Journal of Machine Learning Research 15 (2014) 2185-2211 Submitted 7/13; Revised 11/13; Published 6/14

Ramp Loss Linear Programming Support Vector Machine

Xiaolin Huang HUANGXL06@MAILS.TSINGHUA.EDU.CN
Department of Electrical Engineering, ESAT-STADIUS, KU Leuven
Kasteelpark Arenberg 10, Leuven, B-3001, Belgium
Lei Shi LEISHI@FUDAN.EDU.CN
Department of Electrical Engineering, ESAT-STADIUS, KU Leuven
School of Mathematical Sciences, Fudan University, Shanghai, 200433, P.R. China
Johan A.K. Suykens JOHAN.SUYKENS@ESAT.KULEUVEN.BE
Department of Electrical Engineering, ESAT-STADIUS, KU Leuven
Kasteelpark Arenberg 10, Leuven, B-3001, Belgium

Editor: Mikhail Belkin

Abstract

The ramp loss is a robust but non-convex loss for classification. Compared with other non-convex losses, a local minimum of the ramp loss can be effectively found. The effectiveness of local search comes from the piecewise linearity of the ramp loss. Motivated by the fact that the ℓ_1-penalty is piecewise linear as well, the ℓ_1-penalty is applied for the ramp loss, resulting in a ramp loss linear programming support vector machine (ramp-LPSVM). The proposed ramp-LPSVM is a piecewise linear minimization problem and the related optimization techniques are applicable. Moreover, the ℓ_1-penalty can enhance the sparsity. In this paper, the corresponding misclassification error and convergence behavior are discussed. Generally, the ramp loss is a truncated hinge loss. Therefore ramp-LPSVM possesses some similar properties as hinge loss SVMs. A local minimization algorithm and a global search strategy are discussed. The good optimization capability of the proposed algorithms makes ramp-LPSVM perform well in numerical experiments: the result of ramp-LPSVM is more robust than that of hinge SVMs and is sparser than that of ramp-SVM, which consists of the $\|\cdot\|_{\mathcal{K}}$-penalty and the ramp loss.

Keywords: support vector machine, ramp loss, ℓ_1-regularization, generalization error analysis, global optimization

1. Introduction

In a binary classification problem, the input space is a compact subset $X \subset \mathbb{R}^n$ and the output space $Y = \{-1, 1\}$ represents two classes. Classification algorithms produce binary classifiers $\mathcal{C} : X \to Y$ induced by real-valued functions $f : X \to \mathbb{R}$ as $\mathcal{C} = \mathrm{sgn}(f)$, where the sign function is defined by $\mathrm{sgn}(f(x)) = 1$ if $f(x) \geq 0$ and $\mathrm{sgn}(f(x)) = -1$ otherwise. Since proposed by Cortes and Vapnik (1995), the support vector machine (SVM) has become a popular classification method, because of its good statistical property and generalization capability. SVM is usually based on a Mercer kernel $\mathcal{K}$ to produce non-linear classifiers. Such a kernel is a continuous, symmetric, and positive semi-definite function defined on $X \times X$. Given training data $\mathbf{z} = \{x_i, y_i\}_{i=1}^m$ with $x_i \in X, y_i \in Y$ and a loss function $L : \mathbb{R} \to \mathbb{R}^+$, in the functional analysis setting, SVM can be formulated as the following

optimization problem

$$\min_{f\in\mathcal{H}_\mathcal{K},b\in\mathbb{R}} \quad \frac{\mu}{2}\|f\|_\mathcal{K}^2 + \frac{1}{m}\sum_{i=1}^{m} L(1-y_i(f(x_i)+b)), \tag{1}$$

where $\mathcal{H}_\mathcal{K}$ is the Reproducing Kernel Hilbert Space (RKHS) induced by the Mercer kernel $\mathcal{K}$ with the norm $\|\cdot\|_\mathcal{K}$ (Aronszajn, 1950) and $\mu > 0$ is a trade-off parameter. The constant term b is called offset, which leads to much flexibility. The corresponding binary classifier is evaluated based on the optima of (1) by its sign function. Traditionally, the hinge loss $L_{\text{hinge}}(u) = \max\{u, 0\}$ is used. Besides, the squared hinge loss (Vapnik, 1998) and the least squares loss (Suykens and Vandewalle, 1999; Suykens et al., 2002) also have been widely applied. In classification and the related methodologies, robustness to outliers is always an important issue. The influence function (see, e.g., Steinwart and Christmann, 2008; De Brabanter et al., 2009) related to the hinge loss is bounded, which means that the effect of outliers on the result of minimizing the hinge loss is bounded. Though the effect is bounded, it can be significantly large since the penalty given to the outliers by the hinge loss is quite huge. In fact, any convex loss is unbounded. To remove the effect of outliers, researchers turn to some non-convex losses, such as the hard-margin loss, the normalized sigmoid loss (Mason et al., 2000), the ψ-learning loss (Shen et al., 2003), and the ramp loss (Collobert et al., 2006a,b). The ramp loss is defined as follows,

$$L_{\text{ramp}}(u) = \begin{cases} L_{\text{hinge}}(u), & u \le 1, \\ 1, & u > 1, \end{cases}$$

which is also called a truncated hinge loss in Wu and Liu (2007). The plots of the mentioned losses are illustrated in Figure 1, showing the robustness of these non-convex losses.

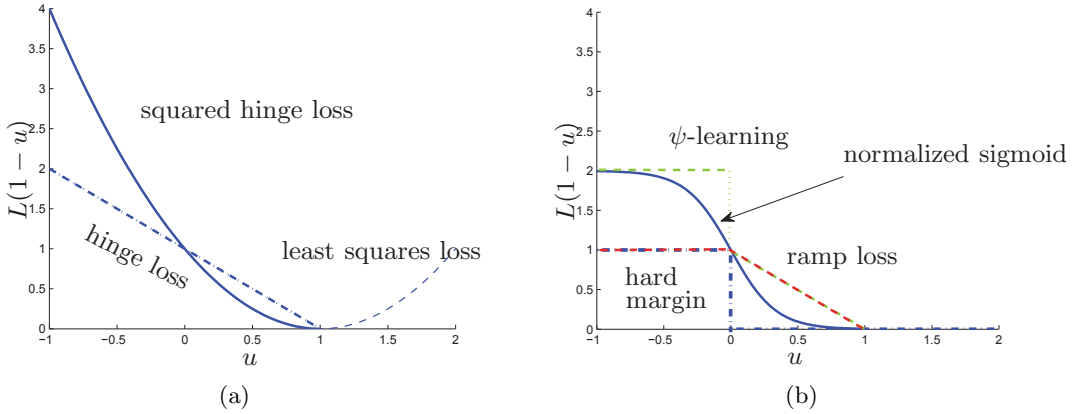

Figure 1: Plots of losses used for classification: (a) convex losses: the hinge loss (dash-dotted line), the squared hinge loss (solid line), and the least squares loss (dashed line); (b) robust but non-convex losses: the hard margin loss (blue dash-dotted line), the ψ-learning loss (green dashed line), the normalized sigmoid loss (blue solid line), and the ramp loss (red dashed line).

Among the mentioned robust but non-convex losses, the ramp loss is an attractive one. Using the ramp loss in (1), one obtains a ramp loss support vector machine (ramp-SVM). Because the ramp loss can be easily written as a difference of convex functions (DC), algorithms based on DC programming are applicable for ramp-SVM. The discussion about DC programming can be found in An et al. (1996), Horst and Thoai (1999), and An and Tao (2005). To apply DC programming in the ramp loss, we first observe the identity

$$L_{\text{ramp}}(u) = \min\{\max\{u, 0\}, 1\} = \max\{u, 0\} - \max\{u - 1, 0\}. \tag{2}$$

Therefore, SVM (1) with $L = L_{\text{ramp}}$ can be decomposed into the convex part $\frac{\mu}{2}\|f\|_{\mathcal{K}}^2 + \frac{1}{m}\sum_{i=1}^m \max\{1 - y_i(f(x_i) + b), 0\}$ and the concave part $-\frac{1}{m}\sum_{i=1}^m \max\{-y_i(f(x_i) + b), 0\}$. Hence DC programming can be used for finding a local minimizer of this problem, which has been applied by Collobert et al. (2006a), Wu and Liu (2007). DC programming for ramp-SVM is also referred to as a concave-convex procedure by Yuille and Rangarajan (2003). Besides the continuous optimization methods, ramp-SVM has been formulated as a mixed integer optimization problem by Brooks (2011) as below,

$$\begin{array}{ll} \min\limits_{f\in\mathcal{H}_{\mathcal{K}}, b\in\mathbb{R}, \omega} & \frac{\mu}{2}\|f\|_{\mathcal{K}}^2 + \frac{1}{m}\sum_{i=1}^m (e_i + \omega_i) \\ \text{s.t.} & \omega_i \in \{0, 1\}, \\ & 0 \le e_i \le 1, \quad i = 1, \cdots, m, \\ & y_i(f(x_i) + b) \ge 1 - e_i, \quad \text{if } \omega_i = 0. \end{array} \tag{3}$$

The optimization problem (3) should be solved over all possible binary vectors $\omega = [\omega_1, \ldots, \omega_m]^T \in \{0, 1\}^m$. Once the binary vector ω is given, this problem can be solved by quadratic programming. Consequently, when the size of the problem grows, the computation time explodes.

It is worth noting the case of taking $L = L_{\text{hinge}}$ in (1). It corresponds to the well-known C-SVM. One can solve C-SVM by its dual form, then the output function is represented as $\sum_{i=1}^m \nu_i^* y_i \mathcal{K}(x, x_i) + b^*$, where $[\nu_1^*, \cdots, \nu_m^*]^T$ is the optimal solution of

$$\begin{array}{ll} \min\limits_{\nu_i\in\mathbb{R}} & \frac{1}{2}\sum_{i,j=1}^m \nu_i\nu_j y_i y_j \mathcal{K}(x_i, x_j) - \sum_{i=1}^m \nu_i \\ \text{s.t.} & \sum_{i=1}^m \nu_i y_i = 0, \\ & 0 \le \nu_i \le \frac{1}{\mu m}, \quad i = 1, \cdots, m. \end{array}$$

The optimal offset b^* can be computed from the Karush-Kuhn-Tucker (KKT) conditions after $\{\nu_i^*\}_{i=1}^m$ is found (see, e.g., Suykens et al., 2002). From the dual form of C-SVM, we find that though we search the function f in a rather large space $\mathcal{H}_{\mathcal{K}}$, the optimal solution actually belongs to a finite-dimensional subspace given by $\mathcal{H}_{\mathcal{K},\mathbf{z}}^+$ with

$$\mathcal{H}_{\mathcal{K},\mathbf{z}}^+ = \left\{\sum_{i=1}^m \alpha_i y_i \mathcal{K}(x, x_i), \forall \alpha = [\alpha_1, \ldots, \alpha_m]^T \succeq 0\right\}.$$

Here the notation $\succeq 0$ means all the elements of the vector being non-negative.

To enhance the sparsity in the output function, the linear programming support vector machine (LPSVM) directly minimizes the data fitting term $\frac{1}{m}\sum_{i=1}^m L_{\text{hinge}}(1 - y_i(f(x_i) + b))$ with a ℓ_1-penalty term (see Vapnik, 1998; Smola et al., 1999). Given $f \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^+$, the ℓ_1-penalty is defined as

$$\Omega(f) = \sum_{i=1}^m \alpha_i, \text{ for } f = \sum_{i=1}^m \alpha_i y_i \mathcal{K}(x, x_i), \tag{4}$$

which is the ℓ_1-penalty of the combinatorial coefficients of f. Then LPSVM can be formulated as follows,

$$\min_{f\in\mathcal{H}_{\mathcal{K}}^{+},b\in\mathbb{R}} \quad \mu\Omega(f)+\frac{1}{m}\sum_{i=1}^{m}L_{\text{hinge}}(1-y_i(f(x_i)+b)). \tag{5}$$

LPSVM is also related to 1-norm SVM proposed by Zhu et al. (2004), which searches a linear combination of basis functions and does not consider the non-negative constraint. The properties of LPSVM have been demonstrated in the literature (e.g., Bradley and Mangasarian, 2000; Kecman and Hadzic, 2000). Generalization error analysis for LPSVM can be found in Wu and Zhou (2005).

