Robust Gradient Learning With Applications

Yunlong Feng, Yuning Yang, and Johan A. K. Suykens, *Fellow, IEEE*

***Abstract*— This paper addresses the robust gradient learning (RGL) problem. Gradient learning models aim at learning the gradient vector of some target functions in supervised learning problems, which can be further used to applications, such as variable selection, coordinate covariance estimation, and supervised dimension reduction. However, existing GL models are not robust to outliers or heavy-tailed noise. This paper provides an RGL framework to address this problem in both regression and classification. This is achieved by introducing a robust regression loss function and proposing a robust classification loss. Moreover, our RGL algorithm works in an instance-based kernelized dictionary instead of some fixed reproducing kernel Hilbert space, which may provide more flexibility. To solve the proposed nonconvex model, a simple computational algorithm based on gradient descent is provided and the convergence of the proposed method is also analyzed. We then apply the proposed RGL model to applications, such as nonlinear variable selection and coordinate covariance estimation. The efficiency of our proposed model is verified on both synthetic and real data sets.**

***Index Terms*— Gradient learning (GL), instance-based kernelized dictionary, nonlinear variable selection, regularization, robustness.**

I. Introduction and Motivation

THE gradient learning (GL) model proposed in [1] aims at learning the gradients of the regression function, which is directly driven by variable selection and coordinate covariance problems. However, in real-life applications, data sets might be contaminated by outliers or heavy-tailed noise, which may appear in both response or the predictors. In this case, the GL model cannot help in learning gradients. This paper presents a framework of robust GL (RGL) model to learn the gradients of the regression function robustly. To explain the motivation of learning gradients, we start with an overview of the GL algorithm to illustrate.

Manuscript received March 28, 2014; revised April 17, 2015; accepted April 18, 2015. Date of publication May 11, 2015; date of current version March 15, 2016. This work was supported in part by the IWT: Ph.D./Post-Doctoral Grants through the SBO POM Project under Grant 100031, in part by iMinds Medical Information Technologies under Grant SBO 2014, in part by the European Research Council within the European Union Seventh Framework Programme (FP7/2007-2013) through the ERC AdG ADATADRIVE-B Project under Grant 290923, in part by the Research Council KUL through the MaNet Project under Grant GOA/10/09, OPTEC Project under Grant CoE PFV/10/002 and Grant BIL12/11T, in part by the Ph.D./Post-Doctoral Grants, in part by the Flemish Government: FWO: Ph.D./Post-Doctoral Grants through the Structured Systems Project under Grant G.0377.12 and Tensor Based Data Similarity Project under Grant G.088114N, and in part by the Belgian Federal Science Policy Office: IUAP P7/19 through the Dynamical Systems, Control and Optimization 2012–2017.

The authors are with the Center for Dynamical Systems, Signal Processing and Data Analytics, Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven 3001, Belgium (e-mail: yunlong.feng@esat.kuleuven.be; yuning.yang@esat.kuleuven.be; Johan.Suykens@esat.kuleuven.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2015.2425215

A. Overview of the Gradient Learning Algorithms

Let us assume that X is the input variable that takes values in $\mathcal{X} \subset \mathbb{R}^n$, Y stands for the response variable that takes value in $\mathcal{Y} \subset \mathbb{R}$. Assume that we are given a set of observations $\mathbf{z} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$, which are drawn independent identically distributed (i.i.d) from some unknown probability distribution over $\mathcal{X} \times \mathcal{Y}$. We further assume that the regression model is given as

$$Y = f^\star(X) + \epsilon \tag{1}$$

where ϵ is the noise term and $\mathbb{E}(\epsilon \mid X = \mathbf{x}) = 0$. The vector-valued function $\nabla f^\star$ is denoted as the gradient of the regression function $f^\star$, which is given as

$$\nabla f^\star(\mathbf{x}) = \left(\frac{\partial}{\partial x_1} f^\star(\mathbf{x}), \ldots, \frac{\partial}{\partial x_n} f^\star(\mathbf{x})\right)^T \quad \mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{X}. \tag{2}$$

The estimation of the gradient of the regression function is motivated by the following Taylor series expansion: $f^\star(\mathbf{x}) \approx f^\star(\mathbf{x}') + \nabla f^\star(\mathbf{x}') \cdot (\mathbf{x} - \mathbf{x}')$, for $\mathbf{x} \approx \mathbf{x}'$, and $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. Evaluating at the data points $\{\mathbf{x}_i\}_{i=1}^m$, one gets $f^\star(\mathbf{x}_i) \approx f^\star(\mathbf{x}_k) + \nabla f^\star(\mathbf{x}_k) \cdot (\mathbf{x}_i - \mathbf{x}_k)$, for $\mathbf{x}_i \approx \mathbf{x}_k$. Integrating the ideas of local regression and utilizing an empirical risk minimization strategy endowed with a least squares loss, the gradient of the regression function can be estimated. More explicitly, let $\mathcal{H}_K$ be a reproducing kernel Hilbert space (RKHS) induced by a Mercer kernel K and further denote $\mathcal{H}_K^n$ as an n-fold RKHS [1]. To learn an empirical estimator $\vec{g}_\mathbf{z}$ for the vector-valued function $\nabla f^\star$, [1] proposed the following algorithm:

$$\vec{g}_\mathbf{z} = \arg \min_{\vec{g}=(g_1,\ldots,g_n)^T \in \mathcal{H}_K^n} \left\{ \frac{1}{m^2} \sum_{i,k=1}^m \omega_{ik} \left(y_i - y_k + \vec{g}(\mathbf{x}_i) \cdot (\mathbf{x}_k - \mathbf{x}_i)\right)^2 + \lambda \|\vec{g}\|_\mathcal{K}^2 \right\} \tag{3}$$

where λ is a regularization parameter, $\|\vec{g}\|_\mathcal{K}^2 = \sum_{j=1}^n \|g_j\|_\mathcal{K}^2$ and ω_{ik} are weights, a typical choice of which can be $\omega_{ik} = \exp\{-(\|\mathbf{x}_i - \mathbf{x}_k\|^2/2s^2)\}$ with some constant $s > 0$. Note that the above model is presented to deal with the GL problem in a regression setting. In fact, the GL problem can also be proceeded in the classification setting by replacing the least squares loss with the hinge loss or logistic loss, as investigated in [1] and [2].

The curse of dimensionality reminds us that the proposed GL model (3) may not be able to help in high-dimensional spaces due to using the localization dependent weights $\{\omega_{ik}\}_{i,k=1}^m$. However, it is a common belief that,

2162-237X © 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

in high-dimensional data analysis, high-dimensional data usually lies on a manifold which admits a much smaller intrinsic dimensionality for many real-life applications. Following this line, model (3) has also been interpreted in a manifold setting in [3].

To explain more clearly on the applications of the GL model, let us start with the observation that $\nabla f^\star(\mathbf{x})$ given in (2) indicates how the regression function $f^\star(\cdot)$ changes with respect to its coordinates at the point $\mathbf{x} \in \mathcal{X}$. The general idea of using the estimated gradient function for variable selection and coordinate covariance estimation is indicated by the following two observations. First, the norm of each ingredient $\partial f^\star/\partial x_j$ indicates the salience of the corresponding variable, e.g., the larger the norm is, the more relevant the variable will be. Second, the inner product between components of $\nabla f^\star$ specifies the covariance between corresponding coordinates. Specifically, Mukherjee and Zhou [1] define the following relative magnitude of the norm of the coordinate to provide a rank for variable selection:

$$r_j = \frac{\|g_{\mathbf{z},j}\|_{\mathcal{K}}^2}{\sum_{l=1}^n \|g_{\mathbf{z},j}\|_{\mathcal{K}}^2}, \quad j = 1, \ldots, n$$

where $\vec{g}_{\mathbf{z}} = (g_{\mathbf{z},1}, \ldots, g_{\mathbf{z},n})^T$. Correspondingly, the empirical covariance matrix $[\langle g_{\mathbf{z},j}, g_{\mathbf{z},l}\rangle_{\mathcal{K}}]_{j,l=1}^n$ can be used to characterize the covariance between the coordinates. The GL model has also been extended to the cases, where the gradient and the regression function/classifier can be learned simultaneously [4]. References [5]–[7] investigated the GL problem within a multitask learning framework while [8] studied a sparsified version of the GL (SGL) model (3) which is driven by simultaneously variable selection and supervised dimension reduction.

B. Related Work and Our Contributions

So far, we have presented an overview of the GL and explained its applications in nonlinear variable selection and coordinate covariance estimation. In this section, we will list related work that focuses on the above two applications and discuss their limitations. The contribution of our work will also be summarized in this section.

Concerning the variable selection problem, there is an enormous amount of the literature. The interested reader may refer to [9] for a detailed survey. For linear models, numerous variable selection algorithms have been proposed, among which the most well-known ones include lasso [10], Smoothly Clipped Absolute Deviation [11], adaptive lasso [12], elastic-net regularization [13], group lasso [14], and Minimax Concave Penalty regularization [15]. The robust variable selection problem has also been investigated for linear models in [16]–[18]. We are also aware that there are many existing publications dealing with the nonlinear variable selection problems, e.g., [19]–[27]. Instead of selecting the variable directly, the GL model (3) learns the gradient at each point in the instance space $\mathcal{X}$ and can be further applied to estimate the coordinate covariance [1], [2] and supervised dimension reduction. On the other hand, most existing heuristics require assumptions on the regression model (1), e.g., semiparametric model, additive model, and so on. However, the GL model (3) does not apply any prerequisite conditions on $f^\star$. These distinguish the GL model (3) from almost all the above mentioned methods.

Despite the empirical success, there are still some limitations of the GL model (3) and also its extensions mentioned above. First, the above mentioned GL models cannot be resistant to outliers due to using the unbounded least-squares loss, which is not robust [28]–[30]. However, in real-life applications, data might be contaminated by heavy-tailed errors or outliers. In this case, the least-squares criterion is reputed to be not efficient. Second, note that the above mentioned GL models are restricted to the RKHS generated by a Mercer kernel. However, Roth [31], Luss and d'Aspremont [32], and Ying *et al.* [33] remind us that in some situations this requirement may not be able to be satisfied. More flexibility may be obtained if we are able to relax this requirement. This will be further explained in the following context.

To overcome these limitations, we propose a framework of RGL model. Our contributions can be summarized as follows.

1) The GL model that we propose can learn the gradient function robustly in regression and classification problems due to using a robust loss function instead of the least squares loss. Using a robust loss function is motivated by M-estimation in robust statistics [28]. Specifically, we employ a nonconvex robust regression loss function and also propose a new nonconvex classification loss. Numerical experimental results verify the robustness of the proposed model.
2) The GL model that we propose is in fact a framework of learning the gradients. This assertion relies on the following three observations.
   a) For the proposed learning model, we study a general regularizer. The general regularizer can be reduced to some specific regularizers by choosing different indices. The choice is usually made according to the type of the data sets.
   b) Other robust loss functions can also be applied to the model, although we have chosen a specific one.
   c) Instead of working in a RKHS, our model works in a instance-based kernelized dictionary. Thus, the restriction of the positive definite condition on the kernel function is removed, which further yields more flexibility as illustrated in the paper.
3) A gradient descent iterative soft thresholding method is provided to solve the proposed nonconvex model. By showing that the gradient of the loss function is Lipschitz continuous, convergence of the method to a stationary point is proved.

