Communications for Statistical Applications and Methods
2017, Vol. 24, No. 6, 651–662

https://doi.org/10.29220/CSAM.2017.24.6.651
Print ISSN 2287-7843 / Online ISSN 2383-4757

A convenient approach for penalty parameter selection in robust lasso regression

Jongyoung Kim[a], Seokho Lee[1,a]

[a]Department of Statistics, Hankuk University of Foreign Studies, Korea

Abstract

We propose an alternative procedure to select penalty parameter in L_1 penalized robust regression. This procedure is based on marginalization of prior distribution over the penalty parameter. Thus, resulting objective function does not include the penalty parameter due to marginalizing it out. In addition, its estimating algorithm automatically chooses a penalty parameter using the previous estimate of regression coefficients. The proposed approach bypasses cross validation as well as saves computing time. Variable-wise penalization also performs best in prediction and variable selection perspectives. Numerical studies using simulation data demonstrate the performance of our proposals. The proposed methods are applied to Boston housing data. Through simulation study and real data application we demonstrate that our proposals are competitive to or much better than cross-validation in prediction, variable selection, and computing time perspectives.

Keywords: adaptive lasso, cross validation, lasso, robust regression, variable selection

1. Introduction

Regularized regression is popularly used to incorporate various assumptions that the model should require (Bishop, 2006; Hastie *et al.*, 2001; Murphy, 2012). Type of regularization depends on a specific model assumption. For example, in functional regression, the coefficient parameter is assumed to be a smooth function. In sparse regression, some coefficient parameters corresponding to insignificant variables are expected to be zero. To incorporate smoothness and sparsity on parameter estimate, the regularization technique is often used. Such regularization is achieved under a penalized loss minimization framework. Consider a regression model $y_i = \beta_0 + \mathbf{x}_i^T\boldsymbol{\beta} + \epsilon_i$ with a training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)|\mathbf{x}_i \in \mathbb{R}^p, y_i \in \mathbb{R}^1, i = 1, 2, \dots, n\}$ and error variance $\mathrm{var}(\epsilon_i) = \sigma^2$. The objective function to be minimized is a penalized loss:

$$\ell(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n} \rho(r_i) + P(\boldsymbol{\beta}; \lambda), \tag{1.1}$$

where $r_i = (y_i - \beta_0 - \mathbf{x}_i^T\boldsymbol{\beta})/\sigma$. Squared loss function $\rho(u) = u^2$ is used in regression. In presence of outliers in training data, robust loss functions, Huber loss or bisquare loss for example, are often used to circumvent harmful effects from outliers. Penalty function $P(\cdot\,;\cdot)$ is chosen according to the purpose of regularization. Roughness penalty is used for a smooth function estimate and sparsity-inducing penalty for sparse estimate. Regularization and goodness-of-fit are balanced at an optimal

[1] Corresponding author: Department of Statistics, Hankuk University of Foreign Studies, 50 Oedae-ro 54beon-gil, Mohyeon-myeon, Cheoin-gu, Yongin 17035, Korea. E-mail: lees@hufs.ac.kr

Published 30 November 2017 / journal homepage: http://csam.or.kr

penalty parameter λ. The optimal λ is chosen for the best prediction. In absence of test data, cross validation (CV) is commonly used for test mean squared errors (MSE) estimation in regression. CV is a generic model selection criterion which can be applied to many predictive models where the response variable is based on even no concrete probabilistic ground. Even with its versatility, CV suffers from heavy computational burden.

In this research, we propose an alternative approach to CV in robust regression with L_1 penalization. This approach is based on marginalizing prior distribution over the penalty parameter. Thus, resulting objective function does not include the penalty parameter due to marginalizing it out. But its estimating equation automatically sets a tentative penalty parameter, and we can use it as the penalty parameter in the penalized regression estimating procedure. This idea is straightforwardly generalized to variable-wise penalization, which is often prohibitive through CV.

This paper is organized as follows. In Section 2, we introduce robust lasso regression and provide Bayesian approach to marginalizing penalty parameter and its estimating algorithm. Variable-wise penalization is introduced in the same section. The performance of our proposals are demonstrated through simulation studies and Boston housing data in Section 3. The paper ends with some remarks in Section 4.

2. Methodology

2.1. Robust lasso regression

Consider a linear model $y_i = \beta_0 + \mathbf{x}_i^T\boldsymbol{\beta} + \epsilon_i$ for $i = 1, \dots, n$. In the presence of outlying observations among the training dataset, robust loss is used to reduce the outlier effect in regression (Maronna *et al.*, 2006). When we impose some regularization on the coefficient $\boldsymbol{\beta} \in \mathbb{R}^p$, we add a penalty function on $\boldsymbol{\beta}$ to the empirical loss and minimize the penalized empirical loss (1.1) to find a robust coefficient estimate. Well-known robust loss functions are Huber loss,

$$\rho_H(u|k) = \begin{cases} u^2, & \text{if } |u| \le k, \\ 2k|u| - k^2, & \text{if } |u| > k, \end{cases}$$

and Tukey's bisquare loss,

$$\rho_B(u|k) = \begin{cases} 1 - \left\{1 - \left(\dfrac{u}{k}\right)^2\right\}^3, & \text{if } |u| \le k, \\ 1, & \text{if } |u| > k. \end{cases}$$

Both loss functions include the additional parameter k, which regulates robustness and efficiency of resulting coefficient estimate. Throughout this study, we use $k = 1.345$ for Huber loss and $k = 4.685$ for bisquare loss for 95% efficiency under normality (Maronna *et al.*, 2006).