For problem (1), one can choose different penalty terms and different loss functions. For example, using $\|f\|_{\mathcal{K}}$ together with the hinge loss, we obtain C-SVM. The property of C-SVM can be observed from the properties of $\|f\|_{\mathcal{K}}$ and the hinge loss: since $\|f\|_{\mathcal{K}}$ is a quadratic function and the hinge loss is piecewise linear (pwl), the objective function of C-SVM is piecewise quadratic (pwq) and can be solved by constrained quadratic programming. For LPSVM, which consists of the ℓ_1-penalty and the hinge loss, the objective function is convex piecewise linear and hence can be minimized by linear programming. In Table 1, we summarize the properties of several penalties and losses.

	$\|f\|_{\mathcal{K}}$	$\Omega(f)$	hinge	squared hinge	least squares	ψ-learning	normalized sigmoid	ramp
function type	quadratic	pwl	pwl	pwq	quadratic	discontinuous	log	pwl
convexity	√	√	√	√	√	×	×	×
continuity	√	√	√	√	√	×	√	√
smoothness	√	×	×	√	√	×	√	×
sparsity	×	√	√	√	×	√	×	√
bounded influence fun.	—	—	√	×	×	√	√	√
bounded penalty value	—	—	×	×	×	√	√	√

* "pwl" stands for piecewise linear; "pwq" stands for piecewise quadratic.

Table 1: Properties of Different Penalties and Losses

The ramp loss gives a constant penalty for any large outlier and it is obviously robust. From Table 1, we observe that both $\Omega(f)$ and the ramp loss are continuous piecewise linear. It follows that if we choose $\Omega(f)$ and the ramp loss, the objective function of (1) is continuous piecewise linear and can be minimized by linear programming. Besides, minimizing $\Omega(f)$ enhances the sparsity. Motivated by this observation, in this paper we study the binary classifiers generated by minimizing the ramp loss and the ℓ_1-penalty, which is called a ramp loss linear programming support vector machine (ramp-LPSVM). The ramp-LPSVM has the following formulation,

$$(f^*_{\mathbf{z},\mu},b^*_{\mathbf{z},\mu})=\operatorname*{argmin}_{f\in\mathcal{H}_{\mathcal{K},\mathbf{z}}^{+},b\in\mathbb{R}} \quad \mu\Omega(f)+\frac{1}{m}\sum_{i=1}^{m}L_{\text{ramp}}(1-y_i(f(x_i)+b)), \tag{6}$$

where $\Omega(\cdot)$ is the ℓ_1-penalty defined by (4). And the induced classifier is given by $\mathrm{sgn}(f^*_{\mathbf{z},\mu} + b^*_{\mathbf{z},\mu})$. We call (6) ramp-LPSVM, which implies that the algorithm proposed later involves linear programming problems. Similarly to ramp-SVM, the proposed ramp-LPSVM enjoys robustness. Moreover, it can give a sparser solution. In addition to enhancing the sparsity, replacing the $\|\cdot\|_{\mathcal{K}}$-penalty in ramp-SVM by the ℓ_1-penalty is mainly motivated by the fact that both the ramp loss and the ℓ_1-penalty are piecewise linear, which helps developing more efficient algorithms.

Resulting from the identity (2), the problem related to ramp-LPSVM leads to a polyhedral concave problem, which minimizes a concave function on one polyhedron. A polyhedral concave problem is easier to handle than a regular non-convex problem and some efficient methods were reviewed by Horst and Hoang (1996). Moreover, ramp-LPSVM (6) has a piecewise linear objective function. For such kind of problems, a hill detouring technique proposed by Huang et al. (2012a) has shown good global search capability. As the name suggests, the hill detouring method searches on the level set to escape from a local optimum. One contribution of this paper is that we establish algorithms for solving ramp-LPSVM (6), including DC programming for local minimization and hill detouring for global search. Additionally, we investigate the asymptotic performance of ramp-LPSVM under the framework of statistical learning theory. Our analysis implies that ramp-LPSVM has a similar misclassification error bound and similar convergence behavior as C-SVM. Moreover, one can expect that the output binary classifier of algorithm (6) is robust, due to the ramp loss, and has a sparse representation, due to the ℓ_1-penalty.

The remainder of the paper is organized as follows: some statistical properties for the proposed ramp-LPSVM are discussed in Section 2. In Section 3, we establish problem-solving algorithms including DC programming for local minimization, and hill detouring for escaping from local optima. The proposed algorithms are tested then on numerical experiments in Section 4. Section 5 ends the paper with concluding remarks.

2. Theoretical Properties

In this section, we establish the theoretical analysis for ramp-LPSVM under the framework of statistical learning theory. In the following, we first show that the ramp loss is classification calibrated; see Proposition 1. In other works, we prove that minimizing the ramp loss results in the Bayes classifier. After that, an inequality is presented in Theorem 2 to bound the difference between the risk of the Bayes classifier and that of the classifier induced from minimizing the ramp loss. Finally, we obtain the convergence behavior of ramp-LPSVM, which is given in Theorem 5. To prove Theorem 5, error decomposition theorems for ramp-SVM and ramp-LPSVM are discussed. The analysis on the ramp loss is closely related to the properties of the hinge loss, because the ramp loss can be regarded as a truncated hinge loss. In our analysis, the global minimizer of the ramp loss plays an important role, which motivates us to establish a global search strategy in the next section.

To this end, we assume that the sample $\mathbf{z} = \{x_i, y_i\}_{i=1}^m$ is independently drawn from a probability measure ρ on $X \times Y$. The misclassification error for a binary classifier $\mathcal{C} : X \to Y$ is defined as the probability of the event $\mathcal{C}(x) \neq y$:

$$\mathcal{R}(\mathcal{C}) = \int_{X\times Y} \mathcal{I}_{y\neq\mathcal{C}(x)} d\rho = \int_X \rho(y \neq \mathcal{C}(x)|x) d\rho_X,$$

where $\mathcal{I}$ is the indicator function, ρ_X is the marginal distribution of ρ on X, and $\rho(y|x)$ is the conditional distribution of ρ at given x. It should be pointed out that $\rho(y|x)$ is a binary distribution, which is given by $\text{Prob}(y = 1|x)$ and $\text{Prob}(y = -1|x)$. The classifier that minimizes the misclassification error is the Bayes rule f_c, which is defined as,

$$f_c = \arg \min_{\mathcal{C}:X\to Y} \mathcal{R}(\mathcal{C}).$$

The Bayes rule can be evaluated as

$$f_c(x) = \begin{cases} 1, & \text{if } \text{Prob}(y = 1|x) \geq \text{Prob}(y = -1|x), \\ -1, & \text{if } \text{Prob}(y = 1|x) < \text{Prob}(y = -1|x). \end{cases}$$

The performance of a binary classifier induced by a real-valued function f is measured by the excess misclassification error $\mathcal{R}(\text{sgn}(f)) - \mathcal{R}(f_c)$. Let $f_{\mathbf{z},\mu} = f^*_{\mathbf{z},\mu} + b^*_{\mathbf{z},\mu}$ with $(f^*_{\mathbf{z},\mu}, b^*_{\mathbf{z},\mu})$ being the global minimizer of ramp-LPSVM (6). The purpose of the theoretical analysis is to estimate $\mathcal{R}(\text{sgn}(f_{\mathbf{z},\mu})) - \mathcal{R}(f_c)$ as the sample size m tends to infinity. Convergence rates will be derived under the choice of the parameter μ and conditions on the distribution ρ.

As an important ingredient in classification algorithms, the loss function L is used to model the target function of interest. Concretely, the target function denoted as $f_{L,\rho}$ minimizes the expected L-risk

$$\mathcal{R}_{L,\rho}(f) = \int_{X\times Y} L(1 - yf(x))d\rho$$

over all possible functions $f : X \to \mathbb{R}$ and can be defined pointwisely as below,

$$f_{L,\rho}(x) = \arg\min_{t\in\mathbb{R}} \int_Y L\,(1 - yt)\, d\rho(y|x),\ \forall x \in X.$$

The basic idea on designing algorithms is to replace the unknown true risk $\mathcal{R}_{L,\rho}$ by the empirical L-risk

$$\mathcal{R}_{L,\mathbf{z}}(f) = \frac{1}{m}\sum_{i=1}^{m} L(1 - y_i f(x_i)), \tag{7}$$

and to minimize this empirical risk (or its penalized version) over a suitable function class. When the hard margin loss, which counts the number of misclassification,

$$L_{\text{mis}}(u) = \begin{cases} 0, & u \geq 0, \\ 1, & u < 0, \end{cases}$$

is used, one can check that for any binary classifier $\mathcal{C} : X \to Y$, there holds $\mathcal{R}(\mathcal{C}) = \mathcal{R}_{L_{\text{mis}},\rho}(\mathcal{C})$. Therefore, the excess misclassification error can be written as

$$\mathcal{R}_{L_{\text{mis}},\rho}(\text{sgn}(f)) - \mathcal{R}_{L_{\text{mis}},\rho}(f_c).$$

However, the empirical algorithms based on L_{mis} will lead to NP-hard optimization problems, and thus it is not computationally realizable. One way to resolve this issue is to use surrogate loss functions as discussed in Section 1, and then to minimize the empirical

risk associated with the used surrogate loss. Among these losses, the hinge loss plays an important role, since one has $f_{L_{\text{hinge}},\rho} = f_c$.

Now we investigate the ramp loss. For a given $x \in X$, a simple calculation shows that

$$\begin{aligned} & \int_Y L_{\text{ramp}}\,(1-yt)\,d\rho(y|x) \\ = \;& L_{\text{ramp}}(1-t)\text{Prob}(y=1|x) + L_{\text{ramp}}(1+t)\text{Prob}(y=-1|x) \\ = \;& \begin{cases} \text{Prob}(y=1|x), & t \le -1, \\ \text{Prob}(y=1|x) + (1+t)\text{Prob}(y=-1|x), & -1 < t \le 0, \\ (1-t)\text{Prob}(y=1|x) + \text{Prob}(y=-1|x), & 0 \le t < 1, \\ \text{Prob}(y=-1|x), & t \ge 1. \end{cases} \end{aligned}$$

Obviously, when $\text{Prob}(y=1|x) > \text{Prob}(y=-1|x)$, the minimal value is $\text{Prob}(y=-1|x)$, which is achieved by $t=1$. When $\text{Prob}(y=1|x) < \text{P}(y=-1|x)$, the minimal value is $\text{Prob}(y=1|x)$, which is achieved by $t=-1$. Therefore, the corresponding target function $f_{L_{\text{ramp}},\rho}$ that minimizes the expected L_{ramp}-risk is the Bayes rule. The discussion above can be concluded in the following proposition.

Proposition 1 *For any measurable function $f : X \to \mathbb{R}$, there holds*

$$\mathcal{R}_{L_{\text{ramp}},\rho}(f) \ge \mathcal{R}_{L_{\text{ramp}},\rho}(f_c).$$

That is, the Bayes rule f_c is a minimizer of the expected L_{ramp}-risk.

Next, for a real-valued function $f : X \to \mathbb{R}$, we consider bounding the excess misclassification error by the generalization error $\mathcal{R}_{L_{\text{ramp}},\rho}(f) - \mathcal{R}_{L_{\text{ramp}},\rho}(f_{L_{\text{ramp}},\rho})$. Such kind of bound plays an essential role in error analysis of classification algorithms. When the loss function is convex and satisfies some regularity conditions, the corresponding bound is the so-called self-calibration inequality and has been established by Bartlett et al. (2006) and Steinwart (2007). For example, a typical result presented in Cucker and Zhou (2007) claims that, if a general loss function satisfies the following conditions:

- $L(1-u)$ is convex with respect to u;
- $L(1-u)$ is differentiable at $u=0$ and $\frac{dL(1-u)}{du}|_{u=0} < 0$;
- $\min\{u : L(1-u) = 0\} = 1$;
- $\frac{d^2L(1-u)}{du^2}|_{u=1} > 0$,

then there exists a constant $c_L > 0$ such that for any measurable function $f : X \to \mathbb{R}$,

$$\mathcal{R}_{L_{\text{mis}},\rho}(\text{sgn}(f)) - \mathcal{R}_{L_{\text{mis}},\rho}(f_c) \le c_L\sqrt{\mathcal{R}_{L,\rho}(f) - \mathcal{R}_{L,\rho}(f_{L,\rho})}. \tag{8}$$

This inequality holds for many loss functions, such as the hinge loss, the squared hinge loss, and the least squares loss. For the hinge loss L_{hinge}, Zhang (2004) gave a tighter bound by the following inequality,

$$\mathcal{R}_{L_{\text{mis}},\rho}(\text{sgn}(f)) - \mathcal{R}_{L_{\text{mis}},\rho}(f_c) \leq \mathcal{R}_{L_{\text{hinge}},\rho}(f) - \mathcal{R}_{L_{\text{hinge}},\rho}(f_{L_{\text{hinge}},\rho}).$$

The improvement is mainly due to the property that $\mathcal{R}_{L_{\text{hinge}},\rho}(f_{L_{\text{hinge}},\rho}) = \mathcal{R}_{L_{\text{hinge}},\rho}(f_c)$.

