Correspondence

Robust Regularized Kernel Regression

Jianke Zhu, *Student Member, IEEE*, Steven C. H. Hoi, *Member, IEEE*, and Michael Rung-Tsong Lyu, *Fellow, IEEE*

***Abstract*—Robust regression techniques are critical to fitting data with noise in real-world applications. Most previous work of robust kernel regression is usually formulated into a dual form, which is then solved by some quadratic program solver consequently. In this correspondence, we propose a new formulation for robust regularized kernel regression under the theoretical framework of regularization networks and then tackle the optimization problem directly in the primal. We show that the primal and dual approaches are equivalent to achieving similar regression performance, but the primal formulation is more efficient and easier to be implemented than the dual one. Different from previous work, our approach also optimizes the bias term. In addition, we show that the proposed solution can be easily extended to other noise-reliable loss function, including the Huber-ϵ insensitive loss function. Finally, we conduct a set of experiments on both artificial and real data sets, in which promising results show that the proposed method is effective and more efficient than traditional approaches.**

***Index Terms*—Kernel regression, regularized least squares (RLS), robust estimator, support vector machine (SVM).**

I. Introduction

Learning to fit data with noise is an important research problem in many real-world data mining applications. Robust regression has attracted more and more research attention recently [1]–[5]. The history of robust regression research can be traced back to the early works of scientists in both statistics and mathematics [6]. Recently, a variety of techniques have been proposed to solve the robust regression problem in various real-world applications. One promising group of robust regression techniques is based on the kernel learning techniques, which is motivated by the regularization network theory [7], [8]. Several popular techniques, such as the regularized least squares (RLS) and support vector regression (SVR), are developed under similar theoretical foundation [8]–[10].

In general, given a real-world regression problem, noise is an inevitable challenge, which needs to be carefully handled. To tackle this problem with kernel learning methods, there are two typical ways to attack this challenge. One way is to add some regularization term into the regression optimization, which can avoid the regression overfitting of the data. The other way is to design a proper loss function that is able to tolerate the noisy outliers. For the former way, it is well known that regularization plays a critical role in kernel methods for achieving better generalization performance. The latter way, which is less carefully considered, is also important to tackle the noise problem. For example, traditional kernel-based regression methods often adopt the quadratic loss function, which may receive large impact from outliers; hence, the resulting solution is likely to be dominated by a small amount of noise. In this correspondence, we propose a novel robust regularized kernel regression, which carefully considers both factors in tackling the noise issue.

In addition to noise, another important issue for kernel-based regression methods is the efficiency problem. Typically, most kernel methods, such as support vector machines (SVMs), are often solved based on the dual optimization [8], [10]. Whereas the dual optimization can be solved by some state-of-the-art techniques, such as the sequential minimal optimization [11], some recent studies indicate that the primal formulation of SVMs can also be solved efficiently without using the complicated dual optimization techniques [12]–[14]. The primal approaches usually only involve simple linear equations that can be solved efficiently. The previous study also shows that the primal approaches are usually superior to the dual approaches when considering approximate solutions. However, there is still little attention on solving the robust kernel regression problem based on the primal approach directly. In this correspondence, we propose a novel and efficient algorithm to solve the robust regression problem in primal directly.

To this end, we highlight three of our major contributions: 1) We propose a new robust regression technique, which is the *robust regularized kernel regression*, which is more robust for learning to fit with noisy data than traditional techniques; 2) we study optimization techniques for solving the problem from both dual and primal approaches, and 3) we show that the proposed primal approach is more efficient than the conventional dual approaches and conduct empirical experiments to verify its effectiveness.

The rest of this correspondence is organized as follows. Section II reviews the related work of robust kernel regression. Section III gives the formulation of the robust regularized kernel regression technique. Section IV investigates optimization techniques for solving the robust regression problem from both dual and primal approaches. Section V shows our empirical study on evaluating the accuracy and efficiency performance of the proposed regression techniques. Section VI concludes this correspondence.

II. Related Work

We first review related work on robust kernel regression, followed by the motivation of solving it directly in primal.

To handle the noise problem in regressions, some approaches have been studied by hybrid loss functions, including the squared loss function and the ε-insensitive loss function. The robust Huber regression problem [15] considers a convex differentiable cost function, which is quadratic for small errors and linear otherwise. Hence, the Huber loss function is insensitive to outliers. In [4], the robust Huber regression was formulated into a dual form and solved efficiently by a convex quadratic program. In addition, a unified loss function was proposed in [1], which tries to combine the power of both the Huber and the ε-insensitive loss functions. Another hybrid approach also attempts to introduce the smoothness into the ε-insensitive loss

Manuscript received September 7, 2007; revised January 19, 2008 and May 6, 2008. First published September 16, 2008; current version published November 20, 2008. This work was supported in part by the Innovation and Technology Fund under Grant ITS/084/07, by the Research Grants Council under Earmarked Grant CUHK4150/07E, and by the Singapore NTU AcRF Tier-1 under Research Grant RG67/07. This paper was recommended by Associate Editor H. Qiao.