As in traditional linear regression, we can apply regularization techniques in robust regression (El Ghaoui and Lebret, 1997; Owen, 2006). For variable selection as well as prediction enhancement, L_1 penalty $P(\boldsymbol{\beta}; \lambda) = \lambda\|\boldsymbol{\beta}\|_1 = \lambda\sum_{j=1}^p |\beta_j|$ can be imposed. It becomes lasso if we use the square loss $\rho(u) = u^2$. Thus, robust lasso regression is done by minimizing the penalized objective function (1.1). A usual way for minimizing (1.1) is to use Newton-Raphson algorithm for M-estimates in robust statistics. To look at this more closely, we take a derivative of (1.1) with respect to β_0 and $\boldsymbol{\beta}$. Then, normal equations become

$$\frac{\partial\ell}{\partial\beta_0} = -\sum_{i=1}^n \frac{w_i r_i}{\sigma} = 0, \qquad \frac{\partial\ell}{\partial\boldsymbol{\beta}} = -\sum_{i=1}^n \frac{w_i r_i \mathbf{x}_i}{\sigma} + \lambda\frac{\partial\|\boldsymbol{\beta}\|_1}{\partial\boldsymbol{\beta}} = \mathbf{0} \tag{2.1}$$

with $w_i = \psi(r_i)/r_i$ and $\psi = \rho'$. The above normal equations are exactly same with those from the below weighted lasso problem:

$$Q(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n} w_i r_i^2 + \lambda \sum_{j=1}^{p} |\beta_j|. \tag{2.2}$$

Thus, robust lasso regression (1.1) with robust loss and L_1 penalty can be solved by iteratively optimizing weighted lasso (2.2), where weight w_i is updated at every iteration step. In robust linear regression, scale parameter σ should be estimated in robust fashion. A common way to obtain robust scale is the normalized median absolute deviates (MADN) of residuals from robust regression (Maronna *et al.*, 2006). MADN is defined as $\text{MADN}(x) = \text{median}(|x - \text{median}(x)|)/0.675$. In this study, we initially fit L_1 regression estimate (least absolute deviation (LAD) estimate; Koenker, 2005). Finally, we obtain σ estimate as MADN of residuals from LAD fit.

To choose an optimal regularization parameter λ, CV is a popular choice. In CV procedure, training data is split into several exclusive pieces. Each piece works as a validation set and remaining pieces are used to fit the model. The fitted model is evaluated on the validation set by MSE. This step is cycled until all pieces serve as a validation set. After finishing the whole cycle, CV score is defined as the averaged MSEs and the regularization parameter λ is chosen to minimize CV score. In the presence of outliers in the training data, MSE based CV score is not reliable because of the possible presence of outliers in the validation set. Thus, robust loss for errors is used in CV score computation, instead of squared errors in the traditional MSE. Generalized cross validation (GCV) is an convenient selection criterion which does not require an onerous splitting-and-fitting procedure as in CV. After fitting the model to whole training data, GCV is obtained using hat matrix $\mathbf{H}_\lambda$ where $\hat{\mathbf{y}} = \mathbf{H}_\lambda \mathbf{y}$. However, GCV is not available in robust regression because the regression coefficient estimate is not given as a linear combination of response variable in most robust regression. Information criterion, including Bayesian information criterion (BIC), is a frequently used for model selection. BIC as well as most other information criterion depend on a data model because data distribution specifies the likelihood as a part of the criterion. In robust regression it is difficult to assume data distribution due to the existence of outliers.

2.2. Bayesian approach by marginalizing regularization parameter

In this section, we introduce a convenient way to choose a penalty parameter under a Bayesian framework. The basic idea is to use marginal distribution of $\boldsymbol{\beta}$ by marginalizing out the penalty parameter. Buntine and Weigend (1991) introduce this idea first in penalized regression and logistic regression problems. Here, we use the same idea in a robust lasso regression. Penalty term $P(\boldsymbol{\beta}; \lambda)$ in (1.1) can be regarded as the negative logarithm of prior distribution for $\boldsymbol{\beta}$. For L_1 penalization, the prior over $\boldsymbol{\beta}$ is a joint distribution of independent Laplace random variables

$$f(\boldsymbol{\beta}|\lambda) = \left(\frac{\lambda}{2}\right)^p \exp\{-\lambda \|\boldsymbol{\beta}\|_1\} = \prod_{j=1}^{p} \frac{\lambda}{2} \exp\{-\lambda |\beta_j|\}.$$

The prior of $\boldsymbol{\beta}$ depends on λ as a scale parameter. To remove the dependency on λ, Buntine and Weigend (1991) impose an improper Jeffrey prior on the hyperparameter λ and marginalize it out. Using Jeffrey prior for λ, $f(\lambda) \propto 1/\lambda$, the marginal prior on $\boldsymbol{\beta}$ becomes

$$f(\boldsymbol{\beta}) = \int_0^\infty f(\boldsymbol{\beta}|\lambda) f(\lambda) d\lambda = \int_0^\infty \frac{\lambda^{p-1}}{2^p} \exp\{-\lambda \|\boldsymbol{\beta}\|_1\} d\lambda = \frac{\Gamma(p)}{2^p \|\boldsymbol{\beta}\|_1^p}.$$

