems as convex quadratic programs using Gurobi 10.0.1 solver (default parameters) on the same laptop.

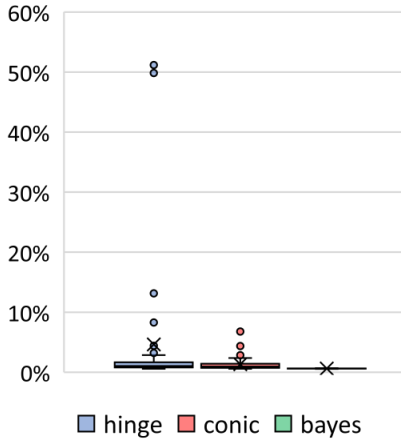
5.1.3. *Results.* Table 1 shows the results in synthetic instances. We observe that in regimes with no outliers, SVMs with the hinge and conic losses result in almost identical performance of the ensuing estimators. However, in regimes with outliers, using the conic loss typically results in superior performance, having better average out-of-sample misclassification *and* smaller variance. This phenomenon is particularly visible in instances with "clustered" outliers and $\sigma = 0.2$ (shown in bold in Table 1), and Figure 2 depicts detailed results in this setting. We observe that the large variance of the hinge estimator is due to a subset of instances where the estimator breaks down, resulting in solutions with over 50% misclassification rate. In contrast, the conic loss does not exhibit such a pathological behavior and performs well consistently.

TABLE 1. Out-of-sample misclassification rate with synthetic instances, as a function of outlier generation. Results in **bold** (clustered, $\sigma = 0.2$) are presented in more detail in Figure 2.

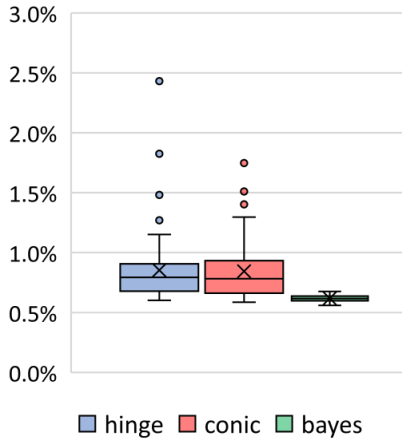
σ	method	none	clustered	spread
0.2	hinge	$0.8\% \pm 0.2\%$	$\mathbf{3.9\% \pm 10.9\%}$	$0.9\% \pm 0.5\%$
	conic	$1.0\% \pm 0.5\%$	$\mathbf{1.2\% \pm 1.2\%}$	$1.0\% \pm 0.6\%$
	bayes	$0.6\% \pm 0.0\%$	$\mathbf{0.6\% \pm 0.0\%}$	$0.6\% \pm 0.0\%$
0.5	hinge	$16.8\% \pm 1.3\%$	$21.7\% \pm 8.7\%$	$18.2\% \pm 2.3\%$
	conic	$17.0\% \pm 1.3\%$	$18.3\% \pm 3.5\%$	$17.5\% \pm 1.8\%$
	bayes	$15.9\% \pm 0.1\%$	$15.9\% \pm 0.1\%$	$15.9\% \pm 0.1\%$
1.0	hinge	$32.5\% \pm 1.9\%$	$36.5\% \pm 5.6\%$	$34.4\% \pm 3.0\%$
	conic	$32.6\% \pm 1.9\%$	$34.1\% \pm 3.4\%$	$32.9\% \pm 2.1\%$
	bayes	$30.9\% \pm 0.1\%$	$30.9\% \pm 0.1\%$	$30.9\% \pm 0.1\%$



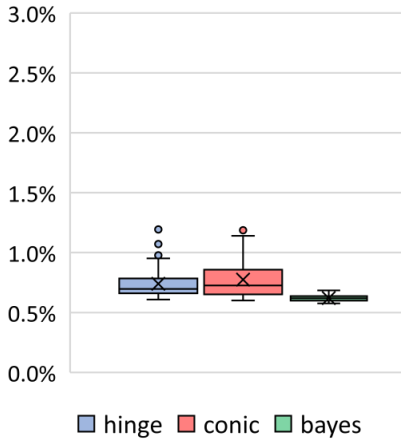
(A) $n = 100$,
hinge $= 17.0\% \pm 9.2\%$, conic $= 1.9\% \pm 2.0\%$