For the ramp loss L_{ramp}, we cannot directly use the conclusion given by (8), since the loss is not convex. However, as L_{ramp} can be considered as a truncated hinge loss and maintains the same property due to Proposition 1, one thus can establish a similar inequality for the ramp loss.

Theorem 2 *For any probability measure ρ and any measurable function $f : X \to \mathbb{R}$,*

$$\mathcal{R}_{L_{\text{mis}},\rho}(\text{sgn}(f)) - \mathcal{R}_{L_{\text{mis}},\rho}(f_c) \leq \mathcal{R}_{L_{\text{ramp}},\rho}(f) - \mathcal{R}_{L_{\text{ramp}},\rho}(f_{L_{\text{ramp}},\rho}). \tag{9}$$

Proof By Proposition 1, we have $\mathcal{R}_{L_{\text{ramp}},\rho}(f_{L_{\text{ramp}},\rho}) = \mathcal{R}_{L_{\text{ramp}},\rho}(f_c)$. Since y and $f_c(x)$ belong to $\{-1, 1\}$, $1-yf_c(x)$ takes value of 0 or 2. We hence have $\mathcal{R}_{L_{\text{mis}},\rho}(f_c) = \mathcal{R}_{L_{\text{ramp}},\rho}(f_c)$, which comes from the fact that

$$L_{\text{mis}}(0) = L_{\text{ramp}}(0) \quad \text{and} \quad L_{\text{mis}}(2) = L_{\text{ramp}}(2).$$

Thus, to prove (9), we need to show that

$$\mathcal{R}_{L_{\text{mis}},\rho}(\text{sgn}(f)) \leq \mathcal{R}_{L_{\text{ramp}},\rho}(f), \tag{10}$$

which is equivalent to

$$\int_{X\times Y} L_{\text{mis}}\Big(1 - y\text{sgn}(f(x))\Big) - L_{\text{ramp}}\Big(1 - yf(x)\Big)d\rho \leq 0.$$

For any y and $f(x)$, if $yf(x) \leq 0$, then $y\text{sgn}(f(x)) \leq 0$, which follows that $L_{\text{mis}}(1 - y\text{sgn}(f(x))) = L_{\text{ramp}}(1 - yf(x)) = 1$. If $yf(x) > 0$, then we have $y\text{sgn}(f(x)) = 1$ and $L_{\text{mis}}(1 - y\text{sgn}(f(x))) = 0$. Since $L_{\text{ramp}}(1 - yf(x))$ is always nonnegative, we have $L_{\text{mis}}(1 - y\text{sgn}(f(x))) - L_{\text{ramp}}(1 - yf(x)) \leq 0$ for this case.

Summarizing the above discussion, we prove (10) and then Theorem 2. ■

From Theorem 2, in order to estimate $\mathcal{R}_{L_{\text{mis}},\rho}(\text{sgn}(f_{\mathbf{z},\mu})) - \mathcal{R}_{L_{\text{mis}},\rho}(f_c)$, we turn to bound $\mathcal{R}_{L_{\text{ramp}},\rho}(f_{\mathbf{z},\mu}) - \mathcal{R}_{L_{\text{ramp}},\rho}(f_c)$. We thus need an error decomposition for the latter. This decomposition process is well-developed in the literature for RKHS-based regularization schemes (see, e.g., Cucker and Zhou, 2007; Steinwart and Christmann, 2008). To explain the details, we take ramp-SVM below as an example. For $\mathbf{z} = \{x_i, y_i\}_{i=1}^m$ and $\lambda > 0$, let $\tilde{f}_{\mathbf{z},\lambda} = \tilde{f}^*_{\mathbf{z},\lambda} + \tilde{b}^*_{\mathbf{z},\lambda}$, where

$$(\tilde{f}^*_{\mathbf{z},\lambda}, \tilde{b}^*_{\mathbf{z},\lambda}) = \underset{f\in\mathcal{H}_K, b\in\mathbb{R}}{\text{argmin}} \ \frac{\lambda}{2}\|f\|_K^2 + \frac{1}{m}\sum_{i=1}^m L_{\text{ramp}}(1 - y_i(f(x_i) + b)). \tag{11}$$

Then the following decomposition holds true:

$$\begin{aligned}\mathcal{R}_{L_{\text{ramp}},\rho}(\tilde{f}_{\mathbf{z},\lambda}) - \mathcal{R}_{L_{\text{ramp}},\rho}(f_c) \;\leq\; & \Big\{\mathcal{R}_{L_{\text{ramp}},\rho}(\tilde{f}_{\mathbf{z},\lambda}) - \mathcal{R}_{L_{\text{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})\Big\} \\ & + \big\{\mathcal{R}_{L_{\text{ramp}},\mathbf{z}}(f_\lambda) - \mathcal{R}_{L_{\text{ramp}},\rho}(f_\lambda)\big\} + \mathcal{A}(\lambda),\end{aligned}$$

where $\mathcal{R}_{L_{\text{ramp}},\mathbf{z}}(f)$ is the empirical L_{ramp}-risk given by (7). The function f_λ depends on λ and is defined by the data-free limit of (11), that is $f_\lambda = f_\lambda^* + b_\lambda^*$ with

$$(f_\lambda^*, b_\lambda^*) = \operatorname*{argmin}_{f\in\mathcal{H}_\mathcal{K}, b\in\mathbb{R}} \; \frac{\lambda}{2}\|f\|_\mathcal{K}^2 + \mathcal{R}_{\text{ramp},\rho}(f+b). \tag{12}$$

The term $\mathcal{A}(\lambda)$ measures the approximation power of the system $(\mathcal{K},\rho)$ and is defined by

$$\mathcal{A}(\lambda) = \inf_{f\in\mathcal{H}_\mathcal{K}, b\in\mathbb{R}} \; \frac{\lambda}{2}\|f\|_\mathcal{K}^2 + \mathcal{R}_{\text{ramp},\rho}(f+b) - \mathcal{R}_{\text{ramp},\rho}(f_c), \quad \forall \lambda > 0. \tag{13}$$

It is easy to establish such kind of decomposition if one notices the fact that both $\tilde{f}_{\mathbf{z},\lambda}$ and f_λ lie in the same function space. However, it is not the case for ramp-LPSVM. The data-dependent nature of $\mathcal{H}_{\mathcal{K},\mathbf{z}}^+$ leads to an essential difficulty in the error analysis. Motivated by Wu and Zhou (2005), we shall establish the error decomposition for ramp-LPSVM (6) with the aid of $\tilde{f}_{\mathbf{z},\lambda}$. To this end, we first show some properties of $\tilde{f}_{\mathbf{z},\lambda}$, which play an important role in our analysis.

Proposition 3 *For any $\lambda > 0$, $(\tilde{f}_{\mathbf{z},\lambda}^*, \tilde{b}_{\mathbf{z},\lambda}^*)$ is given by (11) and $\tilde{f}_{\mathbf{z},\lambda} = \tilde{f}_{\mathbf{z},\lambda}^* + \tilde{b}_{\mathbf{z},\lambda}^*$. Then $\tilde{f}_{\mathbf{z},\lambda}^* \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^+$ and*

$$\Omega(\tilde{f}_{\mathbf{z},\lambda}^*) \leq \lambda^{-1}\mathcal{R}_{L_{\text{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda}) + \|\tilde{f}_{\mathbf{z},\lambda}^*\|_\mathcal{K}^2. \tag{14}$$

Proof Following the idea of Brooks (2011), one can formulate the minimization problem (11) as a mixed integer optimization problem, which is given by (3) with $\mu = \lambda$. We first show that if the binary vector $\omega^* = [\omega_1^*, \cdots, \omega_m^*]^T \in \{0,1\}^m$ is optimal for the optimization problem (3), then the global minimizer of (11) can be obtained by solving the following minimization problem

$$\begin{array}{ll}\min\limits_{f\in\mathcal{H}_\mathcal{K}, e_i, b\in\mathbb{R}} & \frac{\lambda}{2}\|f\|_\mathcal{K}^2 + \frac{1}{m}\sum_{i=1}^m e_i \\ \text{s.t.} & e_i \geq 0, \quad i = 1,\cdots,m, \\ & y_i(f(x_i)+b) \geq 1 - e_i, \quad \text{if } \omega_i^* = 0.\end{array} \tag{15}$$

In fact, when the optimal ω^* is given, the global minimizer of (11) can be solved by the optimization problem (3), which is reduced to

$$\begin{array}{ll}\min\limits_{f\in\mathcal{H}_\mathcal{K}, e_i, b\in\mathbb{R}} & \frac{\lambda}{2}\|f\|_\mathcal{K}^2 + \frac{1}{m}\sum_{i=1}^m e_i \\ \text{s.t.} & 0 \leq e_i \leq 1, \quad i = 1,\cdots,m, \\ & y_i(f(x_i)+b) \geq 1 - e_i, \quad \text{if } \omega_i^* = 0.\end{array} \tag{16}$$

Let $e^* = [e_1^*, \cdots, e_m^*]^T$ be the optimal slack variables in the above minimization problem. Then the triple $(\tilde{f}_{\mathbf{z},\lambda}^*, \tilde{b}_{\mathbf{z},\lambda}^*, e^*)$ is the optimal solution of minimization problem (16). Correspondingly, denote $(\tilde{f}_{\mathbf{z},\lambda}^1, \tilde{b}_{\mathbf{z},\lambda}^1, e^{*1})$ as the optimal solution of minimization problem (15)

with $e^{*1} = [e_1^{*1}, \cdots, e_m^{*1}]^T$. As the constraints in problem (16) is a subset of that in problem (15), we thus have

$$\frac{\lambda}{2}\|\tilde{f}^1_{\mathbf{z},\lambda}\|^2_{\mathcal{K}} + \frac{1}{m}\sum_{i=1}^m e_i^{*1} \leq \frac{\lambda}{2}\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}} + \frac{1}{m}\sum_{i=1}^m e_i^*.$$

To prove our claim, we just need to verify that $0 \leq e_i^{*1} \leq 1$ for $i = 1, \cdots, m$. For $\omega_i^* = 1$, it is easy to see that $e_i^{*1} = 0$. Next we prove the conclusion for the case $\omega_i^* = 0$. Define an index set as $I := \{i \in \{1, \cdots, m\} : \omega_i^* = 0 \text{ and } e_i^{*1} > 1\}$. If I is an non-empty set, we further define a binary vector ω' with $\omega'_i = 1$ for $i \in I$ and $\omega'_i = \omega_i^*$ otherwise. As $\omega_i = 1$ implies the corresponding optimal e_i should equal 0, we then define e'_i as $e'_i = 0$ if $\omega'_i = 1$ and $e'_i = e_i^{*1}$ otherwise. One can check that

$$\frac{\lambda}{2}\|\tilde{f}^1_{\mathbf{z},\lambda}\|^2_{\mathcal{K}} + \frac{1}{m}\sum_{i=1}^m (e'_i + \omega'_i) < \frac{\lambda}{2}\|\tilde{f}^1_{\mathbf{z},\lambda}\|^2_{\mathcal{K}} + \frac{1}{m}\sum_{i=1}^m (e_i^{*1} + \omega_i^*) \leq \frac{\lambda}{2}\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}} + \frac{1}{m}\sum_{i=1}^m (e_i^* + \omega_i^*).$$

We thus derive a contradiction to the assumption that $(\tilde{f}^*_{\mathbf{z},\lambda}, \tilde{b}^*_{\mathbf{z},\lambda}, e^*, \omega^*)$ is a global optimal solution for problem (3) and the conclusion follows.