By replacing the original penalty induced from $-\log f(\boldsymbol{\beta}|\lambda)$ in (1.1) by a new penalty from $-\log f(\boldsymbol{\beta})$, we can obtain a penalized loss $\tilde{\ell}$:

$$\tilde{\ell}(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n} \rho(r_i) + p \log \|\boldsymbol{\beta}\|_1. \tag{2.3}$$

Equation (2.3) is a new objective function without λ. For $\boldsymbol{\beta}$ estimation, normal equations from (2.3) become

$$\frac{\partial \tilde{\ell}}{\partial \beta_0} = -\sum_{i=1}^{n} \frac{w_i r_i}{\sigma} = 0, \qquad \frac{\partial \tilde{\ell}}{\partial \boldsymbol{\beta}} = -\sum_{i=1}^{n} \frac{w_i r_i \mathbf{x}_i}{\sigma} + \frac{p}{\|\boldsymbol{\beta}\|_1} \frac{\partial \|\boldsymbol{\beta}\|_1}{\partial \boldsymbol{\beta}} = \mathbf{0}. \tag{2.4}$$

The above normal equation cannot be solved analytically. Note that, by comparing (2.4) with (2.1) from the original objective function, the penalty parameter λ in (2.1) is replaced by the term $p/\|\boldsymbol{\beta}\|_1$ in (2.4). Thus, Buntine and Weigend (1991) suggest an iterative fitting procedure for a penalized least square problems, where $(\beta_0, \hat{\boldsymbol{\beta}})$ is obtained as a weighted penalized least squares solution with a penalty parameter $\lambda = p/\|\boldsymbol{\beta}^o\|_1 = p/\sum_{j=1}^{p} |\beta_j^o|$ using the previous solution $\boldsymbol{\beta}^o = (\beta_1^o, \ldots, \beta_p^o)^T$. We can use the same idea in robust lasso problem by solving iteratively (2.2). Updating formula for $(\beta_0, \boldsymbol{\beta})$ in (2.2) are obtained as a weighted lasso problem using weights $w_i = \psi(r_i^o)/r_i^o$ and penalty $\lambda = p/\sum_{j=1}^{p} |\beta_j^o|$, where $r_i^o = y_i - \hat{\beta}_0^o - \mathbf{x}_i^T \boldsymbol{\beta}^o$.

Through some simulation studies, we observed that this approach produces reliable performance in prediction but is not satisfactory in variable selection. To enhance variable selection performance, we consider a different penalty function for $\boldsymbol{\beta}$. Instead of the common penalty parameter λ for all β_j, we use separate penalty parameters for each variable, in the similar to adaptive lasso (Zou, 2006). Thus, L_1 penalty function $\lambda \sum_{j=1}^{p} |\beta_j|$ in (1.1) is replaced by $P(\boldsymbol{\beta}; \boldsymbol{\lambda}) = \sum_{j=1}^{p} \lambda_j |\beta_j|$ with $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p)^T$. We call the former "robust lasso" and the latter "robust adaptive lasso." This change causes computational challenge in CV because p penalty parameters should be chosen by p-dimensional grid search. This is infeasible in practice even with a moderate size of variables. However, the same Bayesian approach can be easily implemented. With variable-wise penalty parameters, the objective function becomes

$$\ell(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n} \rho(r_i) + \sum_{j=1}^{p} \lambda_j |\beta_j|. \tag{2.5}$$

The coefficient parameters β_j $(j = 1, 2, \ldots, p)$ are assumed to follow Laplace distribution $p(\beta_j|\lambda_j) = (\lambda_j/2)\exp(-\lambda_j|\beta_j|)$ conditional on λ_j. Similar to robust lasso in above, marginal distribution of β_j with Jeffrey prior $p(\lambda_j) \propto 1/\lambda_j$ is obtained as

$$f(\beta_j) = \int_0^\infty f(\beta_j|\lambda_j) f(\lambda_j) d\lambda_j = \int_0^\infty \frac{1}{2} e^{-\lambda_j|\beta_j|} d\lambda_j = \frac{1}{2|\beta_j|}.$$

Penalty function on $\boldsymbol{\beta}$ becomes $-\log f(\boldsymbol{\beta}) = -\log\{\prod_{j=1}^{p} f(\beta_j)\} = \sum_{j=1}^{p} \log|\beta_j| + \text{constant}$. Thus, after marginalizing λ_j out, objective function (2.5) is modified as

$$\tilde{\ell}(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n} \rho(r_i) + \sum_{j=1}^{p} \log|\beta_j|. \tag{2.6}$$

Normal equations of (2.5) are

$$\frac{\partial \tilde{\ell}}{\partial \beta_0} = -\sum_{i=1}^{n} \frac{w_i r_i}{\sigma} = 0, \qquad \frac{\partial \tilde{\ell}}{\partial \beta_j} = \sum_{i=1}^{n} \frac{w_i r_i x_{ij}}{\sigma} + \frac{1}{|\beta_j|}\frac{\partial |\beta_j|}{\partial \beta_j} = 0, \tag{2.7}$$

for $j = 1, 2, \ldots, p$. This normal equation is exactly same with that from ℓ in (2.5) if λ_j is set to be $1/|\beta_j|$. We can still here employ an iterative scheme of estimation. We summarize the whole procedure into the below conceptual algorithm, where we use coordinate descent algorithm to estimate $\boldsymbol{\beta}$.