(B) $n = 200$,
hinge $= 12.0\% \pm 4.6\%$, conic $= 1.1\% \pm 1.3\%$



(C) $n = 500$,
hinge $= 0.9\% \pm 0.3\%$, conic $= 0.8\% \pm 0.2\%$



(D) $n = 1,000$,
hinge $= 0.7\% \pm 0.1\%$, conic $= 0.8\% \pm 0.2\%$

FIGURE 2. Distribution of out-of-sample misclassification for data with clustered outliers and $\sigma = 0.2$, as a function of the number of datapoints n. In instances with small n (top row), the hinge estimator has a probability of breaking down, resulting in out-of-sample misclassifications above 50%; *the conic loss reduces the average misclassification rate by an order-of-magnitude.* Moreover, the conic estimator performs consistently good in all settings.

Finally, Table 2 presents the computational times required to select a model using cross-validation as described before. We found that the computational times depend on n and p but not on the instance class, thus we present aggregated results across all instance classes. We observe that while

the conic loss is certainly more expensive to solve than the hinge loss, instances with up to $n = 1,000$ are solved in less than a second (and thus the complete cross-validation process requires less than 100 seconds).

TABLE 2. Average solution times and standard deviations (in seconds) across all instance classes required to solve the 100 training problems for cross-validation.

n	method	$p = 3$	$p = 5$	$p = 10$	$p = 30$
100	hinge	0.2 ± 0.0	0.2 ± 0.0	0.2 ± 0.0	0.3 ± 0.0
	conic	2.8 ± 0.3	2.9 ± 0.4	5.2 ± 1.4	41.2 ± 4.7
200	hinge	0.5 ± 0.1	0.5 ± 0.1	1.9 ± 0.2	2.2 ± 2.0
	conic	3.5 ± 0.5	3.7 ± 0.5	8.7 ± 2.6	94.1 ± 42.7
500	hinge	6.2 ± 1.4	6.2 ± 1.5	5.8 ± 0.8	11.3 ± 2.0
	conic	14.7 ± 4.7	14.5 ± 4.7	20.6 ± 4.9	152.8 ± 25.8
1,000	hinge	0.6 ± 0.1	0.7 ± 0.1	1.1 ± 0.1	2.0 ± 0.4
	conic	18.5 ± 2.3	20.5 ± 2.6	25.2 ± 9.3	167.2 ± 62.5

5.2. **Real instances.** We test the proposed methods in five datasets from the UCI machine learning repository [15], whose names and dimensions (along with computational results) are reported in Table 3. We split each dataset randomly with 35% of the observations in a training set, 35% in a validation set and 30% in the testing set. Additionally, given a noise parameter $\tau \in [0, 0.5)$, we introduce outliers by flipping the label of each observation of the training and validation set with probability τ (the test set is not corrupted by outliers). We then perform cross-validation for the hinge and conic loss as described in §5.1.2. For each dataset and value of the noise parameter τ, we perform 20 random splits of train/validation/test and report average results and standard deviations over these replications.

While the results are highly dependent on the specific dataset (e.g., hinge is better across all noise parameters for the sonar dataset; conic is better across all parameters for the breast cancer dataset), we observe that on average SVMs with the hinge loss perform better in instances without outlier $\tau = 0$, while SVMs with the conic loss are better in settings with outliers ($\tau \geq 0.2$). The results are thus consistent with those observed for synthetic data. Moreover, we also comment that, regardless of the value of the noise parameter τ, using the conic loss results in smaller standard deviations. As discussed in Figure 2, large standard deviations indicate poor performance in some instances, thus the smaller standard deviations suggest that the conic loss results in estimators that are more robust to the specific selection of training/validation/testing sets.

TABLE 3. Results with real datasets, where τ is the probability of a label swap and each row shows averages and standard deviations over 20 splits of training/validation/testing. Column "time" represents the total time required solve 100 SVM problems (required for cross-validation). We depict in bold the best average out-of-sample accuracy for each dataset-noise combination.