Now we can prove our desired result based on the optimization problem (15). Let $I_0 = \{i : \omega_i^* = 0\}$ and $I_1 = \{i : \omega_i^* = 1\}$. Since the triple $(\tilde{f}^*_{\mathbf{z},\lambda}, \tilde{b}^*_{\mathbf{z},\lambda}, e^*)$ is the optimal solution of problem (15), from the KKT condition, there exist constants $\{\tilde{\alpha}_i^*\}_{i\in I_0}$, such that

$$\begin{aligned}
&\tilde{f}^*_{\mathbf{z},\lambda}(x) = \sum_{i\in I_0} \tilde{\alpha}_i^* y_i K(x_i, x) \text{ with } 0 \leq \tilde{\alpha}_i^* \leq \frac{1}{\lambda m},\\
&\sum_{i\in I_0} \tilde{\alpha}_i^* y_i = 0,\\
&1 - y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i) + \tilde{b}^*_{\mathbf{z},\lambda}) \leq 0, \quad \text{if } i \in I_0 \text{ and } \tilde{\alpha}_i^* = 0,\\
&0 \leq e_i^* = 1 - y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i) + \tilde{b}^*_{\mathbf{z},\lambda}) \leq 1, \quad \text{if } i \in I_0 \text{ and } \tilde{\alpha}_i^* \neq 0.
\end{aligned}$$

We also have $e_i^* = 0$, if $i \in I_1$. Moreover, by the same argument used in the proof about the equivalence of problems (15) and (16), one can find that when $i \in I_1$, we must have $1 - y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i) + \tilde{b}^*_{\mathbf{z},\lambda}) > 1$ or $1 - y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i) + \tilde{b}^*_{\mathbf{z},\lambda}) < 0$ due to the optimality of ω^*.

From the expression of $\tilde{f}^*_{\mathbf{z},\lambda}$, we can write $\tilde{f}^*_{\mathbf{z},\lambda}$ as $\sum_{i=1}^m \alpha_i^* y_i K(x_i, x)$ with $\alpha_i^* = \tilde{\alpha}_i^*$ if $i \in I_0$ and $\alpha_i^* = 0$ otherwise. Then $\tilde{f}^*_{\mathbf{z},\lambda} \in \mathcal{H}^+_{\mathcal{K},\mathbf{z}}$. Furthermore, the relation $\sum_{i\in I_0} \tilde{\alpha}_i^* y_i = 0$ implies $\sum_{i\in I_0} \tilde{\alpha}_i^* y_i \tilde{b}^*_{\mathbf{z},\lambda} = 0$. Then we have

$$\Omega(\tilde{f}^*_{\mathbf{z},\lambda}) = \sum_{i\in I_0} \tilde{\alpha}_i^* = \sum_{i\in I_0} \tilde{\alpha}_i^*(1 - y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i) + \tilde{b}^*_{\mathbf{z},\lambda})) + \sum_{i\in I_0} \tilde{\alpha}_i^* y_i \tilde{f}^*_{\mathbf{z},\lambda}(x_i).$$

Note that $\tilde{f}^*_{\mathbf{z},\lambda}(x) = \sum_{i\in I_0} \tilde{\alpha}_i^* y_i K(x_i, x)$. By the definition of $\|\cdot\|_{\mathcal{K}}$-norm, it follows that

$$\sum_{i\in I_0} \tilde{\alpha}_i^* y_i \tilde{f}^*_{\mathbf{z},\lambda}(x_i) = \sum_{i,j\in I_0} \tilde{\alpha}_i^* y_i \tilde{\alpha}_j^* y_j K(x_i, x_j) = \|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}}.$$

Additionally, based on our analysis, we also have

$$\sum_{i\in I_0}\tilde{\alpha}_i^*(1-y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i)+\tilde{b}^*_{\mathbf{z},\lambda}))=\sum_{i\in I_0}\tilde{\alpha}_i^* L_{\mathrm{ramp}}(y_i(\tilde{f}^*_{\mathbf{z},\lambda}(x_i)+\tilde{b}^*_{\mathbf{z},\lambda}))\le\lambda^{-1}\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda}).$$

Hence the bound for $\Omega(\tilde{f}^*_{\mathbf{z},\lambda})$ follows. ■

Now we are in the position to make an error decomposition for ramp-LPSVM.

Theorem 4 *For $0<\mu\le\lambda\le 1$, let $\eta=\frac{\mu}{\lambda}$. Recall that $f_{\mathbf{z},\mu}=f^*_{\mathbf{z},\mu}+b^*_{\mathbf{z},\mu}$ where $(f^*_{\mathbf{z},\mu},b^*_{\mathbf{z},\mu})$ is a global minimizer of ramp-LPSVM (6) and $f_\lambda=f^*_\lambda+b^*_\lambda$ with $(f^*_\lambda,b^*_\lambda)$ given by (12). Define the sample error $\mathcal{S}(m,\mu,\lambda)$ as below,*

$$\mathcal{S}(m,\mu,\lambda)=\left\{\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_{\mathbf{z},\mu})-\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_{\mathbf{z},\mu})\right\}+(1+\eta)\left\{\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_\lambda)-\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_\lambda)\right\}.$$

Then there holds

$$\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_{\mathbf{z},\mu})-\mathcal{R}_{\mathrm{ramp},\rho}(f_c)+\mu\Omega(f^*_{\mathbf{z},\mu})\le\eta\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_c)+\mathcal{S}(m,\mu,\lambda)+2\mathcal{A}(\lambda),\qquad(17)$$

where $\mathcal{A}(\lambda)$ is the approximation error given by (13).

Proof Recall that for any $\lambda>0$, $\tilde{f}_{\mathbf{z},\lambda}=\tilde{f}^*_{\mathbf{z},\lambda}+\tilde{b}^*_{\mathbf{z},\lambda}$ where $(\tilde{f}^*_{\mathbf{z},\lambda},\tilde{b}^*_{\mathbf{z},\lambda})$ is given by (11). Due to the definition of $f_{\mathbf{z},\mu}$ and the fact $\tilde{f}^*_{\mathbf{z},\lambda}\in\mathcal{H}^+_{\mathcal{K},\mathbf{z}}$, we have

$$\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_{\mathbf{z},\mu})+\mu\Omega(f^*_{\mathbf{z},\mu})\le\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})+\mu\Omega(\tilde{f}^*_{\mathbf{z},\lambda}).$$

Proposition 3 gives

$$\Omega(\tilde{f}^*_{\mathbf{z},\lambda})\le\lambda^{-1}\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})+\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}}.$$

Hence,

$$\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_{\mathbf{z},\mu})+\mu\Omega(f^*_{\mathbf{z},\mu})\le\left(1+\frac{\mu}{\lambda}\right)\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})+\mu\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}}.$$

This enables us to bound $\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_{\mathbf{z},\mu})+\mu\Omega(f^*_{\mathbf{z},\mu})$ as

$$\begin{aligned}\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_{\mathbf{z},\mu})+\mu\Omega(f^*_{\mathbf{z},\mu}) &\le \left\{\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_{\mathbf{z},\mu})-\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_{\mathbf{z},\mu})\right\}\\ &\quad+\left(1+\frac{\mu}{\lambda}\right)\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})+\mu\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}}.\end{aligned}$$

Next we use the definitions of $\tilde{f}_{\mathbf{z},\lambda}$ and f_λ to analyze the last two terms of the above bound:

$$\begin{aligned}&\left(1+\frac{\mu}{\lambda}\right)\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})+\mu\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}}\\ &\le\left(1+\frac{\mu}{\lambda}\right)\left(\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(\tilde{f}_{\mathbf{z},\lambda})+\lambda\|\tilde{f}^*_{\mathbf{z},\lambda}\|^2_{\mathcal{K}}\right)\\ &\le\left(1+\frac{\mu}{\lambda}\right)\left(\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_\lambda)+\lambda\|f^*_\lambda\|^2_{\mathcal{K}}\right)\\ &=\left(1+\frac{\mu}{\lambda}\right)\left(\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_\lambda)-\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_\lambda)+\mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_\lambda)+\lambda\|f^*_\lambda\|^2_{\mathcal{K}}\right).\end{aligned}$$

Combining the above estimates, we find that $\mathcal{R}_{L_{\rm ramp},\rho}(f_{\mathbf{z},\mu}) - \mathcal{R}_{{\rm ramp},\rho}(f_c) + \mu\Omega(f^*_{\mathbf{z},\mu})$ can be bounded by

$$\left\{\mathcal{R}_{L_{\rm ramp},\rho}(f_{\mathbf{z},\mu}) - \mathcal{R}_{L_{\rm ramp},\mathbf{z}}(f_{\mathbf{z},\mu})\right\} + \left(1+\frac{\mu}{\lambda}\right)\left\{\mathcal{R}_{L_{\rm ramp},\mathbf{z}}(f_\lambda) - \mathcal{R}_{L_{\rm ramp},\rho}(f_\lambda)\right\}$$
$$+ \left(1+\frac{\mu}{\lambda}\right)\left\{\mathcal{R}_{L_{\rm ramp},\rho}(f_\lambda) - \mathcal{R}_{L_{\rm ramp},\rho}(f_c) + \lambda\|f^*_\lambda\|^2_{\mathcal{K}}\right\} + \frac{\mu}{\lambda}\mathcal{R}_{L_{\rm ramp},\rho}(f_c).$$

Recalling the definition of f_λ, one has $\mathcal{A}(\lambda) = \mathcal{R}_{L_{\rm ramp},\rho}(f_\lambda) - \mathcal{R}_{L_{\rm ramp},\rho}(f_c) + \lambda\|f^*_\lambda\|^2_{\mathcal{K}}$. Hence the desired result follows. ■

With the help of Theorem 4, the generalization error is estimated by bounding $\mathcal{S}(m,\mu,\lambda)$ and $\mathcal{A}(\lambda)$ respectively. As the ramp loss is Lipschitz continuous, one can show that

$$\mathcal{R}_{{\rm ramp},\rho}(f) - \mathcal{R}_{{\rm ramp},\rho}(f_c) \le \|f - f_c\|_{L^1_{\rho_X}}.$$

Hence the approximation error $\mathcal{A}(\lambda)$ can be estimated by the approximation in a weighted L^1 space with the norm $\|f\|_{L^1_{\rho_X}} = \int_X |f(x)|d\rho_X$, as done in Smale and Zhou (2003). The following assumption is standard in the literature of learning theory (see, e.g., Cucker and Zhou, 2007; Steinwart and Christmann, 2008).

Assumption 1 *For any $0 < \beta \le 1$ and $c_\beta > 0$, the approximation error satisfies*

$$\mathcal{A}(\lambda) \le c_\beta \lambda^\beta, \qquad \forall \lambda > 0. \tag{18}$$

We also expect that the sample error $\mathcal{S}(m,\lambda,\mu)$ will tend to zero at a certain rate as the sample size tends to infinity. The asymptotical behaviors of $\mathcal{S}(m,\lambda,\mu)$ can be illustrated by the convergence of the empirical mean $\frac{1}{m}\sum_{i=1}^m \varsigma_i$ to its expectation $\mathbb{E}\varsigma_i$, where $\{\varsigma_i\}_{i=1}^m$ are independent random variables defined as

$$\varsigma_i = L_{\rm ramp}(y_i f(x_i)). \tag{19}$$

At the end of this section, we present our main theorem to illustrate the convergence behavior of ramp-LPSVM (6).

Theorem 5 *Suppose that Assumption 1 holds with $0 < \beta \le 1$. Take $\mu = m^{-\frac{\beta+1}{4\beta+2}}$ and $f_{\mathbf{z},\mu} = f^*_{\mathbf{z},\mu} + b^*_{\mathbf{z},\mu}$ with $(f^*_{\mathbf{z},\mu}, b^*_{\mathbf{z},\mu})$ being the global minimizer of ramp-LPSVM (6). Then for any $0 < \delta < 1$, with probability at least $1-\delta$, there holds*

$$\mathcal{R}_{L_{\rm mis},\rho}({\rm sgn}(f_{\mathbf{z},\mu})) - \mathcal{R}_{L_{\rm mis},\rho}(f_c) \le \tilde{c}\left(\log\frac{4}{\delta}\right)^{1/2} m^{-\frac{\beta}{4\beta+2}}, \tag{20}$$

where $\tilde{c}$ is a constant independent δ or m.