Algorithm: Robust lasso and robust adaptive lasso by marginalizing penalty parameters

1. (Initialization) Obtain initial parameters $\hat{\beta}_0^o$, $\hat{\boldsymbol{\beta}}^o$ from LAD or LAD-ridge. Set $\hat{\sigma} = \text{MADN}(r_i)$ where r_i are residuals from LAD.
2. Repeat until convergence is met.
   (a) Set $r_i = (y_i - \hat{\beta}_0^o - \mathbf{x}_i^T \hat{\boldsymbol{\beta}}^o)/\hat{\sigma}$ and $w_i = \psi(r_i)/r_i$ for $i = 1, \ldots, n$. For robust lasso, set $\lambda = |\mathcal{I}| / \sum_{j \in \mathcal{I}} |\hat{\beta}_j^o|$ where $\mathcal{I}$ is the index set of nonzero elements in $\hat{\boldsymbol{\beta}}^o$ and $|\mathcal{I}|$ is the number of nonzeros in $\hat{\boldsymbol{\beta}}^o$. For robust adaptive lasso, set $\lambda_j = 1/|\hat{\beta}_j^o|$. If $\hat{\beta}_j^o = 0$, then $\lambda_j = \infty$.

   (b) Compute $\bar{y}^w = \sum_{i=1}^n w_i y_i / \sum_{i=1}^n w_i$ and $\bar{x}_j^w = \sum_{i=1}^n w_i x_{ij} / \sum_{i=1}^n w_i$ for all $j = 1, 2, \ldots, p$. Then, set $\tilde{y}_i = y_i - \bar{y}^w$ and $\tilde{x}_{ij} = x_{ij} - \bar{x}_j^w$.

   (c) For $j = 1, 2, \ldots, p$, compute $y_{i(-j)}$ and update $\hat{\beta}_j$ by

   $$y_{i(-j)} = \tilde{y}_i - \sum_{l \neq j} \tilde{x}_{il} \hat{\beta}_l^o, \qquad \hat{\beta}_j = \text{soft}\left(\frac{\sum_{i=1}^n w_i \tilde{x}_{ij} y_{i(-j)}}{\sum_{i=1}^n w_i \tilde{x}_{ij}^2}, \frac{\hat{\sigma}^2 \lambda_j^*}{\sum_{i=1}^n w_i \tilde{x}_{ij}^2} \right),$$

   where $\text{soft}(x, t) = \text{sign}(x)(|x| - t)_+$ and $u_+ = \max(u, 0)$. Here, we set $\lambda_j^* = \lambda$ for robust lasso and $\lambda_j^* = \lambda_j$ for robust adaptive lasso.

   (d) Update the intercept $\hat{\beta}_0$ by $\hat{\beta}_0 = \bar{y}^w - \sum_{j=1}^p \bar{w}_j^w \hat{\beta}_j$.

   (e) If convergence is not achieved, update $\hat{\beta}_0^o \leftarrow \hat{\beta}_0$ and $\hat{\boldsymbol{\beta}}^o \leftarrow \hat{\boldsymbol{\beta}}$.

2.3. Connection to existing methods

Robust adaptive lasso described in the previous section allows variable-wise penalty parameters for each coefficient. This reminds adaptive lasso where penalty term is given as $\lambda \sum_{j=1}^p \eta_j |\beta_j|$ (Zou, 2006). Zou (2006) suggests $\eta_j = 1/|\hat{\beta}_j|^\gamma$, where $\hat{\beta}_j$ is a consistent estimator, least squares solution suggested, and γ is another tuning parameter. If $\lambda_j = \lambda \eta_j$ and $\gamma = 1$, then $\lambda_j = 1/|\hat{\beta}_j|$ which seems similar to the proposed approach. However, our approach keeps changing $\hat{\boldsymbol{\beta}}^o$ during iterations, while adaptive lasso from Zou (2006) fixes it as the least squares solution or ridge solution if $n < p$. Zou (2006) suggests to use CV for the choice of (λ, γ), which is computationally burdensome due to a 2-dimensional grid search. Automatic relevance determination (ARD) (MacKay, 1995; Tipping 2001) is another approach that considers variable-wse penalization too. Even though ARD is originally developed for sparse kernel regression, it is easily modified to a regular regression problem. ARD assumes Gaussianity for $\boldsymbol{\beta}$, not Laplacianity in adaptive lasso, and achieves a variable selection by reducing down prior variance for negligible variables. Its estimation can be cast into penalized loss framework. In ARD, however, updating formula for λ_j depends all coefficient parameter estimates in the previous iteration and, thus, cannot be separable variable-wise. This causes relatively slow convergence and cannot exploit the simple thresholding scheme.