name	**n**	**p**	**method**	**time (s)**	**out-of-sample misclassification rate**			
					$\boldsymbol{\tau = 0.0}$	$\boldsymbol{\tau = 0.1}$	$\boldsymbol{\tau = 0.2}$	$\boldsymbol{\tau = 0.3}$
Breast cancer	196	36	hinge	0.1 ± 0.0	$16.9\% \pm 4.4\%$	$18.3\% \pm 4.6\%$	$17.9\% \pm 5.0\%$	$19.4\% \pm 7.7\%$
			conic	45.4 ± 4.1	$\mathbf{16.8\% \pm 4.2\%}$	$\mathbf{17.6\% \pm 4.5\%}$	$\mathbf{17.5\% \pm 4.4\%}$	$\mathbf{18.8\% \pm 5.5\%}$
German credit	1,000	24	hinge	3.1 ± 0.3	$\mathbf{25.0\% \pm 1.9\%}$	$\mathbf{25.8\% \pm 2.3\%}$	$\mathbf{27.5\% \pm 2.0\%}$	$\mathbf{29.4\% \pm 2.4\%}$
			conic	72.0 ± 6.2	$29.6\% \pm 2.1\%$	$29.4\% \pm 2.3\%$	$29.7\% \pm 2.2\%$	$29.5\% \pm 1.9\%$
Immunotherapy	90	6	hinge	0.1 ± 0.0	$\mathbf{20.5\% \pm 8.0\%}$	$24.1\% \pm 9.1\%$	$28.6\% \pm 10.4\%$	$34.6\% \pm 11.9\%$
			conic	2.5 ± 0.1	$20.7\% \pm 6.3\%$	$\mathbf{20.7\% \pm 6.3\%}$	$\mathbf{21.1\% \pm 6.0\%}$	$\mathbf{27.0\% \pm 16.6\%}$
Ionosphere	351	34	hinge	0.5 ± 0.0	$16.0\% \pm 3.3\%$	$19.2\% \pm 4.4\%$	$20.9\% \pm 5.0\%$	$\mathbf{24.0\% \pm 6.9\%}$
			conic	82.0 ± 6.0	$\mathbf{15.8\% \pm 3.1\%}$	$\mathbf{18.0\% \pm 3.6\%}$	$\mathbf{20.5\% \pm 4.8\%}$	$24.1\% \pm 6.4\%$
Sonar	208	60	hinge	0.3 ± 0.0	$\mathbf{25.4\% \pm 7.3\%}$	$\mathbf{30.0\% \pm 7.6\%}$	$\mathbf{32.7\% \pm 7.5\%}$	$\mathbf{42.3\% \pm 10.7\%}$
			conic	202.8 ± 15.1	$30.1\% \pm 6.4\%$	$31.9\% \pm 6.1\%$	$37.3\% \pm 9.5\%$	$45.6\% \pm 6.8\%$
Average			hinge	-	$\mathbf{20.8\% \pm 5.0\%}$	$\mathbf{23.5\% \pm 5.6\%}$	$25.5\% \pm 6.0\%$	$29.9\% \pm 7.9\%$
			conic	-	$22.6\% \pm 4.2\%$	$\mathbf{23.5\% \pm 4.6\%}$	$\mathbf{25.2\% \pm 5.4\%}$	$\mathbf{29.0\% \pm 7.4\%}$

6. Conclusions

We developed an approach for training SVMs, by studying relaxations for mixed-integer optimization formulations of SVMs with the 0-1 loss. The resulting training problems are convex, amenable to solve using existing off-the-shelf solvers, and results in better performance (compared with the commonly used convex estimator involving the hinge loss) in settings with noise and outliers.

Acknowledgments

Andrés Gómez is supported in part by grant FA9550-22-1-0369 from the Air Force Office of Scientific Research.

References

[1] Alizadeh, F. and Goldfarb, D. (2003). Second-order cone programming. *Mathematical Programming*, 95:3–51.

[2] Atamtürk, A. and Gómez, A. (2019). Rank-one convexification for sparse regression. *arXiv preprint arXiv:1901.10334*.

[3] Atamtürk, A. and Gómez, A. (2020). Safe screening rules for L0-regression from perspective relaxations. In *International Conference on Machine Learning*, pages 421–430. PMLR.

[4] Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156.

[5] Bertsimas, D., Cory-Wright, R., and Johnson, N. A. (2023). Sparse plus low rank matrix decomposition: A discrete optimization approach. *Journal of Machine Learning Research*, 24:1–51.

[6] Bertsimas, D., Cory-Wright, R., and Pauphilet, J. (2022). Mixed-projection conic optimization: A new paradigm for modeling rank constraints. *Operations Research*, 70(6):3321–3344.

[7] Bertsimas, D. and Van Parys, B. (2020). Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *The Annals of Statistics*, 48(1):300–323.

[8] Brooks, J. P. (2011). Support vector machines with the ramp loss and the hard margin loss. *Operations Research*, 59(2):467–479.