This theorem will be proved in Appendix by concentration techniques developed by Bartlett and Mendelson (2003). Based on the decomposition formula (17) established for ramp-LPSVM, one can also derive sharp convergence results under the framework applied by Wu and Zhou (2005). Here we use ramp-SVM (11) to conduct an error decomposition

for ramp-LPSVM (6), so the derived convergence rates of the latter are essentially no worse than those of ramp-SVM. Actually, also from our discussion in this section, ramp-SVM and C-SVM should have almost the same error bounds. One thus can expect that ramp-LPSVM enjoys similar asymptotic behaviors as C-SVM. It also should be pointed that, throughout our analysis, the global optimality plays an important role. Therefore, to guarantee the performance of ramp-LPSVM, a global search strategy is necessary.

3. Problem-solving Algorithms

In the previous section, we discussed theoretical properties for ramp-LPSVM. Its robustness and sparsity can be expected, if a good solution of ramp-LPSVM (6) can be obtained. However, (6) is non-convex. Therefore, in this paper, we propose a downhill method for local minimization and a heuristic for escaping a local minimum. Difference of convex function (DC) programming proposed by An et al. (1996) and An and Tao (2005) has been applied for ramp loss minimization problems (see Wu and Liu, 2007; Wang et al., 2010). By Yuille and Rangarajan (2003), Collobert et al. (2006b), Zhao and Sun (2008), this type of methods is also called a concave-convex procedure. For the proposed ramp-LPSVM, the DC technique is applicable as well.

Let $\alpha = [\alpha_1, \cdots, \alpha_m]^T \in \mathbb{R}^m$. Based on the identity (2), ramp-LPSVM (6) can be written as follows,

$$\begin{aligned} \min_{\alpha \succeq 0, b} \quad & \mu \sum_{i=1}^{m} \alpha_i + \frac{1}{m} \sum_{i=1}^{m} \max \left\{ 1 - y_i \left(\sum_{j=1}^{m} \alpha_j y_j \mathcal{K}(x_i, x_j) + b \right), 0 \right\} \\ & - \frac{1}{m} \sum_{i=1}^{m} \max \left\{ -y_i \left(\sum_{j=1}^{m} \alpha_j y_j \mathcal{K}(x_i, x_j) + b \right), 0 \right\}. \end{aligned} \tag{21}$$

We let $\zeta = [\alpha^T, b]^T$ stand for the optimization variable and $D(\zeta)$ for the feasible set of (21). Denote the convex part (the first line of) as $g(\zeta)$, and the concave part (the second line of (21)) as $h(\zeta)$. After that, (21) can be written as $\min_{\zeta \in D(\zeta)} g(\zeta) - h(\zeta)$. Then DC programming developed by Horst and Thoai (1999) and An and Tao (2005) is applicable. We give the following algorithm for local minimization for ramp-LPSVM.

Algorithm 1: DC programming for ramp-LPSVM from $\hat{\alpha}, \hat{b}$

- Set $\delta > 0$, $k := 0$ and $\zeta_0 := [\hat{\alpha}^T, \hat{b}]^T$;

repeat

- Select $\eta_k \in \partial h(\zeta_k)$;
- $\zeta_{k+1} := \arg \min_{\zeta \in D(\zeta)} g(\zeta) - \big(h(\zeta_k) + (\zeta - \zeta_k)^T \eta_k\big)$;
- Set $k := k + 1$;

until $\|\zeta_k - \zeta_{k-1}\| < \delta$;

- Algorithm ends and returns ζ_k.

Since $g(\zeta)$ is convex and piecewise linear, Algorithm 1 involves only LP, which can be effectively solved. One noticeable point is that $h(\zeta)$ is not differentiable at some points.

The non-differentiability of $h(\zeta)$ comes from $\max\{u, 0\}$, of which the sub-gradient at $u = 0$ is in the interval $[0, 1]$:

$$\frac{\partial \max\{u, 0\}}{\partial u}\Big|_{u=0} \in [0, 1].$$

In our algorithm, we choose 0.5 as the value of the above sub-gradient and then $\eta_k \in \partial h(\zeta_k)$ is uniquely defined. The local optimality condition for DC problems has been investigated by An and Tao (2005) and references therein. For a differentiable function, one can use the gradient information to check whether the solution is locally optimal. However, ramp-LPSVM is non-smooth and a sub-gradient technique should be considered. The local minimizer of a non-smooth objective function should meet the local optimality condition for all vectors in its sub-gradient set. In Algorithm 1, we only consider one value of the sub-gradient, thus, the result of the above process is not necessarily a local minimum. The rigorous local optimality condition and the related algorithm can be found in Huang et al. (2012b). However, because of the effectiveness of DC programming, we suggest Algorithm 1 for ramp-LPSVM in this paper.

As a local search algorithm, DC programming can effectively decrease the objective value of (21). The main difficulty of solving (21) is that it is non-convex and hence we may be trapped in a local optimum. To escape from a local optimum, we introduce slack variable $c = [c_1, \cdots, c_m]^T$ and transform (21) into the following concave minimization problem,

$$\begin{aligned} \min_{\alpha,b,c} \quad & \mu \sum_{i=1}^{m} \alpha_i + \frac{1}{m} \sum_{i=1}^{m} c_i - \frac{1}{m} \sum_{i=1}^{m} \max \left\{ -y_i \left(\sum_{j=1}^{m} \alpha_j y_j \mathcal{K}(x_i, x_j) + b \right), 0 \right\} \\ \text{s.t.} \quad & c_i \geq 1 - y_i \left(\sum\nolimits_{j=1}^{m} \alpha_j y_j \mathcal{K}(x_i, x_j) + b \right), \quad i = 1, 2, \ldots, m, \\ & c_i \geq 0, \quad i = 1, 2, \ldots, m, \\ & \alpha_i \geq 0, \quad i = 1, 2, \ldots, m. \end{aligned} \tag{22}$$

This is a concave minimization problem constrained in a polyhedron, which is called a polyhedral concave problem by Horst and Hoang (1996). Generally, among non-convex problems, a polyhedral concave problem is relatively easy to deal with. Various techniques, such as γ-extension, vertex enumeration, partition algorithm, concavity cutting, have been discussed insightfully in Horst and Hoang (1996) and successfully applied (see, e.g., Porembski, 2004; Mangasarian, 2007; Shu and Karimi, 2009). Moreover, the objective function of (22) is piecewise linear, which makes the hill detouring method proposed by Huang et al. (2012a) applicable. In the following, we first introduce the basic idea of the hill detouring method and then establish a global search algorithm for ramp-LPSVM.

For notational convenience, we use $\xi = [\alpha^T, b, c^T]^T$ to denote the optimization variable of (22). The objective function is continuous piecewise linear and is denoted as $p(\xi)$. The feasible set, which is a polyhedron, can be written as $A\xi \leq q$. Then (22) is compactly represented as the following polyhedral concave problem, of which the objective function is piecewise linear:

$$\min\nolimits_{\xi} \quad p(\xi), \quad \text{s.t.} \quad A\xi \leq q. \tag{23}$$

Assume that we are trapped in a local optimum $\tilde{\xi}$ with value $\tilde{p} = p(\tilde{\xi})$ and we are trying to escape from it. We observe that (in a non-degenerated case): i) the local optimum $\tilde{\xi}$ is a

vertex of the feasible set; ii) any level set $\{\xi : p(\xi) = u\}, \forall u$ is the boundary of a polyhedron. The first property can be derived from the concavity of the objective function. The second property comes from the piecewise linearity of $p(\xi)$. These properties imply a new method searching on the level set to find another feasible solution $\hat{\xi}$ with the same objective value $p(\hat{\xi}) = \tilde{p}$. If such $\hat{\xi}$ is found, we escape from $\tilde{\xi}$ and a downhill method can be used to find a new local optimum. Otherwise, if such $\hat{\xi}$ does not exist, one can conclude that $\tilde{\xi}$ is the optimal solution. Searching on the level set of $p(\xi) = \tilde{p}$ will not decrease neither increase the objective value and it is hence called hill detouring. In practice, in order to avoid to find $\tilde{\xi}$ again, we search on $\{p(\xi) = \tilde{p} - \varepsilon\}$ with a small positive ε for computational convenience. If $\{p(\xi) = \tilde{p} - \varepsilon\} = \emptyset$, we know that $\tilde{\xi}$ is ε-optimal. The performance of hill detouring is not sensitive to the ε value, when ε is small (but large enough to distinguish $\tilde{p} - \epsilon$ and $\tilde{p}$). In this paper, we set $\varepsilon = 10^{-6}$.

Hill detouring, which is to solve the feasibility problem

$$\text{find} \quad \xi, \quad \text{s.t.} \quad p(\xi) = \tilde{p} - \varepsilon, \ A\xi \leq q, \tag{24}$$

is a natural idea for global optimization but it is hard to implement for a regular concave minimization functions. The main difficulty is the nonlinear equation $p(\xi) = \tilde{p} - \varepsilon$. In ramp-LPSVM, the objective function of (22) is continuous and piecewise linear, thus, $p(\xi) = \tilde{p} - \varepsilon$ can be transformed into (finite) linear equations. That means (24) can be written as a series of LP feasibility problems, which makes line search on $\{\xi : p(\xi) = \tilde{p} - \varepsilon\}$ possible.

To investigate the property of (23) and the corresponding hill detouring technique, we consider a 2-dimensional problem. In this intuitive example, the objective function is $p(\xi) = a_0^T \xi + b_0 - \sum_{i=1}^{6} \max\{0, a_i^T \xi + b_i\}$, where

$$a_0 = \begin{bmatrix} 0.05 \\ -0.1 \end{bmatrix} \quad a_1 = \begin{bmatrix} -1 \\ -0.4 \end{bmatrix} \quad a_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad a_3 = \begin{bmatrix} 0.5 \\ 0.1 \end{bmatrix} \quad a_4 = \begin{bmatrix} -0.9 \\ 0.4 \end{bmatrix} \quad a_5 = \begin{bmatrix} -0.6 \\ -1 \end{bmatrix} \quad a_6 = \begin{bmatrix} 0.9 \\ 0.9 \end{bmatrix}$$
$$b_0 = -0.2 \quad b_1 = 0.8 \quad b_2 = -0.2 \quad b_3 = -0.5 \quad b_4 = 0.2 \quad b_5 = 1 \quad b_6 = 0.8.$$

The feasible domain is an octagon, of which the vertices are $[2, 1]^T, [1, 2]^T, \ldots, [1, -2]^T$. The plots of $p(\xi)$ and the feasible set are shown in Figure 2, where $\tilde{\xi} = [2, 1]^T$ is a local optimum and the global optimum is $\xi^\star = [-2, -1]^T$.

Now we try to escape from $\tilde{\xi}$ by hill detouring. In other words, we search on the level set $\{\xi : p(\xi) = \tilde{p} - \varepsilon\}$ to find a feasible solution. The level set is displayed by the green dashed line in Figure 3. According to the property that $\tilde{\xi}$ is a vertex of the feasible domain, we can first search along the corresponding active edges, which are shown by the black solid lines, to find the γ-extensions. The definition of γ-extension was given by Horst and Hoang (1996) and is reviewed below.

Definition 6 *Suppose f is a concave function, ξ is a given point, γ is a scalar with $\gamma \leq f(\xi)$, and θ_0 is a positive number large enough. Let $d \neq 0$ be a direction and $\theta = \min\{\theta_0, \sup\{t : f(\xi + td) \geq \gamma\}\}$, then $\xi + \theta d$ is called the γ-extension of $f(\xi)$ from ξ along d.*

Set $\gamma = \tilde{p} - \varepsilon$. γ-extensions from $\tilde{\xi}$ can be easily found by bisection according to the concavity of $p(\xi)$. For any direction d, we set $t_1 = 0$ and t_2 as a large enough positive number. If $p(\tilde{\xi} + t_2 d) > \gamma$, there is no γ-extension along this direction. Otherwise, after the following bisection scheme, $\frac{1}{2}(t_1 + t_2)$ is the γ-extension from $\tilde{\xi}$ along d,

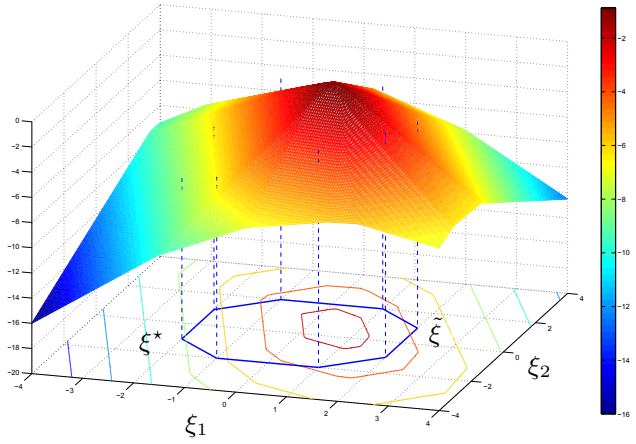

Figure 2: Plots of the objective function $p(\xi)$ and the feasible domain $A\xi \leq q$, of which the boundary is shown by the blue solid line. $\tilde{\xi} = [2, 1]$ is a local optimum and $\tilde{p} = p(\tilde{\xi}) = -4.5$; $\xi^\star = [-2, -1]$ with $p(\xi^\star) = -8.2$ is the global optimum.