Organization: This paper is organized as follows. In Section II, we present our RGL model. We then introduce and compare some other robust loss functions in regression and classification. Regularizers in which we are interested in this paper will also be discussed. We also present an illustration example in this section to show the efficiency of our model. Section III is concerned with the computational aspects of the proposed model. A gradient descent algorithm is proposed and convergence analysis will also be provided. We then

discuss on the generalization property of proposed model. Numerical experiments on both synthetic and real data sets will be implemented in Section IV. We end this paper in Section V with conclusions and point out several promising extensions of our proposed model.

II. Proposed RGL Model and Properties

This section presents the proposed RGL models in regression and classification. We then discuss properties of the proposed RGL models. An illustrative experimental example will be provided to show the effectiveness of the proposed RGL model in a regression setting. Throughout this paper, the indices i, k denote instances, while the indices j, l refer to features.

A. Proposed RGL Model

Let $\mathcal{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a kernel function which is not restricted to be a positive definite kernel. We denote $\mathcal{H}_{\mathcal{K},\mathbf{z}}$ as a linear span over the instance-based kernelized dictionary $\{\mathcal{K}(\mathbf{x}, \mathbf{x}_i)\}_{i=1}^m$ with coefficients $\{\alpha_i\}_{i=1}^m$. More explicitly, $\mathcal{H}_{\mathcal{K},\mathbf{z}}$ is defined as the following function set:

$$\left\{ g : g(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}),\ \alpha_i \in \mathbb{R}, \quad i = 1, \ldots, m, \mathbf{x} \in \mathcal{X} \right\}.$$

Let $\vec{g} = (g_1, g_2, \ldots, g_n)^T \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^n$ with the coefficient $A = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_m)$, where $\boldsymbol{\alpha}_i = (\alpha_{1,i}, \ldots, \alpha_{n,i})^T$ for $i = 1, \ldots, m$ and

$$\begin{aligned} g_j : \mathcal{X} &\to \mathbb{R} \\ \mathbf{x} &\longmapsto g_j(\mathbf{x}), \quad j = 1, \ldots, n. \end{aligned}$$

Here $\mathcal{H}_{\mathcal{K},\mathbf{z}}^n$ is an n-fold $\mathcal{H}_{\mathcal{K},\mathbf{z}}$ and defined by

$$\mathcal{H}_{\mathcal{K},\mathbf{z}}^n = \{(g_1, \ldots, g_n)^T \,|\, g_j \in \mathcal{H}_{\mathcal{K},\mathbf{z}}, \quad j = 1, \ldots, n\}.$$

We denote $\mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g})$ as the empirical risk when taking $\vec{g}$ as an empirical estimator of $\nabla f^{\star}$. Denote ω_{ik} values as weights which are given by $\omega_{ik} = \exp\{-(\|\mathbf{x}_i - \mathbf{x}_k\|^2/2s^2)\}$ with some $s > 0$.

In the regression setting, $\mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g})$ is defined as

$$\mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g}) = \frac{1}{m^2} \sum_{i,k=1}^{m} \omega_{ik} \rho((y_i - y_k) + \vec{g}(\mathbf{x}_i) \cdot (\mathbf{x}_k - \mathbf{x}_i))$$

where $\rho(\cdot)$ is a robust distance-based regression loss. Turning to the classification setting, $\mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g})$ is formulated as

$$\mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g}) = \frac{1}{m^2} \sum_{i,k=1}^{m} \omega_{ik} \rho(y_i(y_k + \vec{g}(\mathbf{x}_i) \cdot (\mathbf{x}_i - \mathbf{x}_k)))$$

where $\rho(\cdot)$ is a robust margin-based classification loss.

Based on the above notations, our RGL model takes the following form:

$$\vec{g}_{\mathbf{z}} = \arg \min_{\vec{g} \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^n} \left\{ \mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g}) + \lambda \|A\|_{p,q}^q \right\} \tag{4}$$

where $\lambda > 0$ is a regularization parameter, and

$$\|A\|_{p,q}^q := \sum_{i=1}^{n} \left(\sum_{j=1}^{m} |\alpha_{i,j}|^p \right)^{\frac{q}{p}}$$

with $p \geq 1$ and $q \geq 1$.

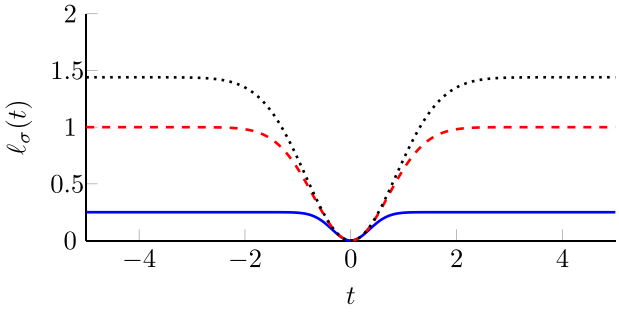


Fig. 1. Plot of the regression loss ℓ_σ for different σ values: $\sigma = 0.5$ for the solid curve, $\sigma = 1$ for the dashed curve, and $\sigma = 1.2$ for the dotted curve.

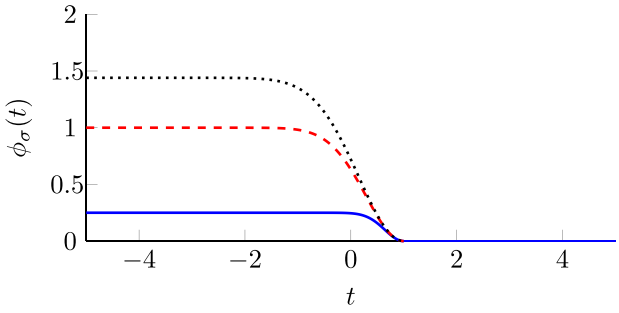


Fig. 2. Plot of the classification loss ϕ_σ for different σ values: $\sigma = 0.5$ for the solid curve, $\sigma = 1$ for the dashed curve, and $\sigma = 1.2$ for the dotted curve.

The robust loss function proposed in this paper will be explicitly defined in Section II-B. The first index of the subscript in the regularizer refers to instances, while the second one represents features. In this paper, we are interested in some specific regularizers, e.g., $(p, q) \in \{(2, 1), (2, 2), (1, 1)\}$. More discussions on the regularization terms will be expanded in Section II-D.

B. Proposed Robust Loss Functions for Gradient Learning

In this section, we introduce two robust loss functions to learn the gradient in regression and classification problems. To distinguish loss functions in regression and classification problems, we denote the regression loss as ℓ_σ and the classification loss as ϕ_σ. With a positive constant σ, the robust regression loss ℓ_σ is defined as

$$\ell_\sigma(t) = \sigma^2(1 - \exp(-t^2/\sigma^2)), \quad t \in \mathbb{R}. \tag{5}$$

To learn the gradient robustly in classification problems, we introduce the following margin-based classification loss ϕ_σ:

$$\phi_\sigma(t) = \sigma^2(1 - \exp(-(1 - t)_+^2/\sigma^2)), \quad t \in \mathbb{R}$$

where $(1 - t)_+ := \max\{1 - t, 0\}$.

Plots of the above two loss functions are shown in Figs. 1 and 2. We postpone discussions on the two loss functions in Section II-D.

C. Illustrative Example

In this section, we first give a simple illustrative example to show the efficiency of our proposed RGL model (4). Detailed numerical experiments will be expanded in Section IV.

We consider the nonlinear variable selection problems in regression. Let the observation data set $\mathbf{z} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{400}$

TABLE I
SELECTION RESULTS WITH 100 RUNS FOR CASE I

Models	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
GL [1]	99	100	100	100	100	0	0	0	0	1
SGL [8]	100	100	100	100	100	0	0	0	0	0
RGL	100	100	100	100	100	0	0	0	0	0

TABLE II
SELECTION RESULTS WITH 100 RUNS FOR CASE II

Models	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
GL [1]	100	100	100	100	100	0	0	0	0	0
SGL [8]	100	100	100	100	100	0	0	0	0	0
RGL	100	100	100	100	100	0	0	0	0	0

with $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,10})^T$ be generated by the following nonadditive model:

$$y_i = x_{i,1}x_{i,2} + (2x_{i,3} - 0.5)^2 + x_{i,4} + x_{i,5} + \epsilon$$

where $\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,5}$ are i.i.d. drawn from the uniform distribution on $[0, 1]$, and ϵ represents the outliers or heavy-tailed noise. Besides the input variables, we introduce five variables, $\mathbf{x}_{i,6}, \ldots, \mathbf{x}_{i,10}$, as the noise variables, which are also i.i.d. and uniformly distributed on $[0, 1]$. The quadratic kernel $\mathcal{K}(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^2$, $\mathbf{x}, \mathbf{x}' \in [0, 1]^{10}$ is used in our experiment.

We use our RGL model (4) with $p = 2$ and $q = 2$ in the regularizer to select the five contributed variable $x_1, \ldots, x_5$ and then compare the selection results with the GL and SGL model, where GL model proposed in [1], and SGL model introduced in [8]. We repeat experiments for 100 times with the observation set $\mathbf{z}$ generated in the following four different situations.

1) *Case I:* The observation data are not contaminated by outliers or noise. Variable selection results are reported in Table I.
2) *Case II:* The observation data are contaminated by Gaussian noise and scaled by 0.05. There are no outliers added to the data. Variable selection results are reported in Table II.
3) *Case III:* The observation data are contaminated by outliers and noise. The added noise is drawn from the chi-square distribution with the degree of freedom 4 and scaled by 0.05. Outliers added to the data are also randomly generated by the chi-square distribution and scaled by 30. The ratio of the outliers added to the observation data is set to be 10%. Experimental results are reported in Table III.
4) *Case IV:* The observation data are only contaminated by noise. The noise is drawn from the Cauchy distribution with the location parameter $a = 2$, scale parameter $b = 4$, and further scaled by 0.05. Experimental results are reported in Table IV.

D. Robust Loss Functions, Instance-Based Kernelized Dictionaries, and Regularizers

This part presents some discussions on the loss function, the kernelized dictionaries, and also the penalty term in model (4).