[9] Carrizosa, E. and Morales, D. R. (2013). Supervised classification and mathematical optimization. *Computers & Operations Research*, 40(1):150–165.

[10] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20:273–297.

[11] d'Aspremont, A., El Ghaoui, L., Jordan, M., and Lanckriet, G. (2004). A direct formulation for sparse PCA using semidefinite programming. *Advances in Neural Information Processing Systems*, 17.

[12] De Rosa, A. and Khajavirad, A. (2022). The ratio-cut polytope and k-means clustering. *SIAM Journal on Optimization*, 32(1):173–203.

[13] Dey, S. S., Molinaro, M., and Wang, G. (2023). Solving sparse principal component analysis with global support. *Mathematical Programming*, 199(1-2):421–459.

[14] Dong, H., Chen, K., and Linderoth, J. (2015). Regularization vs. relaxation: A conic optimization perspective of statistical variable selection. *arXiv preprint arXiv:1510.06083*.

[15] Dua, D. and Graff, C. (2017). Uci machine learning repository.

[16] Ghosh, A., Manwani, N., and Sastry, P. (2015). Making risk minimization tolerant to label noise. *Neurocomputing*, 160:93–107.

[17] Gómez, A. (2021). Outlier detection in time series via mixed-integer conic quadratic optimization. *SIAM Journal on Optimization*, 31(3):1897–1925.

[18] Guan, W., Gray, A., and Leyffer, S. (2009). Mixed-integer support vector machine. In *NIPS workshop on optimization for machine learning*.

[19] Günlük, O. and Linderoth, J. (2010). Perspective reformulations of mixed integer nonlinear programs with indicator variables. *Mathematical Programming*, 124:183–205.

[20] Hazimeh, H., Mazumder, R., and Saab, A. (2022). Sparse regression at scale: Branch-and-bound rooted in first-order optimization. *Mathematical Programming*, 196(1-2):347–388.

[21] Kim, J., Tawarmalani, M., and Richard, J.-P. P. (2022). Convexification of permutation-invariant sets and an application to sparse principal component analysis. *Mathematics of Operations Research*, 47(4):2547–2584.

[22] Li, Y. and Xie, W. (2020). Exact and approximation algorithms for sparse PCA. *arXiv preprint arXiv:2008.12438*.

[23] Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). Applications of second-order cone programming. *Linear algebra and its applications*, 284:193–228.

[24] Maldonado, S., Pérez, J., Weber, R., and Labbé, M. (2014). Feature selection for support vector machines via mixed integer linear programming. *Information sciences*, 279:163–175.

[25] Manwani, N. and Sastry, P. (2013). Noise tolerance under risk minimization. *IEEE Transactions on Cybernetics*, 43(3):1146–1151.

[26] Mason, L., Baxter, J., Bartlett, P., and Frean, M. (1999). Boosting algorithms as gradient descent. *Advances in Neural Information Processing Systems*, 12.

[27] Miller, A. (2002). *Subset selection in Regression*. Chapman and Hall/CRC.

[28] Peng, J. and Xia, Y. (2005). A new theoretical framework for k-means-type clustering. *Foundations and advances in data mining*, pages 79–96.

[29] Pilanci, M., Wainwright, M. J., and El Ghaoui, L. (2015). Sparse learning via boolean relaxations. *Mathematical Programming*, 151(1):63–87.

[30] Rudin, C. (2022). Why black box machine learning should be avoided for high-stakes decisions, in brief. *Nature Reviews Methods Primers*, 2(1):81.

[31] Shen, X., Tseng, G. C., Zhang, X., and Wong, W. H. (2003). On ψ-learning. *Journal of the American Statistical Association*, 98(463):724–734.

[32] Sion, M. (1958). On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171 – 176.

[33] Song, Q., Hu, W., and Xie, W. (2002). Robust support vector machine with bullet hole image classification. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)*, 32(4):440–448.

[34] Ustun, B., Traca, S., and Rudin, C. (2013). Supersparse linear integer models for predictive scoring systems. *arXiv preprint arXiv:1306.5860*.

[35] Wu, Y. and Liu, Y. (2007). Robust truncated hinge loss support vector machines. *Journal of the American Statistical Association*, 102(479):974–983.

[36] Xie, W. and Deng, X. (2020). Scalable algorithms for the sparse ridge regression. *SIAM Journal on Optimization*, 30(4):3359–3386.