Our approach is closely related to a traditional Bayesian approach. If $\rho(u) = u^2$, i.e., non-robust regression, lasso can be formulated in Bayesian framework using Gaussian scale mixture as $\beta_j|\tau_j^2, \lambda \sim N(0, \tau_j^2)$ and $\tau_j^2|\lambda \sim \text{gamma}(1, \lambda^2/2)$ (Park and Casella, 2008; West, 1987). Using normality on the response y_i, full Bayesian approach allows Bayesian inference for lasso because the full posterior $f(\boldsymbol{\beta}|\mathcal{D})$ is available. However, typical Markov chain Monte Carlo (MCMC) approximation does not often produce an exact zero solution. Contrast to MCMC, expectation-maximization (EM) algorithm allows sparse estimation under Bayesian formulation (Figueiredo, 2003). For either MCMC or EM, hyperparameter λ should be chosen by empirical Bayes or evidence procedure, where λ is chosen by the maximizer of $f(\mathcal{D}|\lambda)$ called marginal likelihood or evidence. In robust lasso, however, $f(\mathcal{D}|\lambda)$ is not available because Huber or biweight loss functions are not derived from any distribution. Penalty function in robust adaptive lasso can also be formulated under a Bayesian structure by assuming $\beta_j|\tau_j^2, \lambda \sim N(0, \tau_j^2)$, $\tau_j^2|\lambda_j \sim \text{gamma}(1, \lambda_j^2/2)$, and $\lambda_j|a, b \sim \text{inverse-gamma}(a, b)$. It is called hierarchical adaptive lasso (HAL; Lee *et al.*, 2010). Under non-robust situation, EM algorithm is similar to the proposed approach in the sense that M-step produces the same $\boldsymbol{\beta}$ estimation and E-step gives $E(\lambda_j) = (a+1)/(b + |\hat{\beta}_j^o|)$. If $a = b = 0$, then the expected value of λ_j is exactly same with the proposed approach. However, HAL is not appropriate for robust adaptive lasso either because HAL also assumes normality on the response. Another interesting connection can be found in log penalized regression in Zou and Li (2008). Zou and Li (2008) derived logarithm penalty as a limiting version of Bridge penalty with $q \to 0$ using local linear approximation. The difference is that our approach resorts to full iteration while Zou and Li (2008) suggests one-step iteration only.

3. Numerical Studies

3.1. Synthetic data

Artificial data is used to test our proposal in this section. Input variables x_{ij} $(i = 1, 2, \ldots, n; j = 1, 2, \ldots, p)$ are independently generated from $N(0, 1)$. The intercept parameter β_0 is set to zero, and the first 10 slope parameters are generated uniformly on the positive interval from 1 to 2 and remaining slopes are set to zero. i.e., $\beta_j \sim \text{uniform}(1, 2)$ for $j = 1, 2, \ldots, 10$ and $\beta_j = 0$ for $j = 11, \ldots, p$. Response variables y_i are constructed by $y_i = \mathbf{x}_i^T\boldsymbol{\beta} + \epsilon_i$ for $i = 1, \ldots, n$. Thus, the first 10 variables are important in prediction and the remaining $p - 10$ variables are not. To mimic contamination in training data, random error ϵ_i are generated from the contaminated normal distribution

$$\epsilon_i \sim (1-\pi)N(0, \sigma^2) + \pi N(m\sigma^2, \sigma^2).$$

Here, π is a contamination rate. If $\pi = 0$, there is no outlier. Outliers come from normal distribution with shifted mean by m-factor of error variance. The error variance is set to have a five times signal-to-noise ratio in a standard deviation scale. i.e., $\sigma = \text{sd}(\mathbf{x}_i^T\boldsymbol{\beta}) \times 0.2$. We used glmnet() and cv.glmnet() functions in glmnet package for CV. For non-robust case $(\pi = 0)$, the direct use of cv.glmnet() is sufficient and efficient. For a robust case, weight w_i should be updated at every iteration. Thus, we devise a loop for CV and use glmnet() function for model fitting inside the loop. We conducted 5-fold CV over 100 grid points as suggested in glmnet package. BIC is also considered for comparison study (Chang *et al.*, 2017; Lambert-Lacroix and Zwald, 2011) and the same grid set used in CV was also used in BIC.

Table 1: Simulation 1 - average of 100 test RMSEs and its standard error (in parenthesis)

n	π	RMSEs										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
100	0.0	2.7167 (0.0250)	2.7373 (0.0280)	2.7444 (0.0230)	2.7413 (0.0260)	2.7457 (0.0280)	2.7589 (0.0240)	2.7352 (0.0230)	2.7748 (0.0270)	2.7696 (0.0280)	2.8025 (0.0250)	2.7723 (0.0250)
	0.1	4.5879 (0.0540)	5.3437 (0.0580)	4.8752 (0.0720)	2.8645 (0.0290)	5.1600 (0.0560)	2.9041 (0.0270)	2.8670 (0.0270)	2.7688 (0.0280)	5.2952 (0.0450)	2.7862 (0.0260)	2.7486 (0.0260)
	0.2	6.4616 (0.0820)	6.8108 (0.0660)	6.9528 (0.1150)	3.4884 (0.0490)	5.2903 (0.0600)	3.5763 (0.0470)	3.5501 (0.0500)	2.7830 (0.0300)	5.0319 (0.0680)	2.7922 (0.0280)	2.7468 (0.0290)
	0.3	8.3657 (0.1160)	8.5412 (0.0970)	8.9019 (0.1570)	6.8079 (0.1810)	6.9848 (0.1890)	7.5472 (0.1690)	7.6094 (0.1700)	6.1575 (0.2590)	6.3042 (0.2500)	6.9350 (0.2670)	6.9094 (0.2650)
1,000	0.0	2.5476 (0.0070)	2.5470 (0.0070)	2.5384 (0.0060)	2.5349 (0.0060)	2.5389 (0.0060)	2.5388 (0.0060)	2.5352 (0.0060)	2.5344 (0.0060)	2.5384 (0.0060)	2.5387 (0.0060)	2.5348 (0.0060)
	0.1	3.5918 (0.0080)	3.6337 (0.0080)	3.6147 (0.0080)	2.5858 (0.0060)	2.6246 (0.0070)	2.5908 (0.0060)	2.5861 (0.0060)	2.5369 (0.0060)	2.5718 (0.0070)	2.5400 (0.0060)	2.5349 (0.0060)
	0.2	5.4718 (0.0160)	5.5318 (0.0160)	5.4964 (0.0170)	2.8059 (0.0060)	2.8917 (0.0070)	2.8146 (0.0050)	2.8054 (0.0050)	2.5401 (0.0060)	2.5818 (0.0070)	2.5424 (0.0060)	2.5352 (0.0060)
	0.3	7.5890 (0.0260)	7.6413 (0.0250)	7.6104 (0.0260)	3.6360 (0.0060)	4.1826 (0.0300)	3.6681 (0.0060)	3.6514 (0.0060)	2.5429 (0.0060)	2.7074 (0.0190)	2.5447 (0.0060)	2.5343 (0.0060)
10,000	0.0	3.7873 (0.0050)	3.6779 (0.0050)	2.5022 (0.0020)	2.5022 (0.0020)	2.5033 (0.0020)	2.5024 (0.0020)	2.5021 (0.0020)	2.5022 (0.0020)	2.5032 (0.0020)	2.5024 (0.0020)	2.5020 (0.0020)
	0.1	4.0398 (0.0030)	4.0137 (0.0030)	3.4280 (0.0030)	2.5387 (0.0020)	2.5433 (0.0020)	2.5396 (0.0020)	2.5390 (0.0020)	2.5024 (0.0020)	2.5054 (0.0020)	2.5026 (0.0020)	2.5021 (0.0020)
	0.2	5.4909 (0.0040)	5.4722 (0.0040)	5.2731 (0.0050)	2.7283 (0.0020)	2.7396 (0.0020)	2.7315 (0.0020)	2.7305 (0.0020)	2.5030 (0.0020)	2.5070 (0.0020)	2.5033 (0.0020)	2.5026 (0.0020)
	0.3	7.4267 (0.0070)	7.4242 (0.0070)	7.3777 (0.0080)	3.4506 (0.0020)	3.4673 (0.0020)	3.4560 (0.0020)	3.4542 (0.0020)	2.5029 (0.0020)	2.5089 (0.0020)	2.5032 (0.0020)	2.5021 (0.0020)