TABLE III
SELECTION RESULTS WITH 100 RUNS FOR CASE III

Models	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
GL [1]	53	53	68	59	56	38	46	45	44	38
SGL [8]	52	53	69	59	57	38	46	45	43	38
RGL	**98**	**100**	**100**	**100**	**100**	**2**	**0**	**0**	**0**	**0**

TABLE IV
SELECTION RESULTS WITH 100 RUNS FOR CASE IV

Models	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
GL [1]	50	51	82	67	63	37	43	35	37	35
SGL [8]	50	51	82	66	63	38	43	35	37	35
RGL	**96**	**94**	**100**	**100**	**100**	**1**	**3**	**2**	**2**	**2**

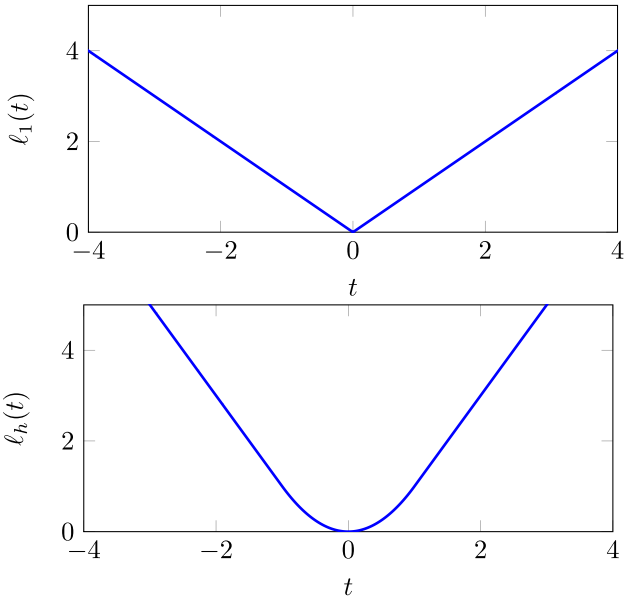


Fig. 3. Plot of the LAD loss ℓ_1 (top) and the Huber's loss ℓ_h (bottom).

Robust statistics [28] reminds us that associated with the robust loss function $\rho(\cdot)$, the proposed GL model (4) is essentially a regularized M-estimator of the gradient vector $\nabla f^\star$. This motivates us to explore other robust loss functions in the GL model (4). For regression problems, two typical examples proposed in the context of robust statistics are the least absolute deviation (LAD) loss $\ell_1(\cdot)$ and the Huber's loss $\ell_h(\cdot)$, where $\ell_1(t) = |t|$, $t \in \mathbb{R}$ and for some fixed $c > 0$

$$\ell_h(t) = \begin{cases} t^2, & |t| \le c \\ 2c|t| - c^2, & |t| > c. \end{cases} \tag{6}$$

Plots of the two loss functions are listed in Fig. 3. It is easy to see that the LAD loss, ℓ_1, penalizes strongly the small errors. Consequently, the estimator obtained by using ℓ_1 loss can be less efficient than the least squares loss when there are no outliers and no heavy-tailed noise. In real-world applications, the distribution of the noise is usually unknown. Therefore, it is necessary to consider some models that can be efficient in both situations. The Huber's loss, ℓ_h, seems to be a good choice. However, either the Huber's loss or

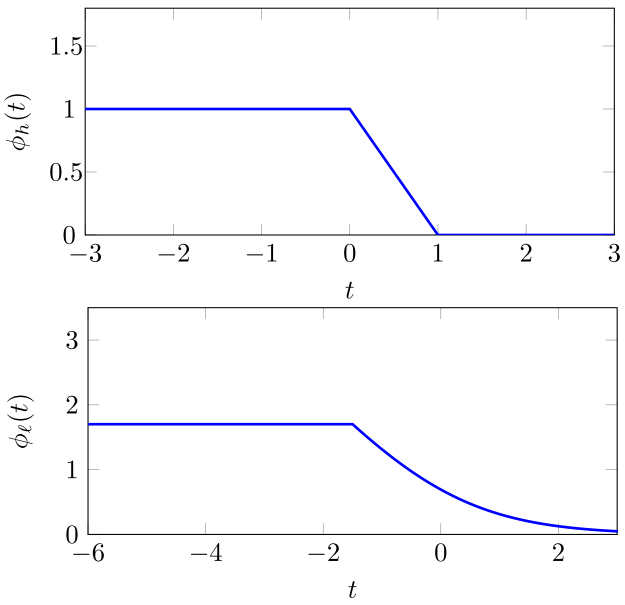


Fig. 4. Plot of the truncated hinge loss ϕ_h (top) and the truncated logistic loss ϕ_ℓ (bottom).

the LAD loss cannot be robust to large gross errors [28]. Concerning the classification problem, some robust classification loss functions have also been proposed in the literature, e.g., truncated hinge loss ϕ_h [34] and truncated logistic loss ϕ_ℓ [35] (see Fig. 4). However, both truncated hinge loss and truncated logistic loss are nonsmooth. This motivates us to introduce the robust regression loss ℓ_σ and robust classification loss ϕ_σ. The loss function ℓ_σ is originally proposed in [36] and [37] and has drawn much attention recently [18], [38], [39]. Some discussions on the properties of the loss function ℓ_σ can be found in [39]. It is easy to see that ℓ_σ can eliminate the influence of large gross errors completely. Concerning the classification loss ϕ_σ, one can also see that ϕ_σ is Fisher consistent according to [40]. Moreover, one can see that ϕ_σ can be taken as an approximate to the squared hinge loss with sufficient large σ and it approximates the hinge loss with proper chosen σ. A commonly nice property of ℓ_σ and ϕ_σ is that their derivatives are Lipschitz continuous, which may lead to computational efficiency, as illustrated in Section III. Unless otherwise specified, in what follows, the RGL model refers to the RGL model associated with the ℓ_σ loss in regression or ϕ_σ loss in classification.

As shown in model (4), the optimization problem searches within the instance-based kernelized dictionary $\{\mathcal{K}(\mathbf{x}, \mathbf{x}_i)\}_{i=1}^m$, which is capable of capturing the nonlinearity of the vector-valued gradient function ∇f_ρ. Meanwhile, by removing the positive definite condition on the kernel $\mathcal{K}$, we get more flexibility. On the other hand, the RGL model is more general. As pointed out in [3], by choosing specific kernel function, the Outer Product of Gradients method proposed in [25] can be cast as a special case of the model (4). Moreover, by substituting the loss function with the least squares loss in the RGL model (4) and considering certain RKHS, the proposed model can be reduced to the GL or SGL model with properly chosen regularizers. Second, this relaxation enables us to capture both feature-wise and instance-wise information.

For regularized learning schemes, such feature-wise or instance-wise information is usually revealed in terms of their regularizers. For instance, by choosing some specific values of p and q in the regularizer of model (4), it can be reduced to some frequently adopted ones. This property well explains the flexibility of learning over the instance-based kernelized dictionary instead of the RKHS. In this paper, we mainly focus on the following three regularizers. As shown later, the computational algorithm that we will propose can be used to deal with the optimization problem (4) associated with the following three regularizers.

1) $\ell_{2,2}$ *Regularizer:* For any $\vec{g} \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^n$, the $\ell_{2,2}$ regularizer is defined as $\|A\|_{2,2}^2 = \sum_{i=1}^n \sum_{j=1}^m |\alpha_{i,j}|^2$. The $\ell_{2,2}$ regularizer possesses both good approximation and stable computational ability. Note that the regularizer used in (3) falls into this type.
2) $\ell_{2,1}$ *Regularizer:* For any $\vec{g} \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^n$, the $\ell_{2,1}$ regularizer is given by $\|A\|_{2,1} = \sum_{i=1}^n (\sum_{j=1}^m |\alpha_{i,j}|^2)^{1/2}$. Using the $\ell_{2,1}$ regularizer, we can obtain similar sparseness across features while the feature-wise sparseness allows us to select those informative variables in a direct manner. This can be especially helpful in some learning scenarios such as the multitask learning [41]. Recently, this type of regularizer has also been employed to deal with the problem of simultaneous nonlinear variable selection and dimension reduction in [8].
3) $\ell_{1,1}$ *Regularizer:* For any $\vec{g} \in \mathcal{H}_{\mathcal{K},\mathbf{z}}^n$, the $\ell_{1,1}$ regularizer is defined as $\|A\|_{1,1} = \sum_{i=1}^n \sum_{j=1}^m |\alpha_{i,j}|$. It has been adopted in the sparse coding literature [42]. Besides the sparseness of the features, another nice property of the $\ell_{1,1}$-norm regularizer $\|\vec{g}\|_{1,1}$ is that it simultaneously pursues instance-wise and feature-wise sparseness. That is, the instance-wise information has also been liberated via this regularizer. In the GL context, the instance-wise sparseness can be particularly meaningful if one considers to learn the gradient function and the classifier/regression function simultaneously as addressed in [4] and [5].

III. Computational Algorithms and Convergence Analysis

Although the proposed RGL model (4) enjoys its robustness from the robust loss ℓ_σ, the nonconvexity of ℓ_σ also leads to computational difficulty. To tackle this problem, in this section, we propose a gradient descent based iterative soft thresholding algorithm to solve the proposed RGL model (4). The algorithm is simple and easy to implement. For simplicity, we mainly discuss the algorithm for the (4) associated with the regression loss ℓ_σ, while the algorithm for the classification version can be obtained similarly. By showing that the gradient of the loss term of (4) is Lipschitz continuous, the convergence of the proposed algorithm to a stationary point of (4) is also obtained.

For the convenience of our analysis, we rewrite the model (4) as

$$\min_{A \in \mathbb{R}^{n \times m}} \mathcal{E}(A) + \lambda \|A\|_{p,q}^q$$

where the fidelity term $\mathcal{E}(A)$ associated with ℓ_σ for regression is given by

$$\mathcal{E}(A) = \frac{\sigma^2}{2} \sum_{i,k=1}^{m} \omega_{ik} \big(1 - \exp\big(- Z_{ik}^2/\sigma^2\big)\big).$$

Here, $Z_{ik} = y_i - y_k + (\mathbf{x}_k - \mathbf{x}_i)^T A K_k$, and K_k is the kth column of the kernel matrix K with $K_{ik} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_k)$. The gradient of $\mathcal{E}$ at A is given by

$$\nabla \mathcal{E}(A) = \sum_{i,k=1}^{m} \omega_{ik} \exp\big(- Z_{ik}^2/\sigma^2\big) Z_{ik} (\mathbf{x}_k - \mathbf{x}_i) K_k^T.$$

A. Proposed Algorithm and Its Convergence

Since $\mathcal{E}(A)$ is a nonconvex function, (4) becomes a nonconvex unconstrained minimization problem. Supposing that $\overline{A}$ is a stationary point of (4), we then have

$$0 \in \nabla \mathcal{E}(\overline{A}) + \lambda \partial \|\overline{A}\|_{p,q}^q \tag{7}$$

where $\partial \|\overline{A}\|_{p,q}^q$ is the subgradient of $\|\cdot\|_{p,q}^q$ at $\overline{A}$. Denoting $\overline{D} = \overline{A} - \frac{1}{\tau} \nabla \mathcal{E}(\overline{A})$ with $\tau > 0$, (7) can be rewritten as

$$0 \in \overline{A} - \overline{D} + \frac{\lambda}{\tau} \partial \|\overline{A}\|_{p,q}^q. \tag{8}$$

It is easy to see that $\overline{A}$ satisfies (8) if and only if it is an optimal solution of the following unconstrained convex optimization problem:

$$\min_{A \in \mathbb{R}^{n \times m}} \frac{1}{2} \|\overline{D} - A\|_F^2 + \frac{\lambda}{\tau} \|A\|_{p,q}^q.$$

Observing this, we propose the following fixed point iterative scheme:

$$A^{t+1} = \arg \min_{A \in \mathbb{R}^{n \times m}} \frac{1}{2} \|D^{t+1} - A\|_F^2 + \frac{\lambda}{\tau} \|A\|_{p,q}^q \tag{9}$$

where $D^{t+1} = A^t - \frac{1}{\tau} \nabla \mathcal{E}(A^t)$.