J. Zhu and M. R. Lyu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: jkzhu@cse.cuhk.edu.hk; lyu@cse.cuhk.edu.hk).

S. C. H. Hoi is with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: chhoi@ntu.edu.sg).

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/TSMCB.2008.927279

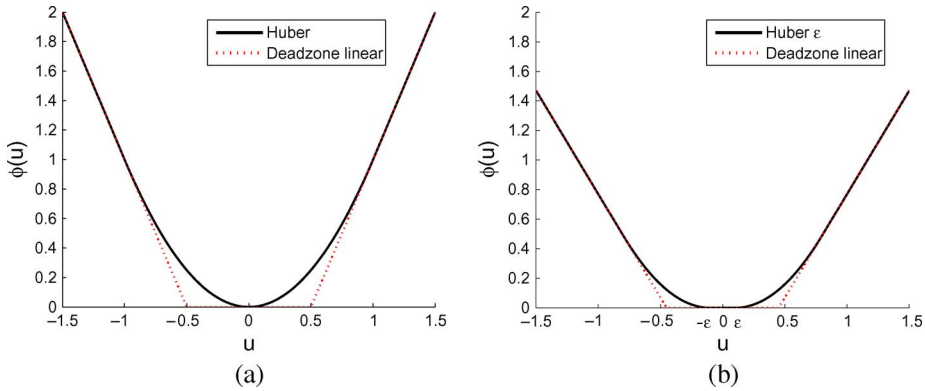

Fig. 1. Illustration of two loss functions.

function [3]. Although training efficiency can be improved by a fast Newton–Armijo algorithm, their objective function is still based on the squared loss function. In addition, L_1 regularized logistic regression [2] also demonstrates some good performance for classification tasks.

Aside from the robust loss functions, some previous studies also attempt to improve the robustness of large margin training to handle outliers in SVMs. A robust SVR is proposed to tackle the overfitting issue and fine-tune the parameters, which attempts to avoid selecting the outliers as support vectors [16]. In [17], some indicator variables are introduced into soft margin SVMs to detect and remove outliers selectively. Although the relaxation of the soft margin SVM remains convex, the problem becomes a complicated semidefinite program problem, which is hard to be solved efficiently.

All of the above methods often solve the problems in the dual approaches. One of the main contributions in this correspondence is to solve the robust regularized kernel regression in the primal, which leads to an unconstrained optimization problem.

III. Robust Regularized Kernel Regression

A. Theoretical Foundation

In general, a regression problem can be defined as a problem of approximating a multivariate function from the data, which is usually an ill-posed problem. One effective way to solve the problem is based on the theoretical framework of regularization networks [7], [18]. It usually formulates the issue as a variation of finding the embedded function f to solve the following minimization problem:

$$\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} V\left(y_i, f(\mathbf{x}_i)\right) + \lambda\|f\|_{\mathcal{H}}^2 \tag{1}$$

where $V(\cdot,\cdot)$ is the loss function, and $(\mathbf{x}_i, y_i)_{i=1}^n$ denotes the n pairs of samples. The second term is known as smoothness functional.

According to the representer theorem [19], any $f \in \mathcal{H}$ minimizing the regularized risk function in (1) will have a representation of the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \boldsymbol{\alpha}_i k(\mathbf{x}_i, \mathbf{x}) \tag{2}$$

or

$$f(\mathbf{x}) = \sum_{i=1}^{n} \boldsymbol{\alpha}_i k(\mathbf{x}, \mathbf{x}_i) + b \tag{3}$$

where b is an offset term that may sometimes be ignored for simplicity, and $k(\cdot,\cdot)$ is the kernel function that is either fully or compactly supported.

There are various choices for the loss function. For example, RLS, also known as ridge regression [20], considers a squared error loss function, which is the classical L_2 regularization networks. The loss function is defined as $V(y, f(\mathbf{x})) = u^2$, where the variable u is defined as $u = y - f(\mathbf{x})$. If $V(\cdot,\cdot)$ is an ε-insensitive function $V(y, f(\mathbf{x})) = u_\varepsilon$, the resulting problem becomes the ε-SVR.

B. Robust Regularized Kernel Regression

In general, the outliers are overemphasized using the L_2 norm in RLS. Whereas the L_1-norm loss function can avoid this problem, it overemphasizes error on points close to the predicted line. Fortunately, the Huber loss function [15] provides a compound solution, which enjoys the advantages of both the L_1 and L_2 norms. The Huber loss function is shown in Fig. 1(a). It is also named as robust regression in the literature [4], and the loss function is defined as follows:

$$\mathcal{V}(u) = \begin{cases} u^2 & |u| \le m \\ m\,(2|u| - m) & |u| > m \end{cases} \tag{4}$$

where m indicates the switcher from quadratic to linear. In addition, when m becomes larger, the loss function is more like quadratic. Thus, it achieves an appropriate emphasis on the large and small errors.