Best performer for each case is highlighted.
RMSE = root mean squared error; S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

The first simulation considers $n > p$ situation. We increase sample size $n = 100, 1{,}000, 10{,}000$ with a fixed $p = 20$. Proportion of outliers is $\pi = 0, 0.1, 0.2, 0.3$ and $m = 5$ for outlier generation. Test data of size 10,000 is additionally generated without outliers and used to measure prediction error. Prediction error is calculated as root mean square of error (RMSE). Simulation is repeated 100 times and we report the average and standard error of RMSEs in Table 1. We name square, Huber, bisquare losses by “S”, “H”, “B”, respectively. “BIC” for BIC, “L” for robust lasso, and “A” for robust adaptive lasso. Thus, “B.A” means robust adaptive lasso with bisquare loss. From Table 1, in absence of outliers ($\pi = 0$), traditional lasso using square loss under CV is best in prediction. However, as π increases, square loss is outperformed by robust loss. Bisquare loss is generally better than Huber loss, which demonstrates the well-known fact that nonconvex loss is better than convex loss in robust regression. Overall, robust adaptive lasso performs best among other competitors. We observe that CV and our proposals are hardly distinguishable in standard error scale, as the sample size increases. We compute a false negative rate (FN) and false positive rate (FP), and report their average in Table 2. FN is defined as a ratio of a falsely claimed zero, while they are actually not zero; however, FP is defined as a ratio of falsely claimed nonzero, while they are actually zero. FN is almost zero for all methods, except for a small sample size. But we observe that FP seems quite high. Note that low FN and high FP indicates that almost all variables are selected from a variable selection, implying a low selection ability. Considering this, the BIC selection seems to perform very well in a large sample case. Table 3 presents average computing time for each method. Robust lasso and robust adaptive lasso are enormously faster than CV while their prediction and variable selection performance is comparable or better than CV. Our proposals are much faster than CV and BIC in penalized robust regression.

We conduct the second simulation where variable size is larger than sample size. Sample size is fixed as $n = 100$ and variable size are set to be $p = 50, 100, 200, 400$. In this simulation, outliers

Table 2: Simulation 1 - average of false negative and false positive rates

n	π	False negative										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
100	0.0	0.000	0.001	0.000	0.001	0.001	0.000	0.001	0.001	0.001	0.000	0.002
	0.1	0.174	0.778	0.004	0.002	0.906	0.001	0.003	0.000	0.959	0.000	0.001
	0.2	0.457	0.909	0.003	0.047	0.824	0.003	0.036	0.000	0.807	0.000	0.005
	0.3	0.544	0.949	0.036	0.184	0.197	0.059	0.294	0.155	0.184	0.059	0.301
1,000	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.2	0.000	0.005	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.3	0.000	0.016	0.000	0.000	0.138	0.000	0.000	0.000	0.040	0.000	0.000
10,000	0.0	0.016	0.010	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.3	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
n	π	False positive										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
100	0.0	0.631	0.364	0.976	0.630	0.380	0.966	0.400	0.642	0.419	0.954	0.390
	0.1	0.424	0.046	0.985	0.616	0.001	0.952	0.400	0.648	0.000	0.935	0.307
	0.2	0.287	0.013	0.983	0.496	0.013	0.935	0.407	0.627	0.006	0.883	0.183
	0.3	0.241	0.011	0.935	0.587	0.573	0.900	0.465	0.582	0.538	0.852	0.379
1,000	0.0	0.197	0.149	0.991	0.677	0.240	0.980	0.320	0.667	0.253	0.979	0.305
	0.1	0.629	0.225	0.996	0.632	0.020	0.981	0.339	0.646	0.004	0.969	0.242
	0.2	0.649	0.191	0.993	0.641	0.017	0.975	0.367	0.641	0.000	0.977	0.174
	0.3	0.626	0.194	0.990	0.620	0.038	0.968	0.432	0.650	0.002	0.935	0.069
10,000	0.0	0.000	0.000	0.993	0.606	0.169	0.993	0.290	0.604	0.162	0.993	0.275
	0.1	0.000	0.000	0.996	0.617	0.006	0.991	0.300	0.609	0.001	0.991	0.219
	0.2	0.000	0.000	0.999	0.604	0.003	0.993	0.322	0.598	0.000	0.992	0.151
	0.3	0.018	0.016	0.998	0.573	0.014	0.989	0.379	0.618	0.000	0.976	0.045