In fact, the iterative scheme (9) belongs to the class of proximal gradient methods, which has been employed to deal with the ℓ_1-norm regularization problems [43]. The algorithm is also known as the forward–backward splitting [44], [45]. Generally speaking, the forward–backward splitting method aims at solving the following convex optimization problem:

$$\min_x \ f(x) + g(x) \tag{10}$$

where f is assumed to be convex and continuously differentiable, while g is convex but allowed to be nonsmooth. Associated with a regularization parameter $\eta > 0$, the proximity operator [45] of g at x, $\text{prox}_{g,\eta}$, is defined as

$$\text{prox}_{g,\eta}(x) = \arg\min_y \frac{1}{2} \|x - y\|_F^2 + \eta g(y).$$

Correspondingly, (10) can be solved via the following proximal gradient method:

$$x^{t+1} = \text{prox}_{g,\eta}(x^t - \eta \nabla f(x^t)). \tag{11}$$

One can see that the optimization problem (9) has a similar iteration scheme as (11). However, there is an essential difference between the optimization problem (4) and (10), which is caused by the nonconvexity of the empirical risk term $\mathcal{E}(A)$. The nonconvex property is also a barrier in analyzing the convergence of the proposed algorithm.

Note that when $(p, q) \in \{(1, 1), (2, 1), (2, 2)\}$, (9) possesses a closed-form solution, which is given by the soft thresholding operator $S_{p,q}^{\lambda/\tau}$. We describe A^{t+1} via the thresholding operators explicitly given below.

We first consider the case where $(p, q) = (1, 1)$. In this case, we know that

$$\begin{aligned} A^{t+1} &= S_{1,1}^{\lambda/\tau}(D^{t+1}) \\ &= \text{sgn}(D^{t+1}) \circ \max\left\{|D^{t+1}| - \frac{\lambda}{\tau}, 0\right\} \end{aligned}$$

where $\text{sgn}(\cdot)$ denotes the sign function, and $\circ$ denotes the Hadamard operator, i.e., entry-wise product.

To derive the formulation of A^{t+1} where $(p, q) = (2, 1)$, we first write D^{t+1} as $(D^{t+1})^T = [d_1^{t+1}, d_2^{t+1}, \ldots, d_n^{t+1}] \in \mathbb{R}^{m \times n}$, where $(d_i^{t+1})^T$ refers to the ith row of D^{t+1}. Following from [46], we have:

$$S_{2,1}^{\lambda/\tau}\big(d_i^{t+1}\big) = \begin{cases} 0, & \text{if } \big\|d_i^{t+1}\big\|_F \le \frac{\lambda}{\tau} \\ \dfrac{\big\|d_i^{t+1}\big\|_F - \lambda/\tau}{\big\|d_i^{t+1}\big\|_F} d_i^{t+1}, & \text{otherwise.} \end{cases}$$

For the $\ell_{2,1}$ regularizer, we note that the ith row of A^{t+1} is only dependent on d_i^{t+1}. Therefore, (9) can be reduced into p subproblems, each of which involves only the ith row of A^{t+1} and D^{t+1}. Therefore

$$(A^{t+1})^T \triangleq \Big[S_{2,1}^{\lambda/\tau}\big(d_1^{t+1}\big), S_{2,1}^{\lambda/\tau}\big(d_2^{t+1}\big), \ldots, S_{2,1}^{\lambda/\tau}\big(d_n^{t+1}\big)\Big].$$

The optimal solution for $(p, q) = (2, 2)$ is easy to obtain, this is due to the fact that (8) is differentiable everywhere in this case. Consequently, we have

$$A^{t+1} = S_{2,2}^{\lambda/\tau}(D^{t+1}) = \frac{D^{t+1}}{1 + 2\lambda/\tau}.$$

Based on the soft thresholding operators obtained above, our proposed algorithm for GL is described in Algorithm 1. At the $(t+1)$th iteration, we first compute the gradient step $D^{t+1} = A^t - (1/\tau)\nabla \mathcal{E}(A^t)$, where $(1/\tau) > 0$ is the step size. The proximity operator step $A^{t+1} = S_{p,q}^{\lambda/\tau}(D^{t+1})$ is then performed on D^{t+1}. The algorithm will stop whenever $\|A^{t+1} - A^t\|_F \le \epsilon$, where ϵ is a certain given positive number. By choosing a suitable step size, the convergence of Algorithm 1 can be derived, which will be shown in Section III-B.

For the computational algorithm of the RGL model in classification, the empirical risk is defined as

$$\mathcal{E}(A) = \frac{1}{2} \sum_{i,k=1}^{m} \omega_{ik} \phi_\sigma\big(y_i (y_k + (\mathbf{x}_i - \mathbf{x}_k)^T A K_k)\big).$$

Since ϕ_σ is differentiable everywhere, the derivative of ϕ_σ is given by

$$\phi_\sigma'(\cdot) = -2 \exp\big(-(1 - \cdot)_+^2/\sigma^2\big)(1 - \cdot)_+.$$

Therefore, $\nabla \mathcal{E}(A)$ equals

$$\sum_{i,k=1}^{m} \omega_{ik} \phi_\sigma'\big(y_i \big(y_k + (\mathbf{x}_i - \mathbf{x}_k)^T A K_k\big)\big) y_i (\mathbf{x}_i - \mathbf{x}_k) K_k^T.$$

Algorithm 1 Gradient Descent Iterative Soft Thresholding Algorithm for Robust Gradient Learning (4)

Input: data $\{\mathbf{x}_i, y_i\}_{i=1}^m$, kernel matrix $K \in \mathbb{R}^{m\times m}$, weight matrix $\omega \in \mathbb{R}^{m\times m}$, regularization parameter $\lambda > 0$, step-size $\frac{1}{\tau} > 0$, initial guess $A^0 = \left[\boldsymbol{\alpha}_1^0, \boldsymbol{\alpha}_2^0, \ldots, \boldsymbol{\alpha}_m^0\right]$, with $\boldsymbol{\alpha}_i^0 \in \mathbb{R}^n$, $\sigma > 0$, $\epsilon > 0$

Output: the learned gradient coefficients $A^{t+1} = \left[\boldsymbol{\alpha}_1^{t+1}, \boldsymbol{\alpha}_2^{t+1}, \ldots, \boldsymbol{\alpha}_m^{t+1}\right]$.

while the stopping criterion $\|A^{t+1} - A^t\|_F \le \epsilon$ is not satisfied **do**

- Compute $Z_{ik} = y_i - y_k + (\mathbf{x}_k - \mathbf{x}_i)^T A K_k$, $1 \le i, k \le m$, where K_k is the kth column of K.
- Compute the gradient

$$\nabla\mathcal{E}(A^t) = \sum_{i,k=1}^{m} \omega_{ik} \exp\left(-Z_{ik}^2/\sigma^2\right) Z_{ik}(\mathbf{x}_k - \mathbf{x}_i)K_k^T.$$

- Compute the descent step

$$D^{t+1} = A^t - \frac{1}{\tau}\nabla\mathcal{E}(A^t).$$

- Perform the soft thresholding $S_{p,q}^{\lambda/\tau}$ on D^{t+1} to obtain A^{t+1}, where (p, q) is selected from $\{(1, 1), (2, 1), (2, 2)\}$

$$A^{t+1} = S_{p,q}^{\lambda/\tau}(D^{t+1}),$$

and set $t := t + 1$.

end while

Observing this, we can easily modify Algorithm 1 to solve the classification version of our proposed model (4).

B. Convergence Analysis of the Proposed Algorithm

This part is dedicated to proving the convergence of Algorithm 1. We first rewrite $\nabla\mathcal{E}$ as a more concise form. To this end, vec$(\cdot)$ is denoted as the vectorization operator over any matrix space $\mathbb{R}^{s\times t}$, which is given as

$$\text{vec}(B)_{(i-1)t+j} = B_{ij}, \quad 1 \le i \le s, \quad 1 \le j \le t, \quad B \in \mathbb{R}^{s\times t}.$$

We further define matrix $G \in \mathbb{R}^{m^2\times nm}$ with

$$G = \begin{pmatrix} \text{vec}\big((\mathbf{x}_1 - \mathbf{x}_1)K_1^T\big) \\ \text{vec}\big((\mathbf{x}_2 - \mathbf{x}_1)K_2^T\big) \\ \vdots \\ \text{vec}\big((\mathbf{x}_m - \mathbf{x}_1)K_2^T\big) \\ \text{vec}\big((\mathbf{x}_1 - \mathbf{x}_2)K_2^T\big) \\ \vdots \\ \text{vec}\big((\mathbf{x}_m - \mathbf{x}_m)K_m^T\big) \end{pmatrix}$$

and the diagonal matrix $\Lambda \in \mathbb{R}^{m^2\times m^2}$ by

$$\begin{aligned}\Lambda = \text{Diag}\big(&\omega_{11}\exp\big(-Z_{11}^2/\sigma^2\big),\ \omega_{12}\exp\big(-Z_{12}^2/\sigma^2\big), \ldots, \\ &\omega_{1m}\exp\big(-Z_{1m}^2/\sigma^2\big),\ \omega_{21}\exp\big(-Z_{21}^2/\sigma^2\big), \ldots, \\ &\omega_{2m}\exp\big(-Z_{2m}^2/\sigma^2\big), \ldots, \omega_{mm}\exp\big(-Z_{mm}^2/\sigma^2\big)\big).\end{aligned}$$

Based on the above notations, the vectorization of the gradient $\nabla\mathcal{E}(A)$ can be written as

$$\text{vec}(\nabla\mathcal{E}(A)) = G^T\Lambda\text{vec}(Z).$$

We then prove the Lipschitz continuity of the gradient of the empirical risk $\mathcal{E}$.

Proposition 1: The gradient of $\nabla\mathcal{E}$ is Lipschitz continuous, i.e., for any $A_1, A_2 \in \mathbb{R}^{n\times m}$

$$\|\nabla\mathcal{E}(A_1) - \nabla\mathcal{E}(A_2)\|_F \le L\|A_1 - A_2\|_F$$

where $L = \|G\|_2^2$ is the Lipschitz constant of $\nabla\mathcal{E}$, with $\|G\|_2$ being the spectral norm of the linear operator G.