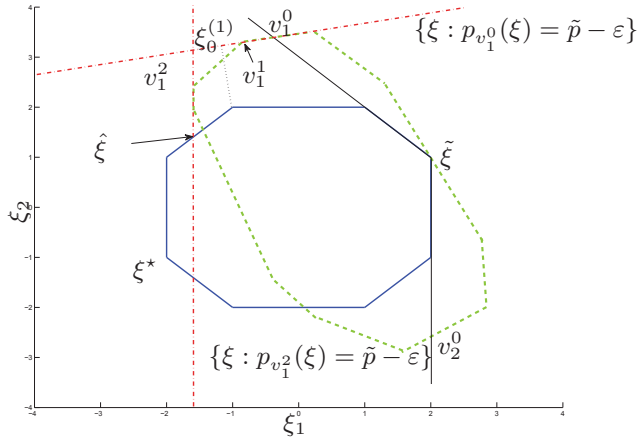

Figure 3: Hill detouring method. From a local optimum $\tilde{\xi}$, we can find v_1^0, which is the γ-extension along the active edge. Searching in the hyperplane of the level set, we arrive at v_1^1, v_1^2, and $\hat{\xi}$, successively. $\hat{\xi}$ is feasible and has a less objective value than $p(\tilde{\xi})$, then we successfully escape from the local optimum $\tilde{\xi}$.

While $t_2 - t_1 > 10^{-6}$

If $f(\tilde{\xi} + \frac{1}{2}(t_1+t_2)d)) > \gamma$, set $t_1 = \frac{1}{2}(t_1+t_2)$; **Else** set $t_2 = \frac{1}{2}(t_1+t_2)$.

For the concerned example, along the edges of the feasible set, which are active at $\tilde{\xi}$, we find the γ-extensions, denoted by v_1^0 and v_2^0. If the convex hull of v_1^0, v_2^0 and $\tilde{\xi}$ covers the feasible set, $\tilde{\xi}$ is ε-optimal for (23). Otherwise, these extensions provide good initial points for hill detouring.

The objective function $p(x)$ is piecewise linear and there exist a finite number of subregions, in each of which, $p(\xi)$ becomes a linear function. Therefore, for any given ξ_0, we can find a subregion, denoted by D_{ξ_0}, such that $\xi_0 \in D_{\xi_0}$ and there is a corresponding linear function, denoted by $p_{\xi_0}(\xi)$, satisfying: $p(\xi) = p_{\xi_0}(\xi), \forall \xi \in D_{\xi_0}$. Constrained in the region related to ξ_0, the feasibility problem (24) becomes

$$\begin{aligned} \text{find} \quad & \xi \\ \text{s.t.} \quad & p_{\xi_0}(\xi) = \tilde{p} - \varepsilon, \quad \xi \in D_{\xi_0} \\ & A\xi \le q. \end{aligned} \tag{25}$$

Since $p(\xi)$ is concave and $p_{\xi_0}(\xi)$ is essentially the first order Taylor expansion of $p(\xi)$, we know that $p(\xi) \le p_{\xi_0}(\xi), \forall \xi_0, \xi$, where the equality holds when $\xi \in D_{\xi_0}$. For a solution ξ' satisfying $p_{\xi_0}(\xi') = \tilde{p} - \varepsilon$ but outside D_{ξ_0}, we have $p(\xi') < \tilde{p} - \varepsilon$. If ξ' is feasible ($A\xi' \le q$), then a better solution is found. Therefore, in hill detouring method, we ignore the constraint $\xi \in D_{\xi_0}$ in (25) and consider the following optimization problem,

$$\begin{aligned} \min_{\xi^{(1)}, \xi^{(2)}} \quad & \|\xi^{(1)} - \xi^{(2)}\|_\infty \\ \text{s.t.} \quad & p_{\xi_0}(\xi^{(1)}) = \tilde{p} - \varepsilon \\ & A\xi^{(2)} \le q, \end{aligned} \tag{26}$$

for which $\xi^{(1)} = \xi_0, \xi^{(2)} = \tilde{\xi}$ provides a feasible solution. Notice that after introducing a slack variable $s \in \mathbb{R}$, minimizing $\|\xi^{(1)} - \xi^{(2)}\|_\infty$ is equivalently to minimize s with the constraint that each component of $\xi^{(1)} - \xi^{(2)}$ is between $-s$ and s. Then (26) is essentially an LP problem. Starting from v_1^0, we set $\xi_0 = v_1^0$ and solve (26), of which the solution is denoted by $\xi_0^{(1)}, \xi_0^{(2)}$. As displayed in Figure 3, $\xi_0^{(1)}$ is the point which is closest to the feasible domain among all the points in hyperplane $p_{v_1^0}(\xi) = \tilde{p} - \varepsilon$. Heuristically, we search on the level set towards $\xi_0^{(1)}$: going along the direction $d_0 = \xi_0^{(1)} - \xi_0$ and finding point v_1^1, where $p(\xi)$ becomes another linear function. v_1^1 is also a vertex of the level set $\{\xi : p(\xi) = \tilde{p} - \varepsilon\}$. Then we construct a new linear function $p_{v_1^1}(\xi)$, which is different to $p_{v_1^0}(\xi)$. Repeating the above process, we can get v_1^2. After that, solving (26) for $\xi_0 = v_1^2$ leads to $\hat{\xi}$, which is feasible and has a objective value $\tilde{p} - \varepsilon$, then we successfully escape from $\tilde{\xi}$ by hill detouring.

We have shown the basic idea of the hill detouring method by one 2-dimensional problem. For ramp-LPSVM, the hill detouring method for (22) is similar to the above process. Specifically, the local linear function for a given $\xi_0 = [\alpha_0^T, b_0, c_0^T]^T$ is below,

$$p_{\xi_0}(\xi) = \mu \sum_{i=1}^{m} \alpha_i + \frac{1}{m} \sum_{i=1}^{m} c_i + \frac{1}{m} \sum_{i \in \mathcal{M}_{\xi_0}} y_i \left(\sum_{j=1}^{m} \alpha_j y_j \mathcal{K}(x_i, x_j) + b \right), \tag{27}$$

where $\mathcal{M}_{\xi_0}$ is a union of $\mathcal{M}^+_{\xi_0}$ and any subset of $\mathcal{M}^0_{\xi_0}$ and the related sets are defined below,

$$\begin{aligned}\mathcal{M}^+_\xi &= \left\{i : -y_i\left(\sum_{j=1}^m \alpha_j y_j \mathcal{K}(x_i,x_j)+b\right) > 0\right\},\\ \mathcal{M}^0_\xi &= \left\{i : -y_i\left(\sum_{j=1}^m \alpha_j y_j \mathcal{K}(x_i,x_j)+b\right) = 0\right\}.\end{aligned}$$

The above choice means $\mathcal{M}^+_{\xi_0} \subseteq \mathcal{M}_{\xi_0} \subseteq \mathcal{M}^+_{\xi_0} \bigcup \mathcal{M}^0_{\xi_0}$. For a random ξ, $\mathcal{M}^0_\xi$ is usually empty. For a point like v_1^1 in Figure 3, which is a vertex of the level set, $\mathcal{M}^0_{v_1^1} \neq \emptyset$. In this case, there are multiple choices for p_{ξ_0} and we select $\mathcal{M}_{\xi_0}$ which has not been considered. Summarizing the discussions, we give the following algorithm for ramp-LPSVM (6).

Algorithm 2: Global Search for ramp-LPSVM

initialize

- Set δ (the threshold of convergence for DC programming), ε (the difference value in hill detouring), K_{step} (the maximal number of hill detouring steps)
- Give an initial feasible solution $\hat{\alpha}, \hat{b}$;

repeat

- Use Algorithm 1 from $\hat{\alpha}, \hat{b}$ to obtain locally optimal solution $\tilde{\alpha}, \tilde{b}$;
- Compute $\tilde{c}_i := \max\left\{-y_i\left(\sum_{j=1}^m \tilde{\alpha}_j y_j \mathcal{K}(x_i,x_j)+\tilde{b}\right), 0\right\}$;
- Set $\tilde{\xi} := [\tilde{\alpha}^T, \tilde{b}, \tilde{c}^T]^T$, $\gamma := p(\tilde{\xi}) - \varepsilon$, where $p(\xi)$ is the object of (22), and compute the γ-extensions for edges active at $\tilde{\xi}$. We denote the γ-extensions as $v_1, v_2, \ldots$ and the distance of v_i to the feasible set of (22) as dist_i;
- Let $k := 0$ and $\mathcal{S}_\mathcal{M} := \emptyset$;
- **repeat**
  - Let $k := k+1$, select $i_0 := \arg\min_i \text{dist}_i$, and set $\xi_0 := v_{i_0}$;
  - Select $\mathcal{M}_{\xi_0}$ according to $\mathcal{M}^+_{\xi_0}, \mathcal{M}^0_{\xi_0}$ such that $\mathcal{M}_{\xi_0} \notin \mathcal{S}_\mathcal{M}$;
  - Set $\mathcal{S}_\mathcal{M} := \mathcal{S}_\mathcal{M} \bigcup \{\mathcal{M}_{\xi_0}\}$;
  - Construct $p_{\xi_0}(\xi)$ and solve LP (26), of which the solution is $\xi_0^{(1)}, \xi_0^{(2)}$;
  - **if** $\xi_0^{(1)} = \xi_0^{(2)}$ **then**
    - Set $\hat{\alpha}, \hat{b}$ according to $\xi_0^{(1)}$ and terminate the inner loop;
  - **else**
    - Let $d := \xi_0^{(1)} - \xi_0$ and find $\theta := \max\{\theta : p(\xi_0 + \theta d) = p_{\xi_0}(\xi_0 + \theta d)\}$;
    - Set $v_{i_0} := \xi_0 + \theta d$ and update dist_{i_0};
  - **end**
- **until** $k \geq K_{\text{step}}$;

until $\tilde{\alpha} = \hat{\alpha}, \tilde{b} = \hat{b}$;

- Algorithm ends and returns $\tilde{\alpha}, \tilde{b}$.

4. Numerical Experiments

In the numerical experiments, we evaluate the performance of ramp-LPSVM (6) and its problem-solving algorithms. We first report the optimization performance and then discuss

the robustness and the sparsity compared with C-SVM, LPSVM (5), and ramp-SVM (11). C-SVM and LPSVM are convex problems, which are solved by the Matlab optimization toolbox. For ramp-SVM, we apply the algorithm proposed by Collobert et al. (2006a). The data are downloaded from the UCI Machine Learning Repository given by Frank and Asuncion (2010). In data sets "Spect", "Monk1", "Monk2", and "Monk3", the training and the testing sets are provided. For the others, we randomly partition the data into two parts: half data are used for training and the remaining data are for testing. In this paper, we focus on outliers and hence we contaminate the training data set by randomly selecting some instances in class -1 and changing their labels. Since there are random factors in sampling and adding outliers, we repeat the above process 10 times for each data set and report the average accuracy on the testing data. In our experiments, we apply a Gaussian kernel $\mathcal{K}(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2/\sigma^2\right)$. The training data are normalized to $[0,1]^n$ and then the regularization coefficient μ and the kernel parameter σ are tuned by 10-fold cross-validation for each method. In the tuning phase, grid search using logarithmic scale is applied. The range of possible μ value is $[10^{-2}, 10^3]$ and the range of σ value is between 10^{-3} and 10^2. For ramp-LPSVM, since the global search needs more computation time, the parameters tuning by cross-validation is conducted based on Algorithm 1. The experiments are done in Matlab R2011a in Core 2-2.83 GHz, 2.96G RAM.