[37] Xu, G., Cao, Z., Hu, B.-G., and Principe, J. C. (2017). Robust support vector machines based on the rescaled hinge loss function. *Pattern Recognition*, 63:139–148.

[38] Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *Annals of statistics*, 38(2):894–942.

Appendix A. Kernel formulations

Consider again the MIO formulation for SVM with the exact 0-1 loss:

$$\min_{\boldsymbol{w},\,\boldsymbol{z}} \ \|\boldsymbol{w}\|_2^2 + \lambda \sum_{i=1}^n z_i \tag{19a}$$

$$\text{s.t.} \ \left(1 - y_i \boldsymbol{x_i}^\top \boldsymbol{w}\right)(1 - z_i) \le 0 \qquad \forall i \in [n] \tag{19b}$$

$$\left(1 - y_i \boldsymbol{x_i}^\top \boldsymbol{w}\right) z_i \ge 0 \qquad \forall i \in [n] \tag{19c}$$

$$\boldsymbol{w} \in \mathbb{R}^p,\, \boldsymbol{z} \in \{0,1\}^n. \tag{19d}$$

We now use the approach discussed in [8], which involves the substitution $\boldsymbol{w} = \sum_{i=1}^n y_i \boldsymbol{x_i} \alpha_i$ for some variables α_i, resulting in the formulation

$$\min_{\boldsymbol{\alpha},\,\boldsymbol{z}} \ \sum_{i=1}^n \sum_{j=1}^n y_i y_j \boldsymbol{x_i}^\top \boldsymbol{x_j} \alpha_i \alpha_j + \lambda \sum_{i=1}^n z_i \tag{20a}$$

$$\text{s.t.} \ \left(1 - \sum_{j=1}^n y_i y_j \boldsymbol{x_i}^\top \boldsymbol{x_j} \alpha_j\right)(1 - z_i) \le 0 \qquad \forall i \in [n] \tag{20b}$$

$$\left(1 - \sum_{j=1}^n y_i y_j \boldsymbol{x_i}^\top \boldsymbol{x_j} \alpha_j\right) z_i \ge 0 \qquad \forall i \in [n] \tag{20c}$$

$$\boldsymbol{\alpha} \in \mathbb{R}^n, \boldsymbol{z} \in \{0,1\}^n. \tag{20d}$$

The proof that formulations (19) and (20) are indeed equivalent can be found in [8]. The key idea is that inner products $\boldsymbol{x_i}^\top \boldsymbol{x_j}$ can be replaced with $k(\boldsymbol{x_i}, \boldsymbol{x_j})$ for a given kernel function $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$, and if the kernel is positive-definite (as is the case with commonly used kernel functions) then the objective (20a) remains positive semidefinite.

The method proposed in the paper can be readily applied to formulation (20). Indeed, the counterpart of formulation (15) is

$$\begin{aligned}\max_{\boldsymbol{\gamma}\in\Gamma} \min_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^n \\ \boldsymbol{z}\in[0,1]^n}} & \sum_{i=1}^n \sum_{j=1}^n y_i y_j k(\boldsymbol{x_i}, \boldsymbol{x_j}) \alpha_i \alpha_j + \lambda \sum_{i=1}^n z_i - \sum_{i=1}^n \gamma_i \left(1 - \sum_{j=1}^n y_i y_j k(\boldsymbol{x_i}, \boldsymbol{x_j}) \alpha_j\right)^2 \\ & + \sum_{i=1}^n \gamma_i \left(\frac{\left(1 - \sum_{j=1}^n y_i y_j k(\boldsymbol{x_i}, \boldsymbol{x_j}) \alpha_j\right)_+^2}{z_i} + \frac{\left(1 - \sum_{j=1}^n y_i y_j k(\boldsymbol{x_i}, \boldsymbol{x_j}) \alpha_j\right)_-^2}{1 - z_i} \right),\end{aligned} \tag{21}$$

where $\Gamma \subseteq \mathbb{R}^n_+$ is the set that ensures that the quadratic term in the first line is positive semidefinite. Defining $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ as the kernel matrix whose (i,j)-entry is $y_i y_j k(\boldsymbol{x_i}, \boldsymbol{x_j})$, and letting $\boldsymbol{K_i}$ denote the i-th column of this matrix, we find that Γ is defined by nonnegativity constraints and the conic

constraint

$$\boldsymbol{K} - \sum_{i=1}^{n} \gamma_i \boldsymbol{K_i} \boldsymbol{K_i}^\top \succeq 0.$$