It is convenient to extend the Huber loss function by introducing an ε-insensitive region, which is shown in Fig. 1(b). We denote it as the Huber ε-insensitive loss function, and the regression problem is defined as the enhanced robust regression. Moreover, this hybrid loss function is similar to the soft insensitive loss function [1], which enjoys the advantage of sparse solutions. It can be explicitly defined as follows:

$$\mathcal{V}\left(y, f(\mathbf{x})\right) = \begin{cases} m\,(2(u-\varepsilon) - m) & u - \varepsilon > m \\ u^2 & 0 < u - \varepsilon \le m \\ 0 & |u| \le \varepsilon \\ u^2 & -m \le u + \varepsilon < 0 \\ -m\,(2(u+\varepsilon) + m) & u + \varepsilon < -m. \end{cases} \tag{5}$$

Consider the kernel reproducing property, we can combine (1) and (3)

$$\sum_{i=1}^{n} \mathcal{V}\left(y_i, \sum_{i=1}^{n} k(\mathbf{x}_i, \mathbf{x}_j)\boldsymbol{\alpha}_j\right) + \lambda \sum_{i,j=1}^{n} \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j k(\mathbf{x}_i, \mathbf{x}_j)$$

where $\boldsymbol{\alpha}_i$ is the coefficient defined in the primal, which is not interpreted as Lagrange multipliers.

Let $\mathbf{k}_i$ denote $(k(\mathbf{x}_1, \mathbf{x}_i), \ldots, k(\mathbf{x}_l, \mathbf{x}_i))$; rewrite the aforementioned objective function as follows:

$$E = \sum_{i=1}^{n} \mathcal{V}\left(y, \mathbf{k}_i^{\mathrm{T}}\boldsymbol{\alpha}\right) + \lambda \boldsymbol{\alpha}^{\mathrm{T}} K \boldsymbol{\alpha} \tag{6}$$

where K is the kernel matrix with $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

IV. Optimization Methods: Dual Versus Primal

Given the robust regularized kernel regression problem, we first formulate it into dual forms and then offer a direct primal formulation. Furthermore, we propose an efficient algorithm to solve the primal, which is able to reduce the size of the problem. We also provide a similar primal solution for the enhanced robust regression approach.

A. Dual Optimization

Because the previous robust regression methods with the Huber loss function [4] only consider the linear case without adding any regularization term in the objective function, we formulate this problem under the framework of regularization networks in this correspondence. Frequently, a nonlinear mapping function can be employed in the robust regression. Furthermore, the regularization term is employed to avoid the overfitting issue and ensure the numerical stability.

Currently, there are two different approaches for formulating the robust regularized kernel regression into dual forms.

1) Dual I: One can formulate the robust regularized kernel regression into the dual form as follows:

$$\begin{aligned} \min_{\mathbf{v},\mathbf{w},\boldsymbol{\alpha},b} \quad & \mathbf{w}^{\mathrm{T}}\mathbf{w} + +2m\mathbf{1}^{\mathrm{T}}\mathbf{v} + \lambda\boldsymbol{\alpha}^{\mathrm{T}}K\boldsymbol{\alpha} \\ s.t. \quad & -\mathbf{w} - \mathbf{v} \preceq K\boldsymbol{\alpha} + b - \mathbf{y} \preceq \mathbf{w} + \mathbf{v} \\ & 0 \preceq \mathbf{w} \preceq m \\ & 0 \preceq \mathbf{v} \end{aligned}$$

with variables $\boldsymbol{\alpha}, \mathbf{v}, \mathbf{w},$ and b.

2) Dual II: Another dual form can be given as [4]

$$\min_{\mathbf{w},\boldsymbol{\alpha}} \quad \mathbf{w}^{\mathrm{T}}\mathbf{w} + 2m\mathbf{1}^{\mathrm{T}}|\mathbf{u} - \mathbf{w}| + \lambda\boldsymbol{\alpha}^{\mathrm{T}}K\boldsymbol{\alpha}.$$

Moreover, this dual form in turn can be reduced to a simple quadratic programming (QP) problem as follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha},\mathbf{w},\mathbf{v}} \quad & \mathbf{w}^{\mathrm{T}}\mathbf{w} + 2m\mathbf{1}^{\mathrm{T}}\mathbf{v} + \lambda\boldsymbol{\alpha}^{\mathrm{T}}K\boldsymbol{\alpha} \\ & -\mathbf{v} \preceq K\boldsymbol{\alpha} + b - \mathbf{y} - \mathbf{w} \preceq \mathbf{v} \\ & \mathbf{v} \succeq 0. \end{aligned}$$