Intuitively, ramp-LPSVM can provide a sparse and robust result, if a good solution for (6) can be obtained. Hence, we first consider the optimization performance of the proposed algorithms. To evaluate them, we set $\mu = 1/10, \sigma = 1$ and use the four data sets for which the training data are provided. The result of ramp-LPSVM is sparse, we hence use $\hat{\alpha} = 0$, which is optimal when μ is large sufficiently, as the initial solution. When $\hat{\alpha} = 0$, simply calculating shows that $\hat{b} = 1$ is optimal to (6) if there are more training data in class $+1$ than in class -1 ($\#\{i : y_i = 1\} \geq \#\{i : y_i = -1\}$). Otherwise, we set $\hat{b} = -1$. From $\hat{\alpha}, \hat{b}$, we apply Algorithm 2 to minimize (6). Basically, Algorithm 2 in turn applies DC programming for local minimization and hill detouring for escaping local optima. In Table 2, we report the objective values of the obtained local optima and the corresponding computation time. The superscript indicates the sequence and f^1 is the result of Algorithm 1.

Data		f^1	f^2	f^3	f^4	f^5	f^6	GA
Spect	objective value	36.59	9.36	7.38	6.41	5.43	5.40	8.78
	time (s)	0.298	2.64	2.89	5.46	4.63	19.84	39.6
Monk1	objective value	9.94	8.96	7.11	—	—	—	10.10
	time (s)	4.04	8.14	26.8	—	—	—	66.34
Monk2	objective value	9.17	8.24	7.31	5.48	—	—	12.66
	time (s)	12.3	20.9	43.1	43.5	—	—	108.1
Monk3	objective value	4.92	4.02	—	—	—	—	11.38
	time (s)	3.93	32.7	—	—	—	—	69.21

Table 2: Global Search Performance of Algorithm 2 ($\delta = 10^{-6}, \varepsilon = 10^{-6}, K_{\text{step}} = 50$)

From the reported results, one can see the effectiveness of hill detouring for escaping from local optima. Another observation is that with the increasing quality of the local optimum, the hill detouring needs more time for escaping. When the initial point is not good, the

computation time for hill detouring is also small, which means that the performance of Algorithm 2 is not sensitive to the initial solution. To evaluate the global search capability, we also use the Genetic Algorithm (GA) toolbox developed by Chipperfield et al. (1994). The result of GA is random and we run GA algorithm repeatedly in the similar computing time of Algorithm 2. Then we select the best one and report it in Table 2. The comparison illustrates the global search capability of Algorithm 2. The basic elements of Algorithm 1 and Algorithm 2 are both to iteratively solve LPs. For large-scale problems, some fast methods for LP, especially the techniques designed for LPSVM by Bradley and Mangasarian (2000), Fung and Mangasarian (2004), and Mangasarian (2006), are applicable to speed up the solving procedure, which can be potential future work for ramp-LPSVM.

In the experiments above, the proposed algorithms show good minimization capability for ramp-LPSVM (6). Then one can expect good performance of the proposed model and algorithms, according to the robustness, sparsity, and other statistical properties discussed in Section 3. For each training set, we randomly select some data from class -1 and change their labels to be $+1$. The ratio of the outliers, denoted by r, is set to be $r = 0.0, 0.05, 0.10$. Based on the contaminated training set, we use C-SVM, LPSVM (5), ramp-SVM (11), and ramp-LPSVM (6) (solved by Algorithm 1 and Algorithm 2, respectively) to train the classifier and calculate the classification accuracy on the testing data. The above process is repeated 10 times. The average testing accuracy and the average number of support vectors (the corresponding $|\alpha_i|$ is larger than 10^{-6}) are reported in Table 3, where the data dimension n and the size of training data m are reported as well. The best results in the view of classification accuracy are underlined and the sparsest results are given in bold.

From Table 3, we observe that when there are no outliers, C-SVM performs well and LPSVM also provides good classifiers. The number of support vectors of LPSVM is always smaller than that of C-SVM, which relates to the property of ℓ_1 minimization. With an increasing number of outliers, the accuracy of C-SVM and LPSVM decreases. In contrast, the results of ramp-SVM and ramp-LPSVM are more stable, showing the robustness of the ramp loss. The ramp loss also brings some sparsity, since when $y_i f(x_i) \geq 0$, the ramp loss gives a constant penalty, which corresponds to a zero dual variable. The proposed ramp-LPSVM consists of the ℓ_1-penalty and the ramp loss, both of which can enhance the sparsity. Hence, the sparsity of the result of ramp-LPSVM is significant. Comparing the two algorithms for ramp-LPSVM, we find that Algorithm 2, which pursues a global solution, results in a more robust classifier. But the computation time of Algorithm 2 is significantly larger, as illustrated in Table 2. Generally, if there are heavy outliers and plenty allowable computation time, it is worth considering Algorithm 2 to find a good classifier. Otherwise, solving ramp-LPSVM by Algorithm 1 is a good choice.

5. Conclusion

In this paper, we proposed a robust classification method, called ramp-LPSVM. It consists of the ℓ_1-penalty and the ramp loss, which correspond to sparsity and robustness, respectively. The consistency and error bound for ramp-LPSVM have been discussed. Ramp-LPSVM trains a classifier by minimizing the ramp loss together with the ℓ_1-penalty, both of which are piecewise linear. According to the piecewise linearity, a local optimization method using DC programming and a global search strategy using hill detouring technique have been

Data	n	m	r	C-SVM		LPSVM		ramp-SVM		ramp-LPSVM (Algorithm 1)		ramp-LPSVM (Algorithm 2)	
Spect	21	80	0.00	86.03%	#80	88.77%	**#22**	88.77%	#75	88.77%	**#22**	88.77%	**#22**
	21	80	0.05	83.96%	#80	86.90%	**#18**	87.70%	#76	88.77%	#21	88.77%	#21
	21	80	0.10	84.49%	#79	84.33%	#20	85.56%	#74	87.71%	#18	88.37%	**#18**
Monk1	6	124	0.00	85.70%	#70	84.68%	#44	83.25%	#65	83.09%	#39	84.40%	**#38**
	6	124	0.05	82.70%	#72	80.61%	#47	81.17%	#61	81.92%	**#36**	82.07%	#37
	6	124	0.10	76.77%	#73	73.52%	#38	78.66%	#57	79.20%	**#31**	79.91%	#33
Monk2	6	169	0.00	83.80%	#96	81.05%	#62	83.80%	#86	82.62%	#57	82.86%	**#53**
	6	169	0.05	77.78%	#95	79.05%	#59	77.78%	#85	80.05%	**#54**	80.24%	#61
	6	169	0.10	71.76%	#96	75.88%	#58	74.68%	#75	78.53%	**#52**	79.84%	**#52**
Monk3	6	122	0.00	90.15%	#55	91.76%	#34	91.32%	#53	88.73%	**#29**	89.34%	#30
	6	122	0.05	87.13%	#61	88.47%	#33	88.68%	#47	87.42%	**#31**	86.80%	**#31**
	6	122	0.10	81.13%	#69	83.49%	#34	85.00%	#50	84.35%	#32	84.94%	**#30**
Breast	10	350	0.00	96.68%	#87	95.14%	#28	96.78%	#73	96.25%	#18	96.45%	**#17**
	10	350	0.05	95.63%	#90	93.41%	#25	95.90%	#71	96.20%	**#16**	96.07%	**#16**
	10	350	0.10	90.43%	#84	84.66%	#22	91.59%	#79	93.54%	**#16**	95.77%	#18
Pima	8	385	0.00	76.04%	#233	72.53%	#47	75.98%	#67	75.59%	#41	74.22%	**#39**
	8	385	0.05	75.78%	#230	74.31%	**#33**	75.29%	#68	74.56%	#40	74.40%	#42
	8	385	0.10	74.01%	#228	74.28%	**#31**	73.26%	#69	74.35%	#37	74.67%	#37
Trans.	4	375	0.00	76.33%	#199	75.86%	#22	77.01%	#32	77.11%	**#5**	77.11%	**#6**
	4	375	0.05	74.62%	#285	74.97%	#19	76.20%	#31	76.28%	**#6**	76.69%	#7
	4	375	0.10	73.06%	#274	72.90%	#12	76.73%	#30	75.28%	**#8**	76.28%	**#8**
Haber.	3	154	0.00	74.77%	#86	73.69%	#10	74.31%	#57	75.05%	**#5**	74.92%	**#5**
	3	154	0.05	74.38%	#81	73.25%	#9	73.26%	#68	73.79%	**#8**	73.97%	**#8**
	3	154	0.10	71.66%	#75	72.54%	**#11**	73.56%	#57	73.79%	**#11**	73.86%	**#11**
Ionos.	33	176	0.00	93.81%	#93	92.41%	**#29**	93.64%	#92	93.10%	#32	93.01%	#34
	33	176	0.05	92.22%	#97	89.26%	#32	92.89%	#95	92.33%	#32	93.03%	**#31**
	33	176	0.10	90.13%	#98	89.41%	#32	92.63%	#87	92.22%	**#27**	92.94%	#28

Table 3: Classification Accuracy on Testing Data and Number of Support Vectors

proposed. The proposed algorithms have good optimization capability and ramp-LPSVM has shown robustness and sparsity in numerical experiments.

Acknowledgments

The authors are grateful to the anonymous reviewers for insightful comments.
This work was supported in part by the scholarship of the Flemish Government; Research Council KUL: GOA/11/05 Ambiorics, GOA/10/09 MaNet, CoE EF/05/006 Optimization in Engineering (OPTEC), IOF-SCORES4CHEM, several PhD/postdoc & fellow grants; Flemish Government: FWO: PhD/postdoc grants, projects: G0226.06 (cooperative systems and optimization), G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) research communities (WOG: ICCoS, ANMMM, MLDM); G.0377.09 (Mechatronics MPC), G.0377.12 (Structured models), IWT: PhD Grants, Eureka-Flite+, SBO LeCoPro, SBO Climaqs, SBO POM, O&O-Dsquare; Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011); IBBT; EU: ERNSI; ERC AdG A-DATADRIVE-B, FP7-HD-MPC (INFSO-ICT-223854), COST intelliCIS, FP7-EMBOCON (ICT-248940); Contract Research: AMINAL; Other: Helmholtz: viCERP, ACCM, Bauknecht, Hoerbiger. L. Shi is also supported by the National Natural Science Foundation of China (No. 11201079) and the Fundamental Research Funds for the Central Universities of China (No. 20520133238, No. 20520131169). Johan Suykens is a professor at KU Leuven, Belgium.

Appendix A.

In this appendix, we prove Theorem 5 in Section 2. First, we bound the offset by the following lemma.

Lemma 7 *For any $\mu > 0$, $m \in \mathbb{N}$, and $\mathbf{z} = \{x_i, y_i\}_{i=1}^m$, we can find a solution $(f^*_{\mathbf{z},\mu}, b^*_{\mathbf{z},\mu})$ of equation (6) satisfying $\min_{1\le i\le m} |f_{\mathbf{z},\mu}(x_i)| \le 1$, where $f_{\mathbf{z},\mu} = f^*_{\mathbf{z},\mu} + b^*_{\mathbf{z},\mu}$. Hence, $|b^*_{\mathbf{z},\mu}| \le 1 + \|f^*_{\mathbf{z},\mu}\|_\infty$.*

Proof Suppose a minimizer $f_{\mathbf{z},\mu} = f^*_{\mathbf{z},\mu} + b^*_{\mathbf{z},\mu}$ of (6) satisfies

$$r := \min_{1\le i\le m} |f_{\mathbf{z},\mu}(x_i)| = |f_{\mathbf{z},\mu}(x_{i_0})| > 1.$$

Then for each i, either $y_i f_{\mathbf{z},\mu}(x_i) \ge r > 1$ or $y_i f_{\mathbf{z},\mu}(x_i) \le -r < -1$. We consider a function $f^d_{\mathbf{z},\mu} := f_{\mathbf{z},\mu} - d$ with $d = (r-1)\mathrm{sgn}(f_{\mathbf{z},\mu}(x_{i_0}))$. Then $f^d_{\mathbf{z},\mu}$ satisfies $|f^d_{\mathbf{z},\mu}(x_{i_0})| = 1$ and $|f^d_{\mathbf{z},\mu}(x_{i_0})| \ge 1$. When $y_i f_{\mathbf{z},\mu}(x_i) > 1$, one can check that $y_i f^d_{\mathbf{z},\mu}(x_i) \ge 1$. Similarly, if $y_i f_{\mathbf{z},\mu}(x_i) < -1$, one still has $y_i f^d_{\mathbf{z},\mu}(x_i) \le -1$. Then $L_{\mathrm{ramp},\mathbf{z}}(f_{\mathbf{z},\mu}) = L_{\mathrm{ramp},\mathbf{z}}(f^d_{\mathbf{z},\mu})$. Therefore, $f^d_{\mathbf{z},\mu}$ is also a solution of equation (6) and satisfies our requirement.