Proof: Under notations defined previously, we have

$$\begin{aligned}&\|\nabla\mathcal{E}(A_1) - \nabla\mathcal{E}(A_2)\|_F \\ &\quad= \|\text{vec}(\nabla\mathcal{E}(A_1)) - \text{vec}(\nabla\mathcal{E}(A_2))\|_F \\ &\quad= \|G^T\Lambda_{A_1}\text{vec}(Z_{A_1}) - G^T\Lambda_{A_2}\text{vec}(Z_{A_2})\|_F \\ &\quad\le \|G\|_2\|\Lambda_{A_1}\text{vec}(Z_{A_1}) - \Lambda_{A_2}\text{vec}(Z_{A_2})\|_F.\end{aligned}$$

To derive the desired conclusion, it remains to show that

$$\|\Lambda_{A_1}\text{vec}(Z_{A_1}) - \Lambda_{A_2}\text{vec}(Z_{A_2})\|_F \le \|G\|_2\|A_1 - A_2\|_F.$$

Note that $\|\Lambda_{A_1}\text{vec}(Z_{A_1}) - \Lambda_{A_2}\text{vec}(Z_{A_2})\|_F^2$ equals

$$\begin{aligned}\sum_{i,k=1}^{m} \omega_{ik}^2\big(&\exp\big(-(Z_{A_1})_{ik}^2/\sigma^2\big)(Z_{A_1})_{ik} \\ &- \exp\big(-(Z_{A_2})_{ik}^2/\sigma^2\big)(Z_{A_2})_{ik}\big)^2.\end{aligned}$$

Following from the fact that for any $s_1, s_2 \in \mathbb{R}$ and $\sigma > 0$:

$$\big|\exp\big(-s_1^2/\sigma^2\big)s_1 - \exp\big(-s_2^2/\sigma^2\big)s_2\big| \le |s_1 - s_2|$$

we have

$$\begin{aligned}&\|\Lambda_{A_1}\text{vec}(Z_{A_1}) - \Lambda_{A_2}\text{vec}(Z_{A_2})\|_F^2 \\ &\qquad\le \sum_{i,k=1}^{m} \omega_{ik}^2((Z_{A_1})_{ik} - (Z_{A_2})_{ik})^2 \\ &\qquad\le \sum_{i,k=1}^{m} ((Z_{A_1})_{ik} - (Z_{A_2})_{ik})^2 \\ &\qquad= \|Z_{A_1} - Z_{A_2}\|_F^2.\end{aligned}$$

By reminding the definition of Z, we know that

$$\begin{aligned}\|Z_{A_1} - Z_{A_2}\|_F &= \|G\text{vec}(A_1) - G\text{vec}(A_2)\|_F \\ &\le \|G\|_2\|A_1 - A_2\|_F.\end{aligned}$$

As a result, we see that

$$\|\nabla\mathcal{E}(A_1) - \nabla\mathcal{E}(A_2)\|_F \le \|G\|_2^2\|A_1 - A_2\|_F.$$

This completes the proof. ■

Following from Proposition 1, we obtain the following conclusion, which states that the proposed algorithm converges to a stationary point of (4).

Theorem 1: Let $\{A^t\}$ be the sequence generated by Algorithm 1, and let the step-size $1/\tau$ in the algorithm satisfy $\tau > L$. Denote $f_\lambda(A) = \mathcal{E}(A) + \lambda\|A\|_{p,q}^q$. Then, $\{f_\lambda(A^t)\}$ is monotonically decreasing, and every limit of $\{A^t\}$ is a stationary point of (4).

Proof: We first define a surrogate function $W(A, A^t)$ as

$$\frac{\tau}{2}\|A - A^t\|_F^2 + \langle \nabla \mathcal{E}(A^t), A - A^t\rangle + \lambda \|A\|_{p,q}^q.$$

From the definition of A^{t+1}, it is easy to see that A^{t+1} minimizes $W(A, A^t)$. Therefore, we know that $W(A^{t+1}, A^t)$ has the following property:

$$\frac{\tau}{2}\|A^{t+1} - A^t\|_F^2 + \langle \nabla \mathcal{E}(A^t), A^{t+1} - A^t\rangle + \lambda \|A^{t+1}\|_{p,q}^q \leq W(A^t, A^t) = \lambda \|A^t\|_{p,q}^q. \quad (12)$$

On the other hand, it follows from Proposition 1 that:

$$\mathcal{E}(A^{t+1}) \leq \mathcal{E}(A^t) + \langle \nabla \mathcal{E}(A^t), A^{t+1} - A^t\rangle + \frac{L}{2}\|A^{t+1} - A^t\|_F^2.$$

This in connection with (12) implies

$$f_\lambda(A^{t+1}) \leq f_\lambda(A^t) - \frac{\tau - L}{2}\|A^{t+1} - A^t\|_F^2$$

which further shows that $\{f_\lambda\}$ is monotonically decreasing. Consequently, $\lim_{t\to\infty} f_\lambda(A^t)$ exists. Moreover, we also have

$$\lim_{t\to\infty} \|A^{t+1} - A^t\|_F = 0. \quad (13)$$

Observing that $f_\lambda(A)$ is coercive, i.e., $f_\lambda(A) \to \infty$ as $A \to \infty$, the existence of $\lim_{t\to\infty} f_\lambda(A^t)$ reveals the boundedness of $\{A^t\}$. Denote A^* as a stationary point of $\{A^t\}$ and suppose $A^{t_l} \to A^*$ as $l \to \infty$. We have

$$0 \in \tau(A^{t_l+1} - A^{t_l}) + \nabla \mathcal{E}(A^{t_l}) + \lambda \partial \|A^{t_l+1}\|_{p,q}^q. \quad (14)$$

Taking into account of (13) and passing the limit into (14), we come to the conclusion that

$$0 \in \nabla \mathcal{E}(A^*) + \lambda \partial \|A^*\|_{p,q}^q.$$

Thus, A^* is a stationary point and this completes the proof. ∎

C. Choosing the Tuning Parameters

Note that there are three tuning parameters in the RGL model: the parameter s that determines the weights ω_{ik}, the regularization parameter λ, and the scale parameter σ in the loss functions.

The regularization parameter specifies the smoothness of the learned gradient function. To choose λ in local learning schemes, leave-one-out cross validation is frequently adopted [47], [48]. In our context, for the regression setting, in accordance with [49], the regularization parameter can be chosen by minimizing the localized cross-validation error. The localized cross-validation error is defined as

$$CV(\lambda) = \frac{1}{m}\sum_{i,k=1}^{m} \omega_{ik}\ell_\sigma\big(y_i - y_k + \vec{g}_{\mathbf{z}\setminus i}^{\lambda}(\mathbf{x}_i)\cdot(\mathbf{x}_k - \mathbf{x}_i)\big)$$

where $\vec{g}_{\mathbf{z}\setminus i}^{\lambda}$ refers to the learned gradient function with the ith sample being removed from the training data $\mathbf{z}$. Similar cross-validation error can be defined in a classification setting.

For the choice of weights ω_{ik}, as mentioned earlier, we choose the Gaussian weights $\omega_{ik} = \exp\{-(\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_0^2)\}$ with some $\sigma_0 > 0$. A typical choice of σ_0 is the median of pairwise distance of the samples $\{\mathbf{x}_i\}_{i=1}^m$. This choice might not be optimal but usually provide acceptable results, as suggested in [1], [3], and [8].

For the choice of the tuning parameter σ in the loss function, we follow suggestions from [28] and set $\sigma = 2.9486\sigma_0$ to ensure 95% asymptotic efficiency on the standard normal distribution, where σ_0 refers to the standard deviation of the noise ϵ.

D. On the Generalization Ability of the Proposed Method

The generalization ability of a learning paradigm refers to its prediction ability on newly observed instance, the study of which plays an important role in designing and analyzing learning models and is also a central focus in statistical learning theory. The generalization ability of the GL paradigm (3) has been well analyzed using the language of the statistical learning theory [1], [4], [8]. In fact, the generalization ability of the RGL model can also be analyzed and is different from that of (3) due to the presence of the scale parameter in the loss function that controls the robustness.

To proceed, let us denote $\overrightarrow{g_0} = \arg\min_{\vec{g}\in\mathcal{F}} \mathcal{R}(\vec{g})$, where

$$\mathcal{R}(\vec{g}) = \mathbb{E}_{\mathbf{z}}\big\{\mathcal{E}_{\mathbf{z}}^{\rho}(\vec{g})\big\} + \lambda\Omega(\vec{g}) \quad \text{and} \quad \mathcal{F} := \cup_{\mathbf{z}\in\mathbb{Z}^m}\mathcal{H}_{\mathcal{K},\mathbf{z}}^n.$$

Then, the evaluation of the generalization error is reduced to measuring the deviation $\mathcal{R}(\overrightarrow{g_{\mathbf{z}}}) - \mathcal{R}(\overrightarrow{g_0})$. Bounding the quantity $\mathcal{R}(\overrightarrow{g_{\mathbf{z}}}) - \mathcal{R}(\overrightarrow{g_0})$ can be much involved due to the varying sample $\mathbf{z}$. Under the least squares criterion, some analysis can be found in [50]–[52]. Taking the regression problem for instance, concerning the quantitative relation between the least squares criterion and the employed loss function (5), recently [39] presents some analysis. According to [39], the convergence rates of the learning model induced by the ℓ_σ loss can be deduced by using [39, Lemma 7].

The second argument of this section concerns the role that the scale parameter σ in the loss function plays. Introducing the intermediate term $\mathcal{R}(\nabla f^\star)$, the generalization error can be decomposed as follows:

$$\mathcal{R}(\overrightarrow{g_{\mathbf{z}}}) - \mathcal{R}(\overrightarrow{g_0}) = (\mathcal{R}(\overrightarrow{g_{\mathbf{z}}}) - \mathcal{R}(\nabla f^\star)) + (\mathcal{R}(\nabla f^\star) - \mathcal{R}(\overrightarrow{g_0})). \quad (15)$$

The first term $\mathcal{R}(\overrightarrow{g_{\mathbf{z}}}) - \mathcal{R}(\nabla f^\star)$ reflects the approximation ability of the empirical target function $\overrightarrow{g_{\mathbf{z}}}$ to the true gradient function $\nabla f^\star$. The second term $\mathcal{R}(\nabla f^\star) - \mathcal{R}(\overrightarrow{g_0})$ is caused using a robust loss function instead of the least-squares criterion, which we term as the model error. Usually, the larger the model error is, the more robust the model (4) will be. However, the decomposition (15) tells us that the approximation ability of $\overrightarrow{g_{\mathbf{z}}}$ to $\nabla f^\star$ will be weakened at the same time. This phenomenon has been observed and well understood in robust statistics for linear models [28]. For nonlinear regression models, concerning the ℓ_σ loss, some discussions and analysis can be found in [39]. Taking the ℓ_σ loss or the Huber's loss ℓ_h, for example, the parameter σ or c in fact determines a tradeoff between the two terms, as suggested in [39]. That is, the enhancement of the robustness is at the sacrifice of the generalization ability of the RGL model. We will also demonstrate this point via numerical experiments in Section IV.

IV. Applications and Simulation Studies

The proposed RGL model (4) associated with different regularizers introduced above can be applied to many applications, such as nonlinear variable selection, supervised dimension reduction, and coordinate covariance estimation. In this section, we carry out simulation studies with the RGL model on both synthetic and real data sets in the nonlinear variable selection problem. All the numerical computations are conducted on an Intel i7-3770 CPU desktop computer with 16 GB of RAM. The supporting software is MATLAB R2013a. Our purpose here is to verify the effectiveness of the RGL model and give some empirical comparisons with other GL models, e.g., GL [1] and SGL [8].