As suggested in [4], a simplification on the aforementioned dual problem will be more efficient, which is derived as follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha},\mathbf{w},\mathbf{v},\mathbf{t}} \quad & \mathbf{w}^{\mathrm{T}}\mathbf{w} + 2m\mathbf{1}^{\mathrm{T}}(\mathbf{v} + \mathbf{t}) + \lambda\boldsymbol{\alpha}^{\mathrm{T}}K\boldsymbol{\alpha} \\ & K\boldsymbol{\alpha} + b - \mathbf{y} - \mathbf{w} = \mathbf{v} - \mathbf{t} \\ & \mathbf{v}, \mathbf{t} \succeq 0. \end{aligned}$$

Obviously, all the aforementioned dual optimizations are QP problems, which can be solved by some standard QP solver [21]. Being a convex optimization problem, the dual optimization can be solved in polynomial time. Due to the introduction of extra variables, the problem size becomes large and hence may affect the efficiency of the solution.

Algorithm 1: Robust Regularized Kernel Regression

Input: K, γ, and m

1: Initialize $\boldsymbol{\alpha}$ and b

2: Calculate the residual and $S = \{S_1 \cup S_2 \cup S_3\}$

3: **Repeat**

4: Estimate $\boldsymbol{\alpha}, b$ through Eqn. (12)

5: $\mathbf{u} = K\boldsymbol{\alpha} + b \cdot 1 - \mathbf{y}$

6: Categorize the training data by the residual error $S = \{S_1 \cup S_2 \cup S_3\}$

7: Calculate the energy E, and ΔE

8: **Until** ($\Delta E > \epsilon$ and $S_{old} \cap S \neq \emptyset$)

Output: K, $\boldsymbol{\alpha}$ and b

Fig. 2. Robust regularized kernel regression (R3) algorithm.

B. Primal Optimization

Instead of solving the optimization problem in dual, we propose a novel primal solution for the robust regularized kernel regression problem. The algorithm is shown in Fig. 2. We rewrite the Huber loss function as follows:

$$\mathcal{V}(y, f(\mathbf{x})) = \begin{cases} m(2u - m) & S_1 = \{\mathbf{x} | u > m\} \\ u^2 & S_2 = \{\mathbf{x} | -m \leq u \leq m\} \\ -m(2u + m) & S_3 = \{\mathbf{x} | u < -m\}. \end{cases}$$

The finite Newton method and its modification are efficient and effective to solve this problem [12], [14]. It can handle the complex piecewise loss function, whereas it is usually challenging for the standard Newton method. According to the objective function in (6), the derivatives of $E(\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$ can be derived as follows:

$$\begin{aligned} \frac{\partial E}{\partial \boldsymbol{\alpha}} &= 2\left(\lambda K\boldsymbol{\alpha} + \sum_{i=1}^{|S_2|} \mathbf{k}_i\left(\mathbf{k}_i^{\mathrm{T}}\boldsymbol{\alpha} - y_i\right) + m\sum_{i=1}^{|S_1|}\mathbf{k}_i - m\sum_{i=1}^{|S_3|}\mathbf{k}_i\right) \\ &= 2(\lambda K\boldsymbol{\alpha} + KI^0K\boldsymbol{\alpha} + K\mathbf{q}) \end{aligned}$$

where $\mathbf{q}$ is an n-dimensional vector, which equals

$$\mathbf{q} = -I^0\mathbf{y} + m\mathbf{e} \tag{7}$$

and I^0 is an $n \times n$ diagonal matrix with the first $|S_2|$ entries being one and the others zero. Moreover, $\mathbf{e}$ is defined as

$$\mathbf{e}_i = \begin{cases} 1 & \mathbf{x}_i \in S_1 \\ 0 & \mathbf{x}_i \in S_2 \\ -1 & \mathbf{x}_i \in S_3. \end{cases}$$

Because the derivatives of $E(\boldsymbol{\alpha})$ with respect to the variable $\boldsymbol{\alpha}$ vanish for optimality, it leads to the following solution:

$$\boldsymbol{\alpha} = -(\lambda I + I^0K)^{-1}\mathbf{q}. \tag{8}$$

When the offset b is considered, there are two approaches to tackle this issue. One is to directly estimate the offset b from the average fitting residual error. Another is to perform a joint optimization on both $\boldsymbol{\alpha}$ and b. We prefer the latter approach for better empirical performance. Let $\acute{\mathbf{q}}$ denote as

$$\acute{\mathbf{q}} = I^0(b \cdot \mathbf{1} - y) + m\mathbf{e}. \tag{9}$$

Consider the bias b; the gradient ∇ becomes

$$\nabla = \begin{bmatrix} \frac{\partial E}{\partial \boldsymbol{\alpha}} \\ \\ \frac{\partial E}{\partial b} \end{bmatrix} = 2 \begin{bmatrix} \lambda K\boldsymbol{\alpha} + KI^0K\boldsymbol{\alpha} + K\acute{\mathbf{q}} \\ \\ I^0K\boldsymbol{\alpha} + \acute{\mathbf{q}} \end{bmatrix}.$$