Finally, a formulation analogous to the one in Theorem 2 is

$$\begin{aligned}
\min_{\boldsymbol{\alpha},\boldsymbol{z},\boldsymbol{A}} \ & \langle \boldsymbol{K}, \boldsymbol{A} \rangle + \lambda \sum_{i=1}^{n} z_i \\
\text{s.t. } & \langle \boldsymbol{K_i} \boldsymbol{K_i}^\top, \boldsymbol{A} \rangle - 2\boldsymbol{K_i}^\top \boldsymbol{\alpha} + 1 \geq \frac{(1 - \boldsymbol{K_i}^\top \boldsymbol{\alpha})_+^2}{z_i} + \frac{(1 - \boldsymbol{K_i}^\top \boldsymbol{\alpha})_-^2}{1 - z_i} \quad \forall i \in [n] \\
& \begin{pmatrix} 1 & \boldsymbol{\alpha}^\top \\ \boldsymbol{\alpha} & \boldsymbol{A} \end{pmatrix} \succeq 0, \ \boldsymbol{\alpha} \in \mathbb{R}^n, \ \boldsymbol{A} \in \mathbb{R}^{n \times n} \\
& \boldsymbol{z} \in [0,1]^n.
\end{aligned}$$

Appendix B. Additional computational results with separable synthetic data and label noise

We also tested the proposed methods in problems where the data is originally completely separable, but the labels of some points are flipped, obfucating the learning process. Specifically, let $0 \leq \tau \leq 0.5$ be a noise parameter, and let $n, p \in \mathbb{Z}_+$ be dimension parameters. To create the instances we first generate a "true" hyperplane $\bar{\boldsymbol{w}} \in \mathbb{R}^{p+1}$ separating the data, where each entry of $\bar{\boldsymbol{w}}$ is generated independently from a uniform distribution in $[-1, 1]$. Then we generate n points $x_i \in \mathbb{R}^{p+1}$ where $(x_i)_1 = 1$ (to account for the intercept) and each $(x_i)_j$ is generated independently from a uniform distribution in $[-1, 1]$. If $\bar{\boldsymbol{w}}^\top \boldsymbol{x_i} \geq 0$ we set label $y_i = 1$ with probability $1 - \tau$ and $y_i = -1$ otherwise; and if $\bar{\boldsymbol{w}}^\top \boldsymbol{x_i} < 0$ we set label $y_i = -1$ with probability $1 - \tau$ and $y_i = 1$ otherwise. Then we generate training, validation and testing sets and perform crossvalidation with the hinge and conic loss, as discussed in §5.1.2. The results are summarized in Table 4.

Table 4. Out-of-sample misclassification rate in instances of type 1, as a function of the noise parameter τ.

method	$\boldsymbol{\tau = 0.1}$	$\boldsymbol{\tau = 0.2}$	$\boldsymbol{\tau = 0.3}$
hinge	$2.7\% \pm 2.5\%$	$4.9\% \pm 5.4\%$	$7.1\% \pm 7.2\%$
conic	$3.1\% \pm 3.1\%$	$4.5\% \pm 4.5\%$	$6.4\% \pm 6.6\%$

The results are consistent with computational results presented elsewhere in the paper: the hinge loss exhibits slightly better performance in noise-free regimes, but is inferior in the presence of outliers.

Appendix C. Cross-validation for SVMs with the conic loss

In practice, we would like to solve (17) for several values of λ and choose the best hyperparameter via cross-validation. However, the magnitude of the best parameters is often unknown and data-dependent. When doing cross-validation, instead of directly using (17), we remove the penalty term $\lambda \sum_{i=1}^{n} z_i$ from the objective and instead add the constraint

$$\sum_{i=1}^{n} z_i \leq \kappa n,$$

where $\kappa \in [0, 1]$ can be loosely interpreted as the proportion of points that can be misclassified. Since the problems are convex, the two problems are equivalent in the sense that for every κ there exists a λ that delivers the same solution, and vice versa. However, by this transformation, when performing cross-validation, we simply select values of κ uniformly in the interval $[0, 0.5]$, matching the prior that there should never be more than 50% misclassified points and thus testing only relevant values of the hyperparameters. We stress that the reason this approach works for the proposed conic formulation (but not for the Hinge loss) is that the formulation is a stronger formulation of the exact 0-1 loss problem (3) (but with a cardinality constraint), thus hyperparameters such as κ retain approximately their conceptual meanings.