A. Synthetic Data

In this numerical experiment, we apply our RGL model (4) to the nonlinear variable selection problems in regression with both the additive regression model and nonadditive regression model.

We do comparisons with two GL methods: GL and SGL. For the GL and SGL model, variables are selected by ranking

$$r_j = \frac{\|g_{\mathbf{z},j}\|_{\mathcal{K}}^2}{\sum_{l=1}^{n} \|g_{\mathbf{z},l}\|_{\mathcal{K}}^2}, \quad j = 1, \ldots, n.$$

For the RGL model, variables are selected by ranking

$$r'_j = \frac{\sum_{i=1}^{m} |\alpha_{i,j}|^2}{\sum_{l=1}^{n} \sum_{i=1}^{m} |\alpha_{i,l}|^2}, \quad j = 1, \ldots, n. \tag{16}$$

In this experiment, we use the type of $\ell_{2,2}$ regularizer. Moreover, to see the robustness of the RGL model (4), we also employ the Huber's loss in (4), which is termed as RGL-ℓ_h. The constant c in Huber's loss (6) is set to 1.345, as suggested in [28]. To make notationally distinguishable, we denote our RGL model associated with the ℓ_σ loss as RGL-ℓ_σ. For each regression model, we consider two situations of the variables: uncorrelated variables and correlated variables.

Additive Model: The data with size 100 are generated from the following additive model:

$$y = -2\tan(0.5x_1) + x_2 + x_3 + \exp(-x_4) + \epsilon. \tag{17}$$

Nonadditive Model: The data with size 100 are generated from the following nonadditive model:

$$y = \frac{x_1}{0.5 + (1.5 + x_2)^2} + \epsilon. \tag{18}$$

This model has been used in [27] to address the nonlinear variable selection problem.

For each model, we consider 10 input variables, i.e., we add 6 noise variables into the additive model and 8 noise variables into the nonadditive one. We consider both correlated and uncorrelated variables, i.e., the following two different situations for $\mathbf{x} = (x_1, x_2, \ldots, x_{10})$ will be implemented.

1) $\mathbf{x} \sim N_{10}(\mathbf{0}_{10}, \mathbf{I}_{10})$.
2) $\mathbf{x} \sim N_{10}(\mathbf{0}_{10}, \Sigma)$, where Σ denotes the covariance matrix with (i, j)th entry given by $\Sigma_{i,j} = 0.5^{|i-j|}$.

We also consider different cases on the noise and outliers. For instance, we do experiments in the presence or absence of noise or outliers. Two kinds of noise are used: Cauchy noise with the location parameter $a = 2$, scale parameter $b = 4$, scaled by 0.01 and Gaussian noise $\mathcal{N}(0, 1)$ scaled by 0.05. The outliers added in the experiment is drawn from the chi-square distribution with 4 DOF. In this experiment, we use the Gaussian kernel $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp\{-(\|\mathbf{x} - \mathbf{x}'\|^2/2\sigma_0^2)\}$, where the hyperparameter σ_0 is set to 2 empirically.

TABLE V
Computational Time Comparison Among the GL, SGL, RGL-ℓ_h, and RGL-ℓ_σ for the Additive Model (17) (in Seconds)

	Uncorrelated Variables					Correlated Variables				
	I	II	III	IV	V	I	II	III	IV	V
GL [1]	0.10	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
SGL [8]	4.94	5.14	6.03	5.26	6.12	5.74	5.46	6.03	5.75	5.84
RGL-ℓ_h	7.50	7.75	8.34	7.68	8.76	7.80	7.79	8.34	7.66	8.17
RGL-ℓ_σ	5.67	5.93	7.63	6.08	8.46	5.63	5.71	7.63	5.72	7.52

For both the additive model and nonadditive model, we carry out the variable selection experiments with the following five different circumstances.

1) I: no noise, no outliers.
2) II: Gaussian noise, no outliers.
3) III: no noise, 30% outliers.
4) IV: Cauchy noise, no outliers.
5) V: Cauchy noise, 30% outliers.

Variable selection results for different circumstances are reported in Tables VI and VII. Table VIII empirically shows the influence of different regularization parameters λ and sample sizes, m, on the variable selection results. The experiments are implemented with the additive model (17). The influence is measured by the average selection percentage (ASP), which is defined as the average selection probability of the first four variables in 100 repeats. For the additive model, the averaged computational time of each method is also recorded and reported in Table V in 100 repeats. To show the influence of the scale parameter σ in the loss function, we also report numerical results in Figs. 5 and 6 by applying the RGL method to the nonadditive model (18), which is carried out when case V is assumed and the variables are correlated. In Figs. 5 and 6, the accuracy is equal to the averaged selected times in 100 repeats and 50 different σ-values are chosen by using the logspace function in MATLAB.

The above-reported results tell us the following.

1) When there are no noise and outliers, all the three GL models, RGL, GL, and SGL, give satisfactory selection results. Moreover, when the variables are correlated, RGL gives slightly better selection results.
2) Comparing with the GL and SGL model, our RGL model gives better results for the cases with Gaussian noise or Cauchy noise, especially for the nonadditive model.
3) With the presence of outliers, the RGL model can always outperform the GL and SGL model in selecting variables. Moreover, experimental results show that the RGL model associated with the ℓ_σ loss can outperform the RGL model associated with the Huber's loss.

TABLE VI
VARIABLE SELECTION RESULTS FOR THE ADDITIVE MODEL (17) FOR DIFFERENT CASES

Case	Models	Uncorrelated Variables										Correlated Variables									
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
I	GL [1]	100	92	91	97	3	3	3	5	4	2	100	85	90	99	5	5	5	5	3	3
	SGL [8]	100	94	93	99	3	2	3	3	2	1	98	90	87	98	12	2	4	4	3	2
	RGL-ℓ_h	100	100	100	100	0	0	0	0	0	0	100	98	97	99	6	0	0	0	0	0
	RGL-ℓ_σ	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**
II	GL [1]	99	96	94	96	3	2	0	2	3	5	100	90	89	94	4	5	7	3	4	4
	SGL [8]	99	94	97	96	4	2	1	1	3	3	99	87	91	96	11	4	4	4	3	1
	RGL-ℓ_h	100	100	100	100	0	0	0	0	0	0	100	96	98	100	6	0	0	0	0	0
	RGL-ℓ_σ	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**
III	GL [1]	38	42	29	34	46	42	53	40	38	38	39	37	35	39	38	46	51	36	37	42
	SGL [8]	45	41	40	38	42	40	49	35	30	40	48	43	35	27	34	46	37	42	41	47
	RGL-ℓ_h	88	74	80	90	12	14	10	11	8	13	72	70	68	79	24	23	17	7	16	24
	RGL-ℓ_σ	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**	**98**	**98**	**99**	**90**	**5**	**2**	**4**	**2**	**0**	**2**
IV	GL [1]	97	86	85	97	7	6	8	2	6	6	97	88	80	95	9	6	9	5	6	5
	SGL [8]	94	91	91	100	7	3	3	3	3	5	93	85	85	96	18	8	3	4	5	3
	RGL-ℓ_h	100	100	100	100	0	0	0	0	0	0	99	96	97	100	6	1	0	0	1	0
	RGL-ℓ_σ	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**	**100**	**100**	**100**	**100**	**0**	**0**	**0**	**0**	**0**	**0**
V	GL [1]	50	37	45	32	38	41	47	37	42	31	43	31	38	46	48	43	37	38	40	36
	SGL [8]	40	43	41	35	39	40	48	37	41	36	55	34	38	44	39	43	36	28	37	46
	RGL-ℓ_h	87	76	76	85	13	8	23	9	12	11	80	61	66	75	27	24	16	14	15	22
	RGL-ℓ_σ	**100**	**99**	**99**	**96**	**0**	**0**	**2**	**2**	**1**	**1**	**97**	**95**	**99**	**90**	**9**	**1**	**2**	**0**	**4**	**3**

TABLE VII
VARIABLE SELECTION RESULTS FOR THE NONADDITIVE MODEL (18) FOR DIFFERENT CASES

Case	Models	Uncorrelated Variables										Correlated Variables									
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
I	GL [1]	100	58	2	4	8	5	7	3	6	7	100	72	3	3	1	2	4	7	6	2
	SGL [8]	100	59	1	11	5	6	3	6	3	6	100	94	3	0	1	0	0	2	0	0
	RGL-ℓ_h	100	76	1	5	6	3	3	2	2	2	100	95	3	0	1	0	0	1	0	0
	RGL-ℓ_σ	**100**	**97**	**1**	**2**	**0**	**0**	**0**	**0**	**0**	**0**	**100**	**98**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
II	GL [1]	100	54	4	4	4	6	6	7	8	7	100	66	5	4	2	4	6	4	4	5
	SGL [8]	100	48	10	10	6	4	5	5	4	8	100	98	1	0	0	1	0	0	0	0
	RGL-ℓ_h	100	78	5	4	3	2	1	4	2	1	100	99	1	0	0	0	0	0	0	0
	RGL-ℓ_σ	**100**	**98**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**100**	**99**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
III	GL [1]	23	27	21	20	26	14	18	14	17	20	16	18	23	23	22	27	24	20	16	11
	SGL [8]	16	20	27	23	27	18	21	17	14	17	20	19	25	22	16	24	18	16	16	24
	RGL-ℓ_h	94	21	11	13	14	11	11	10	9	6	93	40	16	4	7	12	10	4	6	8
	RGL-ℓ_σ	**100**	**84**	**1**	**1**	**1**	**2**	**2**	**4**	**1**	**4**	**100**	**99**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
IV	GL [1]	94	26	8	12	6	10	11	11	11	11	90	40	7	7	9	6	11	13	10	7
	SGL [8]	95	44	5	7	7	11	11	6	11	3	97	71	4	2	3	4	5	7	4	3
	RGL-ℓ_h	100	73	**1**	5	2	2	6	2	6	3	100	95	1	0	0	0	0	0	3	1
	RGL-ℓ_σ	**100**	**92**	2	**2**	**0**	**1**	**2**	**0**	**1**	**0**	**100**	**97**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**0**
V	GL [1]	19	16	16	16	25	23	26	26	16	17	15	22	22	20	25	21	25	20	12	18
	SGL [8]	16	16	17	19	29	21	17	25	19	21	25	24	24	17	18	16	21	21	18	16
	RGL-ℓ_h	89	20	7	7	19	18	10	16	5	9	89	39	10	7	12	9	7	9	9	9
	RGL-ℓ_σ	**100**	**84**	**2**	**2**	**1**	**3**	**3**	**3**	**2**	**0**	**100**	**95**	**2**	**1**	**0**	**0**	**0**	**1**	**1**	**0**

4) Concerning the influence of the regularization parameters on the selection results, we find that satisfactory selection results can be obtained when $\lambda \in [10^{-8}, 10^{-2}]$. This indicates that when the RGL model is applied to the nonlinear variable selection problem, selection results might be not very sensitive to the regularization parameter. According to Table VIII, we find that satisfactory selection results can be obtained with a relatively small size samples with respect to the dimension of features. In our experiments, we also find that the selection results are also not very sensitive to the choice of the kernel. For example, we find that the linear kernel does not work well in selecting the variables. Moreover, we also note that both polynomial kernel and RBF kernel work well and are not very sensitive to the hyperparameters, e.g., the offset and the order of the polynomial kernel, the parameter σ_0 in the RBF kernel. Due to the page limitation, more experimental results are left to the supplemental materials.