Therefore, the Hessian matrix H is derived as follows:

$$H = 2 \begin{bmatrix} \lambda K + KI^0K & I^0K \\ KI^0 & I^0 \end{bmatrix}.$$

Thus, we rewrite the gradient matrix ∇ as follows:

$$\nabla = H \begin{bmatrix} \boldsymbol{\alpha} \\ 0 \end{bmatrix} - 2 \begin{bmatrix} K\acute{\mathbf{q}} \\ \acute{\mathbf{q}} \end{bmatrix}.$$

The update equation for each Newton optimization is as follows:

$$\begin{bmatrix} \boldsymbol{\alpha}' \\ b' \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} - \gamma H^{-1}\nabla \tag{10}$$

where γ is the step size that can be safely set to one.

The Hessian matrix H is able to be decomposed into the product of two matrices

$$H = \begin{bmatrix} K & 0 \\ 1 & -\lambda \end{bmatrix} \begin{bmatrix} \lambda I + I^0K & 1 \\ 1 & 0 \end{bmatrix}.$$

Therefore, it can be derived that

$$H^{-1}\nabla = \begin{bmatrix} \boldsymbol{\alpha} \\ 0 \end{bmatrix} - \begin{bmatrix} \lambda I + I^0K & 1 \\ 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \acute{\mathbf{q}} \\ 0 \end{bmatrix}. \tag{11}$$

Substituting (11) into (12), the final update equation is obtained as follows:

$$\begin{bmatrix} \boldsymbol{\alpha}' \\ b' \end{bmatrix} = \begin{bmatrix} (1-\gamma)\boldsymbol{\alpha} \\ b \end{bmatrix} + \gamma \begin{bmatrix} \lambda I + I^0K & 1 \\ 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \acute{\mathbf{q}} \\ 0 \end{bmatrix}. \tag{12}$$

C. Efficient Algorithm

Directly computing the matrix inversion in (12) may be computationally expensive. In this section, we propose a fast algorithm to compute $\boldsymbol{\alpha}$ and b.

Let us denote the matrix M as $M = \lambda I + I^0K$ and define N as

$$N = \begin{bmatrix} \lambda I + I^0K & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix} = \begin{bmatrix} M & \mathbf{1} \\ \mathbf{1}^{\mathrm{T}} & 0 \end{bmatrix}.$$

According to the matrix inversion lemma

$$N^{-1} = \begin{bmatrix} M^{-1}(I - \mathbf{1}\cdot g\cdot \mathbf{1}^{\mathrm{T}}M^{-1}) & M^{-1}\cdot \mathbf{1}\cdot g \\ g\cdot \mathbf{1}^{\mathrm{T}}\cdot M^{-1} & -g \end{bmatrix}$$

where $g = 1/(\mathbf{1}^{\mathrm{T}}\cdot M^{-1}\cdot \mathbf{1})$. Substituting the previous equation into (12), we can obtain

$$\begin{bmatrix} \boldsymbol{\alpha}' \\ b' \end{bmatrix} = \begin{bmatrix} (1-\gamma)\boldsymbol{\alpha} + \gamma M^{-1}(\mathbf{q} - \mathbf{1}\cdot g\cdot \mathbf{1}^{\mathrm{T}}M^{-1}\mathbf{q} \\ b + \gamma g\cdot \mathbf{1}^{\mathrm{T}}\cdot M^{-1}\mathbf{q} \end{bmatrix}.$$

Assuming that the step size $\gamma = 1$, the update of α and b can be computed by

$$b' = b + g\cdot \mathbf{1}^{\mathrm{T}}\cdot M^{-1}\mathbf{q} \tag{13}$$

$$\boldsymbol{\alpha}' = M^{-1}\left[\mathbf{q} - (b' - b)\cdot \mathbf{1}\right]. \tag{14}$$

It is shown that only the inverse of M is involved in the computation. Given the fact that the lower left block of M is zero, the inverse of M can be efficiently computed. According to the matrix inversion lemma, we then have

$$M^{-1} = \begin{bmatrix} K_{S_2,S_2} + \lambda I & K_{S_2,S_1\cup S_3} \\ 0 & \lambda I \end{bmatrix}^{-1} = \begin{bmatrix} K_{S_2,S_2}^{-1} & -\frac{1}{\lambda}K_{S_2,S_2}^{-1}K_{S_2,S_1\cup S_3} \\ 0 & \frac{1}{\lambda}I \end{bmatrix}$$

where $K_{(\cdot,\cdot)}$ represents the submatrix of K.