Now if $f_{\mathbf{z},\mu} = f^*_{\mathbf{z},\mu} + b^*_{\mathbf{z},\mu}$ satisfies

$$|f_{\mathbf{z},\mu}(x_{i_0})| = \min_{1\le i\le m} |f_{\mathbf{z},\mu}(x_i)| \le 1,$$

we then have

$$|b^*_{\mathbf{z},\mu}| \le 1 + |f^*_{\mathbf{z},\mu}(x_{i_0})| \le 1 + \|f^*_{\mathbf{z},\mu}\|_\infty.$$

In this way, we complete the proof. ■

In the following, we shall always choose $f_{\mathbf{z},\mu}$ as in lemma 7. According to our proof, such kind of solutions can be easily constructed even though the obtained ones from the algorithm do not meet the requirement. Next, we find a function space covering $f_{\mathbf{z},\mu}$ when $\mathbf{z}$ runs over all possible samples.

Lemma 8 *For every $\mu > 0$, we have $f^*_{\mathbf{z},\mu} \in \mathcal{H}_K$ and*

$$\|f^*_{\mathbf{z},\mu}\|_K \le \kappa\Omega(f^*_{\mathbf{z},\mu}) \le \frac{\kappa}{\mu},$$

where $\kappa = \sup_{x,y\in X}\sqrt{|K(x,y)|}$.

Proof It is trivial that $f^*_{\mathbf{z},\mu} \in \mathcal{H}_K$. By the reproducing property (see Aronszajn, 1950), for $f^*_{\mathbf{z},\mu} = \sum_{i=1}^m \alpha^*_{i,\mathbf{z}} y_i K(x, x_i)$,

$$\|f^*_{\mathbf{z},\mu}\|_K = \left(\sum_{i,j=1}^m \alpha^*_{i,\mathbf{z}}\alpha^*_{j,\mathbf{z}}K(x_i,x_j)\right)^{1/2} \le \kappa\left(\sum_{i,j=1}^m \alpha_{i,\mathbf{z}}\alpha_{j,\mathbf{z}}\right)^{1/2} = \kappa\Omega(f^*_{\mathbf{z},\mu}).$$

Due to the definition of $f^*_{\mathbf{z},\mu}$, we have

$$\mathcal{R}_{\mathrm{ramp},\mathbf{z}}(f_{\mathbf{z},\mu}) + \mu\Omega(f^*_{\mathbf{z},\mu}) \le \mathcal{R}_{\mathrm{ramp},\mathbf{z}}(0) + \mu\Omega(0) \le 1.$$

This gives $\Omega(f^*_{\mathbf{z},\mu}) \le \frac{1}{\mu}$, and completes the proof. ■

From Lemma 7, Lemma 8 and the relation

$$\|f\|_\infty \le \kappa\|f\|_K, \quad \forall f \in \mathcal{H}_K,$$

we know that $f_{\mathbf{z},\mu}$ lies in

$$\mathcal{F}_\mu = \left\{f = f^* + b^* : \|f^*\|_K \le \frac{\kappa}{\mu} \text{ and } |b^*| \le 1 + \frac{\kappa^2}{\mu}\right\}. \tag{28}$$

Now we are in the position to prove the main theorem in Section 2. Our analysis mainly focus on estimating the sample error $\mathcal{S}(m,\mu,\lambda)$.

Proof of Theorem 5. We fist estimate $\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_\lambda) - \mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_\lambda)$ by considering the random variable ς_i defined by (19) with $f = f_\lambda$. As $L_{\mathrm{ramp}} : \mathbb{R} \to [0,1]$, there holds $|\varsigma_i - \mathbb{E}\varsigma_i| \le 2$. Then by the Hoeffding inequality (see, e.g., Cucker and Zhou, 2007, Corollary 3.6), with probability at least $1-\delta/2$, we have

$$\mathcal{R}_{L_{\mathrm{ramp}},\mathbf{z}}(f_\lambda) - \mathcal{R}_{L_{\mathrm{ramp}},\rho}(f_\lambda) \le \sqrt{\frac{8\log\frac{2}{\delta}}{m}}. \tag{29}$$

For the term $\mathcal{R}_{L_{\text{ramp}},\rho}(f_{\mathbf{z},\mu}) - \mathcal{R}_{L_{\text{ramp}},\mathbf{z}}(f_{\mathbf{z},\mu})$, note that $f_{\mathbf{z},\mu}$ varies with samples. In order to obtain the corresponding upper bound, we shall apply the uniform concentration inequality to the function set $\mathcal{F}_\mu$. One can directly use Theorem 8 in Bartlett and Mendelson (2003) to deal with this term and find with probability at least $1 - \delta/2$,

$$\mathcal{R}_{L_{\text{ramp}},\rho}(f_{\mathbf{z},\mu}) - \mathcal{R}_{L_{\text{ramp}},\mathbf{z}}(f_{\mathbf{z},\mu}) \leq \mathbb{E}_{\mathbf{z}}\mathbb{E}_\sigma \left[\sup_{g\in\tilde{\mathcal{F}}} \left| \frac{2}{m}\sum_{i=1}^m \sigma_i g(x_i, y_i)\right|\right] + \sqrt{\frac{8\log\frac{4}{\delta}}{m}} \tag{30}$$

where $\tilde{\mathcal{F}} := \{(x,y) \to L_{\text{ramp}}(yf(x)) - L_{\text{ramp}}(0) : f \in \mathcal{F}\}$ and $\sigma_1, \cdots, \sigma_m$ are independent uniform $\{-1, +1\}$-valued random variables. As the ramp loss is Lipschitz with constant 1, we further bound the first term in the right-hand side by the result of Bartlett and Mendelson (2003, Theorem 12) as

$$\begin{aligned}
& \mathbb{E}_{\mathbf{z}}\mathbb{E}_\sigma \left[\sup_{g\in\tilde{\mathcal{F}}} \left| \frac{2}{m}\sum_{i=1}^m \sigma_i g(x_i, y_i)\right|\right] \\
\leq\ & 2\mathbb{E}_{\mathbf{z}}\mathbb{E}_\sigma \left[\sup_{f\in\mathcal{F}} \left| \frac{2}{m}\sum_{i=1}^m \sigma_i f(x_i)\right|\right] \\
\leq\ & 2\mathbb{E}_{\mathbf{z}}\mathbb{E}_\sigma \left[\sup_{\{f^*\in\mathcal{H}_\mathcal{K}:\|f^*\|_\mathcal{K}\leq\frac{\kappa}{\mu}\}} \left| \frac{2}{m}\sum_{i=1}^m \sigma_i f^*(x_i)\right|\right] + \frac{2}{\sqrt{m}} + \frac{2\kappa^2}{\mu\sqrt{m}} \\
\leq\ & \frac{6\kappa^2}{\mu\sqrt{m}} + \frac{2}{\sqrt{m}}.
\end{aligned}$$

Here, the last inequality is from Lemma 22 in Bartlett and Mendelson (2003). Combining the above bound and (29), (30), we then have with probability at least $1 - \delta$,

$$\mathcal{S}(m, \mu, \lambda) \leq (2 + \eta)\sqrt{\frac{8\log\frac{4}{\delta}}{m}} + \frac{6\kappa^2}{\mu\sqrt{m}} + \frac{2}{\sqrt{m}}.$$

Finally, we let $\mu = m^{-\frac{\beta+1}{4\beta+2}}$ and $\lambda = m^{-\frac{1}{4\beta+2}}$. Then $\eta = \frac{\mu}{\lambda} = m^{-\frac{\beta}{4\beta+2}} \leq 1$. Therefore, by Theorem 2 and Theorem 4, we can derive the bound (20) with $\tilde{c} = 15 + 2c_\beta + 6\kappa^2$. This completes our proof. ■

References

L.T.H. An and P.D. Tao. The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Annals of Operations Research*, 133(1):23–46, 2005.

L.T.H. An, P.D. Tao, and L.D. Muu. Numerical solution for optimization over the efficient set by DC optimization algorithms. *Operations Research Letters*, 19(3):117–128, 1996.

N. Aronszajn, Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

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

P.L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2003.

P.S. Bradley and O.L. Mangasarian. Massive data discrimination via linear support vector machines. *Optimization Methods and Software*, 13(1):1–10, 2000.

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

A. Chipperfield, P. Fleming, H. Pohlheim, and C. Fonseca. Genetic algorithm toolbox user's guide. *Research Report*, 1994.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Trading convexity for scalability. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 201–208. ACM, 2006a.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Large scale transductive SVMs. *Journal of Machine Learning Research*, 7:1687–1712, 2006b.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

F. Cucker and D.X. Zhou. *Learning Theory: an Approximation Theory Viewpoint*. Cambridge University Press, 2007.

K. De Brabanter, K. Pelckmans, J. De Brabanter, M. Debruyne, J.A.K. Suykens, M. Hubert, and B. De Moor, Robustness of kernel based regression: a comparison of iterative weighting schemes. In *Proceedings of the 19th International Conference on Artificial Neural Networks,* pages 100–110, 2009.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

R. Horst and T. Hoang. *Global Optimization: Deterministic Approaches*. Springer Verlag, 1996.

F.M. Fung and O.L. Mangasarian. A feature selection Newton method for support vector machine classification. *Computational Optimization and Applications*, 28(2):185–202, 2004.

R. Horst and N.V. Thoai. DC programming: overview. *Journal of Optimization Theory and Applications*, 103(1):1–43, 1999.

X. Huang, J. Xu, X. Mu, and S. Wang. The hill detouring method for minimizing hinging hyperplanes functions. *Computers & Operations Research*, 39(7):1763–1770, 2012a.

X. Huang, J. Xu, and S. Wang. Exact penalty and optimality condition for nonseparable continuous piecewise linear programming. *Journal of Optimization Theory and Applications*, 155(1):145–164, 2012b.

V. Kecman and I. Hadzic. Support vectors selection by linear programming. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, volume 5, pages 193–198. IEEE, 2000.

Y. Lin. A note on margin-based loss functions in classification. *Statistics & Probability Letters*, 68(1):73–82, 2004.

O.L. Mangasarian. Exact 1-norm support vector machines via unconstrained convex differentiable minimization. *Journal of Machine Learning Research*, 7:1517–1530, 2006.

O.L. Mangasarian. Absolute value equation solution via concave minimization. *Optimization Letters*, 1(1):3–8, 2007.

L. Mason, J. Baxter, P.L. Bartlett, and M. Frean. Boosting algorithms as gradient descent in function space. In S.A. Solla, T.K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, 12:512–518, Cambridge, MA, MIT Press, 2000.

M. Porembski. Cutting planes for low-rank-like concave minimization problems. *Operations Research*, pages 942–953, 2004.

S. Smale and D.X. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1(1):17–41, 2003.

A. Smola, B. Schölkopf, and G. Rätsch. Linear programs for automatic accuracy control in regression In *Proceedings of the 9th International Conference on Artificial Nerual Networks*, No. 470, pages 575–580, 1999.

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

J. Shu and I.A. Karimi. Efficient heuristics for inventory placement in acyclic networks. *Computers & Operations Research*, 36(11):2899–2904, 2009.

I. Steinwart. How to compare different loss functions and their risks. *Constructive Approximation*, 26(2):225–287, 2007.

I. Steinwart and A. Christmann. *Support Vector Machines*. New York: Springer, 2008.

J.A.K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

K. Wang, P. Zhong, and Y. Zhao. Training robust support vector regression via DC program. *Journal of Information and Computational Science*, 7(12):23852394, 2010.

Q. Wu and D.X. Zhou. SVM soft margin classifiers: linear programming versus quadratic programming. *Neural Computation*, 17(5):1160–1187, 2005.

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

A.L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4): 915–936, 2003.

T. Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5:1225–1251, 2004.

Y. Zhao and J. Sun. Robust support vector regression in the primal. *Neural Networks*, 21(10): 1548–1555, 2008.

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, 1-norm Support Vector Machines. In S. Thrun, L.K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, 16:49–56, Cambridge, MA, MIT Press, 2004.