S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

Table 3: Simulation 1 - average of 100 computing times (in seconds)

n	π	Computing time in seconds										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
100	0.0	0.0640	0.1490	0.0175	4.4690	0.8743	0.0339	0.0476	6.7561	1.3391	0.0480	0.0559
	0.1	0.0636	0.1502	0.0198	4.3014	0.8390	0.0327	0.0446	5.4319	1.0530	0.0349	0.0441
	0.2	0.0650	0.1520	0.0187	4.6152	0.8293	0.0431	0.0555	4.2654	0.8006	0.0311	0.0360
	0.3	0.0644	0.1532	0.0222	5.3785	1.1189	0.0560	0.0698	8.1409	1.4558	0.0617	0.0649
1,000	0.0	0.0882	0.1748	0.0382	7.4041	1.6617	0.0570	0.1158	8.3973	1.8873	0.0662	0.1079
	0.1	0.0863	0.1774	0.0422	7.0925	1.5869	0.0601	0.1205	7.8927	1.7625	0.0603	0.1130
	0.2	0.0876	0.1794	0.0383	7.2156	1.6033	0.0666	0.1280	7.2707	1.6448	0.0597	0.1014
	0.3	0.0845	0.1780	0.0376	7.3396	1.6076	0.0843	0.1184	6.7575	1.5303	0.0737	0.0873
10,000	0.0	0.3110	0.4687	0.4139	98.2424	27.1424	0.5969	0.9334	101.9374	27.9375	0.6120	0.9167
	0.1	0.3135	0.4831	0.4214	93.1835	25.7640	0.6081	1.0010	95.6232	26.3330	0.6753	0.9977
	0.2	0.3172	0.4887	0.4117	87.8171	24.2369	0.6195	0.9264	89.6735	24.7393	0.6208	0.8778
	0.3	0.3167	0.4988	0.4279	81.4639	22.1943	0.6947	0.9465	80.7683	22.0125	0.5284	0.7187

S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

are generated from mean $m\sigma^2$ with a factor $m = \sqrt{p}$. Thus, outliers locate further from typical data as dimension increases. Prediction performance, variable selection performance, and computing time are summarized in Tables 3, 4, and 5, respectively. Table 4 shows that bisquare loss produces reliable results in presence of outliers. Note that Huber loss quickly deteriorates as contamination get

Table 4: Simulation 2 - average of 100 test RMSEs and its standard error (in parenthesis)