Remark: The complexity of the proposed approach is $\mathcal{O}(|S_2|^3 + n^2)$, where $|S_2| \leq n$, whereas the RLS is $\mathcal{O}(n^3)$.

D. Enhanced Robust Regression

In similar, we solve the enhanced robust regression problem directly in primal and rewrite the loss function as follows:

$$\mathcal{V}(y, f(\mathbf{x})) = \begin{cases} m(2(u-\varepsilon) - m) & S_1 = \{\mathbf{x} | u - \varepsilon > m\} \\ (u-\varepsilon)^2 & S_2 = \{\mathbf{x} | 0 < u - \varepsilon \leq m\} \\ 0 & S_3 = \{\mathbf{x} \| u| \leq \varepsilon\} \\ (u+\varepsilon)^2 & S_4 = \{\mathbf{x} | -m \leq u + \varepsilon < 0\} \\ -m(2(u+\varepsilon) + m) & S_5 = \{\mathbf{x} | u + \varepsilon < -m\}. \end{cases}$$

According to the definition, m must be greater than ε. Obviously, the enhanced robust regression problem degenerated into the robust kernel regression when ε equals zero. The derivatives of $E(\boldsymbol{\alpha})$ are derived as follows:

$$\begin{aligned} \frac{\partial E}{\partial \boldsymbol{\alpha}} =& 2\lambda K\boldsymbol{\alpha} - 2\sum_{i=1}^{|S_2|} K_i\left(K_i^{\mathrm{T}}\boldsymbol{\alpha} - y_i + \varepsilon\right) + 2m\sum_{i=1}^{|S_1|} K_i \\ & - 2\sum_{i=1}^{|S_4|} K_i\left(K_i^{\mathrm{T}}\alpha - y_i - \varepsilon\right) - 2m\sum_{i=1}^{|S_5|} K_i \\ =& 2(\lambda K\boldsymbol{\alpha} + KI^0K\boldsymbol{\alpha} + K\mathbf{q}) \end{aligned}$$

where $\mathbf{q}$ is denoted as $\mathbf{q} = -I^0\mathbf{y} + \mathbf{e}$, and I^0 is an $n\times n$ diagonal matrix with the first $|S_2| + |S_4|$ entries being one and the others zero. Here, $\mathbf{e}$ is a vector; each element is defined as

$$\mathbf{e}_i = \begin{cases} m & \mathbf{x}_i \in S_1 \\ \varepsilon & \mathbf{x}_i \in S_2 \\ 0 & \mathbf{x}_i \in S_3 \\ -\varepsilon & \mathbf{x}_i \in S_4 \\ -m & \mathbf{x}_i \in S_5. \end{cases}$$

One can see that the form of derivatives in the enhanced robust regression is equivalent to the robust regularized kernel regression except the difference of defining $\mathbf{q}$ and $\mathbf{e}$. Thus, we can solve this optimization by using the previous efficient algorithm for the robust regularized regression.

V. Experimental Results

We have implemented both the dual and primal algorithms of the robust regularized kernel regression in Matlab. For simplicity, the Dual I and Dual II algorithms using regularization are denoted as "D-H1" and "D-H3," respectively. The other Dual II algorithm without a regularization term is denoted as "D-H2," which is equivalent to the typical robust regression method [4]. Our proposed robust regularized kernel regression approach is

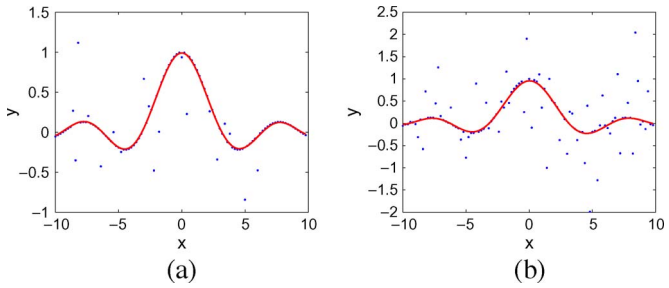

Fig. 3. Two fitting examples on the sinc artificial data set of different noise percentages.

TABLE I
FITTING RESULTS ON sinc DATA SETS WITH DIFFERENT NOISE PERCENTAGES. THE MSE IS ADOPTED AS THE METRIC, AND THE UNIT IS 10^{-2}

0	0.0595	0.001	0.036	0.027	0.028
10	0.603	1.026	0.576	0.003	0.010
std	0.185	0.123	0.179	0.092	0.003
20	0.886	1.292	0.945	0.519	1.317
std	0.120	0.289	0.276	0.103	0.157
30	1.124	1.488	1.145	0.730	1.434
std	0.158	0.218	0.372	0.161	0.381

denoted as "R3." Because the enhanced robust regression performs similarly against the R3 algorithm, our evaluation mainly focuses on the latter. All experiments were carried out on a Pentium-4 3.0-GHz PC with 1-GB RAM.