TABLE VIII
INFLUENCE OF VARYING SAMPLE SIZE AND REGULARIZATION PARAMETER ON THE ASP

λ	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	10^2	10^3	10^4	10^5	10^6
ASP(%)	100.00	100.00	100.00	100.00	96.75	80.75	79.75	79.00	79.50	80.00	79.25	78.50	79.00
m	10	20	30	40	50	60	70	80	90	100	150	200	400
ASP(%)	73.00	95.00	99.25	99.25	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00

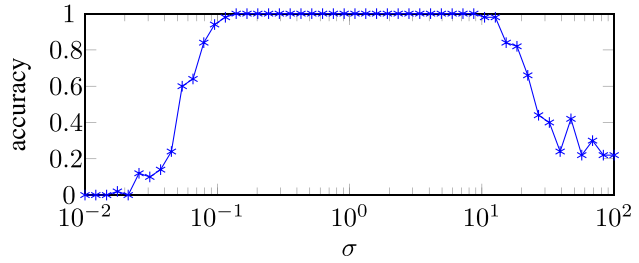


Fig. 5. Selection accuracy of the variable x_1 by applying the RGL method to the nonadditive model (18) for different σ values. x-axis stands for different σ values and y-axis represents the selection accuracy.

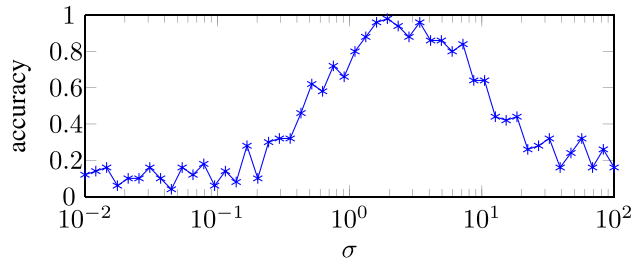


Fig. 6. Selection accuracy of the variable x_2 by applying the RGL method to the nonadditive model (18) for different σ values. x-axis stands for different σ values and y-axis represents the selection accuracy.

5) Concerning the computational time, from Table V, we see that the proposed RGL method takes longer time than GL. Given that solving GL is equivalent to solving a linear equation system, it is reasonable that GL is faster than other methods. Moreover, we also see that the efficiency of RGL is comparable with that of SGL. However, numerical results reported above show that RGL is recommended in variable selection when the computational time is not an important concern.
6) Figs. 5 and 6 show the influence of the scale parameter in the loss function. More explicitly, with smaller σ values, the selection accuracy which can be in a sense referred as the generalization ability of the RGL model is weakened. When the σ value increases, the selection accuracy is also improved. However, when the σ value is larger than certain value, the RGL model cannot be resistant to outliers. This coincides with our previous discussions that the enhancement of the robustness is at the cost of the generalization ability of the proposed RGL method, as discussed in Section III-D.

B. Real Data

We first apply the RGL model to the nonlinear variable selection problem with two real data sets, leukemia data and prostate cancer data. Both data sets are publicly available at: http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi. The leukemia data set consists of 72 samples with 7129 gene expression values. Coming from a study of gene expression in two types of acute leukemias [53], acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML), the data set has been frequently used in real data experiments in the variable selection problems. The purpose of using this data set is to select corresponding genes and then classify patients into the above two types of leukemia: ALL and AML. The data set is split into 38 training samples and 34 test samples. The prostate cancer data set [54] consists of 102 samples in the training data with 52 tumor samples and 50 normal samples. Thirty-four independent test samples are available from a different experiment. For each sample, there are 12 600 genes.

TABLE IX
EXPERIMENTAL RESULTS WITH THE LEUKEMIA DATA

Features (s)	50	106	300	500	700	1000	3000
Test error of GL [1]	1	0	1	1	1	1	1
Test error of SGL [8]	1	0	1	1	1	1	1
Test error of RGL	1	0	1	1	1	1	1

In our experiment, we apply our RGL model associated with the ϕ_σ loss and $\ell_{2,1}$-regularizer to the two data sets. We rank the genes according to the norm of their gradient vector as done in the previous simulated data. We then select s features. Both the training samples and the test samples are then projected into corresponding dimensions. Based on the selected genes, we then do classification by using a linear Support Vector Machine (SVM) classifier. More explicitly, we implement our experiments with the following steps.

1) Using the training data and test data, we apply our RGL model to learn the gradient and then rank the features according to the ranking rule (16).
2) Suppose that s features are needed to be selected. We project the training data and test data into corresponding dimensions according to the rank of the features.
3) We do classification and compute the test error with truncated training data and test data.
4) For each given s, we repeat the above two steps.

Experimental results are reported in Tables IX and X, where, in Table IX, the test error is given in terms of the number of misclassified instances on test set. According to the reported results, we can see that we get comparable classification accuracy for the three GL models.

Next, we continue our numerical experiments with the RGL model using two data sets, Statlog (German Credit Data) and Sonar collected from UCI [55]. The Sonar data contains

TABLE X
EXPERIMENTAL RESULTS WITH THE PROSTATE CANCER DATA

Features (s)	50	100	300	500	650	1000	3000
Test error of GL [1]	7	8	5	1	1	1	9
Test error of SGL [8]	8	8	3	1	1	1	7
Test error of RGL	7	8	5	1	1	1	9

TABLE XI
AVERAGE CLASSIFICATION ACCURACY (%) OF THE SONAR DATA

Features(s)	5	10	20	30	40	50	60
GL [1]	52.90	54.80	64.50	65.55	74.80	68.90	67.45
SGL [8]	52.90	54.80	64.50	65.55	74.80	68.90	67.45
RGL	**53.55**	**55.10**	**64.80**	**65.95**	**75.10**	**68.95**	67.45

TABLE XII
AVERAGE CLASSIFICATION ACCURACY (%) OF THE STATLOG DATA

Features(s)	4	7	10	13	16	19	22
GL [1]	66.05	89.10	88.55	86.15	85.50	**85.25**	83.10
SGL [8]	66.05	89.10	88.55	86.15	85.50	**85.25**	83.10
RGL	**68.50**	**89.35**	**89.05**	**86.25**	**85.80**	84.55	**83.20**

208 samples in two classes. Each sample consists of 60 features. The Statlog data set consists of 1000 samples in two classes with 20 features in each sample.

In our experiment, for the two data sets, we randomly choose 20 samples to learn the gradients. After feature selection, again we employ the linear SVM classifier to the above chosen samples, using fivefold cross-validation. We repeat the experiment with each data set for 100 times and record the average classification accuracy with different sizes of selected features. Experimental results are reported in Tables XI and XII. We find that the RGL model gives slightly better selection results.

In summary, our experiments on both simulated data and real data reveal the effectiveness of the proposed RGL model (4). More concretely, when the data set is contaminated by outliers or heavy-tailed noise, the RGL model outperforms other GL models. The performance of the RGL is also comparable even if there are no outliers or heavy-tailed noise. However, in real-world applications, we might not be able to know in which situation we are. In this sense, the RGL model (4) can be a better and safe choice for GL.

V. CONCLUSION

In this paper, we proposed an RGL model in both regression and classification problems. The robustness was achieved by introducing robust loss functions. Instead of learning over an RKHS, the RGL model learned over an instance-based kernelized dictionary. The kernelized dictionary was induced by a kernel, which was not restricted to be positive definite. The proposed RGL model was then applied to the nonlinear variable selection problem. Experimental comparisons with models based on the Huber's loss and other GL models were presented. The efficiency of our proposed model was validated on both synthetic and real data sets.

It would also be interesting to investigate an extended model as explored in [2]. The modified model allows us to learn the gradient of the regression function, while the regression function can also be learned at the same time. The combined model can be formulated as

$$(f_{\mathbf{z}}, \overrightarrow{g_{\mathbf{z}}}) = \arg \min_{(f,\vec{g})\in\mathcal{H}_{\mathcal{K},\mathbf{z}}^{n+1}} \left\{\mathcal{E}_{\mathbf{z}}^{\rho}(f,\vec{g}) + \lambda\Omega(f,\vec{g})\right\} \quad (19)$$

where $\mathcal{E}_{\mathbf{z}}^{\rho}(f,\vec{g})$ equals

$$\frac{1}{m^2}\sum_{i,k=1}^{m} \omega_{ik}\rho\big((y_i - f(\mathbf{x}_k)) + \vec{g}(\mathbf{x}_i)\cdot(\mathbf{x}_k - \mathbf{x}_i)\big)$$

and $\Omega(f,\vec{g})$ refers to the regularization term. The difference between (4) and (19) lies in that the observation value y_k in (4) is replaced by the estimated value $f(\mathbf{x}_k)$. The tuple $(f_{\mathbf{z}}, \overrightarrow{g_{\mathbf{z}}})$ is expected to be an efficient estimator of $(f^\star, \nabla f^\star)$. Further, theoretical and empirical investigations will be our future work.

REFERENCES

[1] S. Mukherjee and D.-X. Zhou, "Learning coordinate covariances via gradients," *J. Mach. Learn. Res.*, vol. 7, pp. 519–549, Mar. 2006.

[2] Q. Wu, J. Guinney, M. Maggioni, and S. Mukherjee, "Learning gradients: Predictive models that infer geometry and statistical dependence," *J. Mach. Learn. Res.*, vol. 11, pp. 2175–2198, Aug. 2010.

[3] S. Mukherjee, Q. Wu, and D.-X. Zhou, "Learning gradients on manifolds," *Bernoulli*, vol. 16, no. 1, pp. 181–207, 2010.

[4] S. Mukherjee and Q. Wu, "Estimation of gradients and coordinate covariation in classification," *J. Mach. Learn. Res.*, vol. 7, pp. 2481–2514, Nov. 2006.

[5] C. Campbell and Y. Ying, "Learning coordinate gradients with multi-task kernels," in *Proc. 21st Annu. Conf. Learn. Theory*, Helsinki, Finland, Jul. 2008, pp. 217–228.

[6] J. Guinney, Q. Wu, and S. Mukherjee, "Estimating variable structure and dependence in multitask learning via gradients," *Mach. Learn.*, vol. 83, no. 3, pp. 265–287, 2011.

[7] Y. Ying, Q. Wu, and C. Campbell, "Learning the coordinate gradients," *Adv. Comput. Math.*, vol. 37, no. 3, pp. 355–378, 2012.

[8] G.-B. Ye and X. Xie, "Learning sparse gradients for variable selection and dimension reduction," *Mach. Learn.*, vol. 87, no. 3, pp. 303–355, 2012.

[9] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[10] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc., B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[11] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *J. Amer. Statist. Assoc.*, vol. 96, no. 456, pp. 1348–1360, 2001.