To conduct the performance evaluation, we adopt three popular data sets for regression in our experiments, including one artificial data set and two real regression data sets from UCI Machine Learning Repository.

A. Experiment I: The sinc Artificial Data

The function $\text{sinc}(x) = \sin\|x\|/x$ is widely used to examine the performance of regression algorithms [1], [8]. In our experiment, both training and testing data sets are generated by uniformly sampling 100 data points from the interval $[-10, 10]$, respectively. The target values are then corrupted by some noise with a normal distribution. The standard deviation σ of the injected noise is set to one. Fig. 3 shows two fitting results under the settings of different noise percentages. In the experiment, all of the regularization and kernel parameters are fairly determined using the training set. The mean-square error (mse) is adopted as the performance metric, which has been widely used in regression tasks.

In the experiment, we generated several versions of data sets with different noise percentages. Then, we repeated each experiment up to 30 runs and summarize average results in Table I. From the experimental results, several observations can be drawn. First, we found that, among the three dual algorithms, D-H2, which is the one without regularization, performed very well in the noise-free case but suffered significantly when the noise increases. The D-H3 algorithm is consistently better than the D-H1 algorithm. Second, we found that the proposed R3 algorithm is comparable to the dual solutions. In particular, our algorithm is better than the two regularized algorithms, which are the D-H1 and D-H3, when the noise is lower and is significantly better than the nonregularized algorithm D-H2 when the noise is higher. These results show that the proposed R3 algorithm is effective in handling noisy data. In addition, we can also observe that RLS with the L_2 norm only works well in the small noise case; the performance becomes worse when data are nosier.

TABLE II
REGRESSION PERFORMANCE ON THE TESTING DATA FOR THE ROBOT ARM DATA SET. THE MAE AND THEIR STANDARD DEVIATION RESULTS ARE REPORTED. THE UNIT IS 10^{-1}

	D-H1	D-H2	D-H3	R3	RLS
y_1	0.558	1.763	0.662	0.558	0.577
std	0.103	0.562	0.112	0.103	0.101
y_2	0.485	2.249	0.578	0.485	0.486
std	0.334	0.164	0.057	0.033	0.041

TABLE III
RESULTS OF TESTING DATA ON THE BOSTON DATA SET

	D-H1	D-H2	D-H3	R3	RLS
mean	1.79	1.87	1.79	1.79	4.87
std	0.21	0.23	0.21	0.21	0.19

B. Experiment II: The Robot Arm Data Set

In the second experiment, we study a regression task of the well-known robot arm problem introduced by Neal [22]. In the robot arm problem, there are two input variables x_1 and x_2, representing joint angles, and two target values y_1 and y_2, representing arm positions in rectangular coordinates. This data set[1] consists of 600 input–target pairs. The target values are contaminated by independent Gaussian noise with a standard deviation of 0.05. In our experiment, we randomly sample 200 examples (pairs) from the data set as a training set and treat the rest part of the data set as the testing set. For experimental settings, all regularization and kernel parameters are optimized according to the training data set. For the performance metric in this experiment, we adopt the mean absolute error (MAE) by following the previous work in the literature [1], [22].

We repeat the same experiment up to 100 runs and summarize average results in Table II. From the experimental results, similar observations can be found. The proposed R3 method obtained the smallest MAE results compared with other solutions.

C. Experiment III: The Boston Housing Data Set

The third experiment is to study the Boston housing problem, which is to estimate the median price of houses in 506 census tracts within the Boston metropolitan area in 1970. Thirteen attributes pertaining to each census tract are available for prediction.[2] For each run of the experiment, the data set is randomly partitioned into a training set of 481 examples and a testing set of 25 examples. In total, 100 runs were conducted. Table III summarizes the average experimental results with respect to the mse metric.

From the results, we can see that the proposed R3 method achieved the best performance, which is the same as the other two dual methods and is better than the D-H2 approach. Meanwhile, we found that all of the proposed primal and dual methods are significantly better than

[1]http://wol.ra.phy.cam.ac.uk/mackay/bigback/dat/.
[2]http://lib.stat.cmu.edu/ data set s/boston.

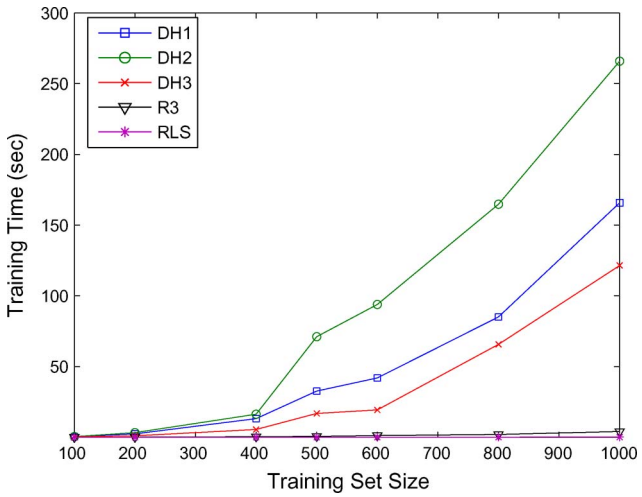

Fig. 4. Comparison of time efficiency with different training sizes.

the RLS solution. The simple L_2-norm loss function leads to the unbiased estimation of the target function, which may suffer difficulty in handling the noisy data. This again verifies the effectiveness of the proposed primal method.

D. Evaluation of Efficiency Performance

Finally, we empirically examine the efficiency performance of the proposed method. To this purpose, we conduct the experiment by comparing the time cost of the proposed algorithm with respect to the other algorithms mentioned earlier. Fig. 4 shows the experimental results of time performance with respect to different training set sizes. From the experimental results, we can clearly see that the proposed primal algorithm is significantly more efficient than the other three dual solutions and is comparable to the RLS approach. It is interesting to find that RLS is slightly faster than the proposed R3 approach. This is because the loss function used in R3 is more complicated than RLS and hence requiring more computational cost.

VI. Conclusion

In this correspondence, we proposed a novel robust regularized kernel regression and suggested an efficient algorithm to solve the problem. We first presented our formulation under a theoretical framework of regularization networks. Based on the solid framework, we solved the robust regularized kernel regression problem directly in the primal form, which leads to an unconstrained optimization. We then proposed an efficient algorithm to reduce the computational cost. Compared with the traditional dual methods, our primal formulation is more efficient to be solved and easier to be implemented. We also extended the proposed solution to the Huber-ϵ insensitive loss function, which enjoys the sparsity in the solution representation. Experimental results on both artificial and real data sets validated the effectiveness and efficiency of our method.

References

[1] W. Chu, S. S. Keerthi, and C. J. Ong, "Bayesian support vector regression using a unified loss function," *IEEE Trans. Neural Netw.*, vol. 15, no. 1, pp. 29–44, Jan. 2004.
[2] S.-I. Lee, H. Lee, P. Abbeel, and A. Y. Ng, "Efficient L_1 regularized logistic regression," in *Proc. AAAI*, 2006, pp. 401–408.
[3] Y.-J. Lee, W.-F. Hsieh, and C.-M. Huang, "ϵ-SSVR: A smooth support vector machine for ϵ-insensitive regression," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 5, pp. 678–685, May 2005.
[4] O. L. Mangasarian and D. R. Musicant, "Robust linear and support vector regression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 9, pp. 950–955, Sep. 2000.
[5] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.
[6] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Stat.*, vol. 35, no. 1, pp. 73–101, 1964.
[7] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Adv. Comput. Math.*, vol. 13, no. 1, pp. 1–50, Apr. 2000.
[8] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[9] O. Mangasarian, "Generalized support vector machines," Comput. Sci. Dept., Univ. Wisconsin, Madison, WI, Tech. Rep. 98-14, 1998.
[10] A. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.
[11] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Learning*. Cambridge, MA: MIT Press, 1999, pp. 185–208.
[12] S. Keerthi, O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," *J. Mach. Learn. Res.*, vol. 7, pp. 1493–1515, Dec. 2006.
[13] O. L. Mangasarian, "A finite Newton method for classification," *Optim. Methods Softw.*, vol. 17, no. 5, pp. 913–929, Jan. 2002.
[14] O. Chapelle, "Training a support vector machine in the primal," *Neural Comput.*, vol. 19, no. 5, pp. 1155–1178, 2007.
[15] P. J. Huber, "The 1972 Wald lecture robust statistics: A review," *Ann. Math. Stat.*, vol. 43, no. 4, pp. 1041–1067, Aug. 1972.
[16] C.-C. Chuang, S.-F. Su, J.-T. Jeng, and C.-C. Hsiao, "Robust support vector regression networks for function approximation with outliers," *IEEE Trans. Neural Netw.*, vol. 13, no. 6, pp. 1322–1330, Nov. 2002.
[17] L. Xu, K. Crammer, and D. Schuurmans, "Robust support vector machine training via convex outlier ablation," in *Proc. AAAI*, 2006, pp. 536–542.
[18] J. Zhu, S. C. Hoi, and M. R. Lyu, "A multi-scale Tikhonov regularization scheme for implicit surface modelling," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2007, pp. 1–7.
[19] B. Schölkopf, R. Herbrich, and A. Smola, "A generalized representer theorem," in *Proc. NeuroCOLT*, 2001, pp. 416–426.
[20] C. Saunders, A. Gammerman, and V. Vovk, "Ridge regression learning algorithm in dual variables," in *Proc. ICML*, 1998, pp. 515–521.
[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[22] R. M. Neal, "Bayesian learning for neural networks," Ph.D. dissertation, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 1995.