[12] H. Zou, "The adaptive lasso and its oracle properties," *J. Amer. Statist. Assoc.*, vol. 101, no. 476, pp. 1418–1429, 2006.

[13] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Roy. Statist. Soc., B (Statist. Methodol.)*, vol. 67, no. 2, pp. 301–320, 2005.

[14] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Roy. Statist. Soc., B (Statist. Methodol.)*, vol. 68, no. 1, pp. 49–67, 2006.

[15] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *Ann. Statist.*, vol. 38, no. 2, pp. 894–942, 2010.

[16] H. Wang, G. Li, and G. Jiang, "Robust regression shrinkage and consistent variable selection through the LAD-lasso," *J. Bus. Econ. Statist.*, vol. 25, no. 3, pp. 347–355, 2007.

[17] S. Lambert-Lacroix and L. Zwald, "Robust regression through the Huber's criterion and adaptive lasso penalty," *Electron. J. Statist.*, vol. 5, pp. 1015–1053, 2011.

[18] X. Wang, Y. Jiang, M. Huang, and H. Zhang, "Robust variable selection with exponential squared loss," *J. Amer. Statist. Assoc.*, vol. 108, no. 502, pp. 632–643, 2013.

[19] H. Liu, L. Wasserman, J. D. Lafferty, and P. K. Ravikumar, "SpAM: Sparse additive models," in *Proc. 21st Annu. Conf. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2007, pp. 1201–1208.

[20] F. Ojeda, J. A. K. Suykens, and B. De Moor, "Low rank updated LS-SVM classifiers for fast variable selection," *Neural Netw.*, vol. 21, nos. 2–3, pp. 437–449, 2008.

[21] F. Bach. (2009). "High-dimensional non-linear variable selection through hierarchical kernel learning." [Online]. Available: http://arxiv.org/abs/0909.0844

[22] L. Rosasco, M. Santoro, S. Mosci, A. Verri, and S. Villa, "A regularization approach to nonlinear variable selection," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, Sardinia, Italy, May 2010, pp. 653–660.

[23] L. Rosasco, S. Villa, S. Mosci, M. Santoro, and A. Verri, "Nonparametric sparsity and regularization," *J. Mach. Learn. Res.*, vol. 14, pp. 1665–1714, Jul. 2013.

[24] M. Yamada, W. Jitkrittum, L. Sigal, E. P. Xing, and M. Sugiyama, "High-dimensional feature selection by feature-wise kernelized lasso," *Neural Comput.*, vol. 26, no. 1, pp. 185–207, 2014.

[25] Y. Xia, H. Tong, W. K. Li, and L.-X. Zhu, "An adaptive estimation of dimension reduction space," *J. Roy. Statist. Soc., B (Statist. Methodol.)*, vol. 64, no. 3, pp. 363–410, 2002.

[26] Q. Wang and X. Yin, "A nonlinear multi-dimensional variable selection method for high dimensional data: Sparse MAVE," *Comput. Statist. Data Anal.*, vol. 52, no. 9, pp. 4512–4520, 2008.

[27] W. Yao and Q. Wang, "Robust variable selection through MAVE," *Comput. Statist. Data Anal.*, vol. 63, pp. 42–49, Jul. 2013.

[28] P. J. Huber and E. M. Ronchetti, *Robust Statistics*. New York, NY, USA: Wiley, 2009.

[29] J. A. K. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: Robustness and sparse approximation," *Neurocomputing*, vol. 48, nos. 1–4, pp. 85–105, 2002.

[30] K. De Brabanter *et al.*, "Robustness of kernel based regression: A comparison of iterative weighting schemes," in *Proc. 19th Conf. Artif. Neural Netw.*, Limassol, Cyprus, Sep. 2009, pp. 100–110.

[31] V. Roth, "The generalized LASSO," *IEEE Trans. Neural Netw.*, vol. 15, no. 1, pp. 16–28, Jan. 2004.

[32] R. Luss and A. d'Aspremont, "Support vector machine classification with indefinite kernels," *Math. Program. Comput.*, vol. 1, nos. 2–3, pp. 97–118, 2009.

[33] Y. Ying, C. Campbell, and M. Girolami, "Analysis of SVM with indefinite kernels," in *Proc. 23rd Annu. Conf. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2009, pp. 2205–2213.

[34] Y. Wu and Y. Liu, "Robust truncated hinge loss support vector machines," *J. Amer. Statist. Assoc.*, vol. 102, no. 479, pp. 974–983, 2007.

[35] S. Y. Park and Y. Liu, "Robust penalized logistic regression with truncated loss functions," *Can. J. Statist.*, vol. 39, no. 2, pp. 300–323, 2011.

[36] P. W. Holland and R. E. Welsch, "Robust regression using iteratively reweighted least-squares," *Commun. Statist.-Theory Methods*, vol. 6, no. 9, pp. 813–827, 1977.

[37] J. E. Dennis, Jr., and R. E. Welsch, "Techniques for nonlinear least squares and robust regression," *Commun. Statist.-Simul. Comput.*, vol. 7, no. 4, pp. 345–359, 1978.

[38] W. Liu, P. P. Pokharel, and J. C. Principe, "Correntropy: Properties and applications in non-Gaussian signal processing," *IEEE Trans. Signal Process.*, vol. 55, no. 11, pp. 5286–5298, Nov. 2007.

[39] Y. Feng, X. Huang, L. Shi, Y. Yang, and J. A. K. Suykens, "Learning with the maximum correntropy criterion induced losses for regression," *J. Mach. Learn. Res.*, to be published.

[40] Y. Lin, "A note on margin-based loss functions in classification," *Statist. Probab. Lett.*, vol. 68, no. 1, pp. 73–82, 2004.

[41] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *Mach. Learn.*, vol. 73, no. 3, pp. 243–272, 2008.

[42] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, Mar. 2010.

[43] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[44] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, Eds. New York, NY, USA: Springer-Verlag, 2011.

[45] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Model. Simul.*, vol. 4, no. 4, pp. 1168–1200, 2005.

[46] M. Kowalski, "Sparse regression using mixed norms," *Appl. Comput. Harmon. Anal.*, vol. 27, no. 3, pp. 303–324, 2009.

[47] D. M. Allen, "The relationship between variable selection and data agumentation and a method for prediction," *Technometrics*, vol. 16, no. 1, pp. 125–127, 1974.

[48] M. Wand, "Local regression and likelihood," *J. Amer. Statist. Assoc.*, vol. 96, no. 453, pp. 339–355, 2001.

[49] Q. Wu and S. Mukherjee, "Variable selection and dimension reduction by learning gradients," unpublished.

[50] Q. Wu and D.-X. Zhou, "Learning with sample dependent hypothesis spaces," *Comput. Math. Appl.*, vol. 56, no. 11, pp. 2896–2907, 2008.

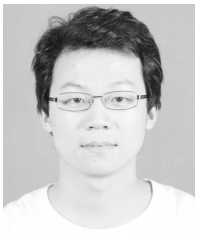
[51] Q.-W. Xiao and D.-X. Zhou, "Learning by nonsymmetric kernels with data dependent spaces and ℓ^1-regularizer," *Taiwanese J. Math.*, vol. 14, no. 5, pp. 1821–1836, 2010.

[52] L. Shi, Y.-L. Feng, and D.-X. Zhou, "Concentration estimates for learning with ℓ^1-regularizer and data dependent hypothesis spaces," *Appl. Comput. Harmon. Anal.*, vol. 31, no. 2, pp. 286–302, 2011.

[53] T. R. Golub *et al.*, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[54] D. Singh *et al.*, "Gene expression correlates of clinical prostate cancer behavior," *Cancer Cell*, vol. 1, no. 2, pp. 203–209, 2002.

[55] M. Lichman. (2013). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml



Yunlong Feng was born in Anhui, China, in 1986. He received the bachelor's degree in mathematics from Nankai University, Tianjin, China, in 2007, and the Ph.D. degree in applied mathematics and computational mathematics from the City University of Hong Kong, Hong Kong, and the University of Science and Technology of China, Hefei, China, in 2012.

He was a Post-Doctoral Fellow with the Department of Mathematics, City University of Hong Kong, from 2012 to 2013. Since 2013, he has been a Post-Doctoral Researcher with the Department of Electrical Engineering, Stadius Centre for Dynamical Systems, Signal Processing and Data Analytics, Katholieke Universiteit Leuven, Leuven, Belgium, with Prof. J. A. K. Suykens. His current research interests include statistical machine learning, learning theory, and nonparametric statistics.



Yuning Yang was born in China in 1984. He received the B.S., M.S., and Ph.D. degrees in computational mathematics from Nankai University, Tianjin, China, in 2007, 2010, and 2013, respectively.

He has been a Post-Doctoral Researcher with the Department of Electrical Engineering, Stadius Centre for Dynamical Systems, Signal Processing and Data Analytics, Katholieke Universiteit Leuven, Leuven, Belgium, with Prof. J. A. K. Suykens, since 2013. His current research interests include optimization, machine learning, and multilinear algebra.



Johan A. K. Suykens (F'15) was born in Willebroek, Belgium, in 1966. He received the master's degree in electromechanical engineering and the Ph.D. degree in applied sciences from the Katholieke Universiteit Leuven (KU Leuven), Leuven, Belgium, in 1989 and 1995, respectively.

He was a Visiting Post-Doctoral Researcher with the University of California at Berkeley, Berkeley, CA, USA, in 1996. He has been a Post-Doctoral Researcher with the Fund for Scientific Research, Flanders, Belgium, and is currently a Professor (Hoogleraar) with KU Leuven. He has authored the books entitled *Artificial Neural Networks for Modeling and Control of Nonlinear Systems* (Kluwer Academic Publishers) and *Least Squares Support Vector Machines* (World Scientific), co-authored a book entitled *Cellular Neural Networks, Multiscroll Chaos and Synchronization* (World Scientific), and edited the books entitled *Nonlinear Modeling: Advanced Black-Box Techniques* (Kluwer Academic Publishers), *Advances in Learning Theory: Methods, Models and Applications* (IOS Press), and *Regularization, Optimization, Kernels, and Support Vector Machines* (Chapman & Hall/CRC).

Prof. Suykens was a recipient of the IEEE Signal Processing Society Best Paper Award in 1999, and several best paper awards at international conferences. He was also a recipient of the International Neural Networks Society Young Investigator Award for his significant contributions in the field of neural networks in 2000. He received the ERC Advanced Grant in 2011. In 1998, he organized the International Workshop on Nonlinear Modeling with Time-Series Prediction Competition. He served as an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS from 1997 to 1999 and 2004 to 2007, and the IEEE TRANSACTIONS ON NEURAL NETWORKS from 1998 to 2009. He served as the Director and an Organizer of the NATO Advanced Study Institute on Learning Theory and Practice, Leuven, in 2002, the Program Co-Chair of the International Joint Conference on Neural Networks in 2004, and the International Symposium on Nonlinear Theory and its Applications in 2005, an Organizer of the International Symposium on Synchronization in Complex Networks in 2007, a Co-Organizer of the NIPS Workshop on Tensors, Kernels and Machine Learning in 2010, and the Chair of ROKS in 2013.