p	π	RMSEs										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
50	0.0	2.8572	2.9005	3.2608	2.8977	2.9604	3.2688	3.1108	3.0635	3.0740	3.5044	3.1815
		(0.0250)	(0.0280)	(0.0290)	(0.0280)	(0.0310)	(0.0300)	(0.0280)	(0.0330)	(0.0340)	(0.0370)	(0.0320)
	0.1	6.1176	6.1751	11.2281	3.1535	5.3629	4.5647	3.8234	3.0421	5.3962	3.4482	3.1631
		(0.0720)	(0.0540)	(0.2730)	(0.0340)	(0.0310)	(0.0950)	(0.0560)	(0.0350)	(0.0380)	(0.0400)	(0.0320)
	0.2	8.6024	8.4683	15.3106	4.1725	5.4820	13.4685	8.6459	3.0188	5.1744	3.3201	3.0847
		(0.1410)	(0.1230)	(0.3640)	(0.0670)	(0.0400)	(0.3700)	(0.2630)	(0.0290)	(0.0570)	(0.0340)	(0.0300)
	0.3	11.3820	11.2803	17.9280	10.3680	10.9391	18.7964	16.3489	6.3431	6.5475	9.3799	7.5900
		(0.2100)	(0.1990)	(0.4310)	(0.4600)	(0.4930)	(0.5100)	(0.3870)	(0.5120)	(0.5030)	(0.7030)	(0.5120)
100	0.0	5.3336	5.3388	5.3360	3.3972	3.5069	5.2086	3.6239	4.2170	4.3317	4.2089	3.4831
		(0.0330)	(0.0330)	(0.0660)	(0.0730)	(0.0760)	(0.0630)	(0.0310)	(0.1030)	(0.1040)	(0.0520)	(0.0300)
	0.1	7.0232	7.0159	41.7878	3.6306	5.3966	25.7004	7.7918	3.7066	5.4368	4.1659	3.5490
		(0.0680)	(0.0630)	(0.9690)	(0.0620)	(0.0330)	(0.5200)	(0.2380)	(0.0840)	(0.0350)	(0.0480)	(0.0350)
	0.2	10.7519	10.5992	55.5286	4.7298	10.4626	39.3374	21.2339	3.3009	5.3223	3.7344	3.3720
		(0.1520)	(0.1400)	(1.3470)	(0.0660)	(1.5880)	(0.7500)	(0.4130)	(0.0540)	(0.0460)	(0.0420)	(0.0420)
	0.3	15.0825	14.7932	58.5546	16.8676	82.2173	49.5735	28.1948	5.6270	5.6168	5.5368	5.4661
		(0.2360)	(0.2210)	(1.6150)	(1.0060)	(5.0710)	(1.0190)	(0.5150)	(0.7570)	(0.5940)	(0.6270)	(0.5240)
200	0.0	5.2792	5.2960	3.7408	3.3144	3.4933	3.7530	3.7157	3.9389	4.2246	3.7315	3.5408
		(0.0340)	(0.0330)	(0.0330)	(0.0430)	(0.0510)	(0.0330)	(0.0350)	(0.0840)	(0.0970)	(0.0340)	(0.0330)
	0.1	8.5121	8.4652	23.9495	3.9219	15.3873	22.8119	22.2143	3.5161	5.4437	3.6966	3.5413
		(0.1320)	(0.1270)	(0.5490)	(0.0540)	(1.0690)	(0.5020)	(0.4800)	(0.0560)	(0.0320)	(0.0340)	(0.0430)
	0.2	14.4405	14.1266	32.4718	5.4247	31.7276	31.4729	34.1979	3.5092	5.0911	3.6295	3.3470
		(0.2890)	(0.2620)	(0.7400)	(0.2220)	(0.9290)	(0.6900)	(0.7320)	(0.0460)	(0.0570)	(0.0420)	(0.0500)
	0.3	20.7929	20.6055	38.8953	30.3725	39.6322	38.5896	42.9984	3.7537	3.9634	3.6696	3.6053
		(0.4330)	(0.4680)	(0.8680)	(1.5450)	(0.9350)	(0.8480)	(0.9140)	(0.0530)	(0.0620)	(0.0460)	(0.0650)
400	0.0	4.0374	4.1494	3.5140	3.3626	3.5749	3.5171	3.3047	3.4149	3.6326	3.5210	3.2603
		(0.0590)	(0.0630)	(0.0310)	(0.0310)	(0.0300)	(0.0310)	(0.0350)	(0.0360)	(0.0400)	(0.0320)	(0.0350)
	0.1	11.6193	10.7873	24.3507	4.5725	25.0613	23.0480	28.2852	3.6127	4.9803	3.6144	3.1086
		(0.2530)	(0.1560)	(0.4800)	(0.0480)	(0.4560)	(0.3950)	(0.5340)	(0.0380)	(0.0580)	(0.0350)	(0.0360)
	0.2	20.4784	19.7779	34.8443	17.6653	35.4930	33.3502	38.8368	3.7503	4.2447	3.7797	3.0991
		(0.4600)	(0.4310)	(0.6920)	(1.4230)	(0.7130)	(0.6010)	(0.6690)	(0.0390)	(0.0630)	(0.0410)	(0.0450)
	0.3	29.3303	35.9054	43.7305	45.2124	44.3634	43.2399	49.8013	3.6527	4.0667	4.4179	3.5856
		(0.5730)	(1.1870)	(0.8180)	(0.8840)	(0.8480)	(0.7940)	(0.8940)	(0.0440)	(0.0410)	(0.0680)	(0.0560)

Best performer for each case is highlighted.
RMSE = root mean squared error; S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

larger, demonstrating that convex Huber loss suffers from severe outliers. In this simulation CV often produces best results, but adaptive lasso version of Bayesian approach is still comparable. Most of all, computing time in Table 6 demonstrates that the latter is more attractive than the former. In Table 5 we provides false rate of slope estimate by combining false negatives and false positives, instead of reporting separate false rates as in Table 2. Thus, false rate is the proportion of elements in $\hat{\boldsymbol{\beta}}$ that true zero is falsely claimed to zero and true nonzero falsely claimed zero. Table 5 demonstrates that robust adaptive lasso using bisquare outperforms CV in variable selection perspective, while they are comparable in prediction sense. Note that CV usually finds the best model which gives the best prediction, while robust adaptive lasso focuses on variable selection as well as a good fit to the data in the high-dimensional cases.

3.2. Real data example

We applied our methods to Boston housing data, available on http://lib.stat.cmu.edu/datasets/boston. The dataset contains medv (median value of owner-occupied homes in thousand dollars) as a response variable and additional 13 explanatory variables: crim (per capita crime rate by town), zn (proportion of residential land zoned for lots over 25,000 sq. ft.), indus (proportion of non-retail business acres per town), chas (Charles river dummy variable), nox (nitric oxides concentration, parts per 10 million), rm (average number of rooms per dwelling), age (proportion of owner-occupied units built prior to

Table 5: Simulation 2 - average of false rate

p	π	False rate										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
50	0.0	0.1799	0.0924	0.4782	0.1988	0.1020	0.4685	0.2069	0.2364	0.1354	0.4682	0.1908
	0.1	0.3944	0.4786	0.4984	0.1804	0.4980	0.4722	0.2281	0.2145	0.4975	0.4516	0.1630
	0.2	0.4421	0.4916	0.5009	0.1946	0.4575	0.4911	0.3479	0.1970	0.4482	0.4029	0.1081
	0.3	0.4662	0.4952	0.4935	0.4135	0.4225	0.4946	0.4506	0.2715	0.2514	0.3640	0.2406
100	0.0	0.4895	0.4950	0.4676	0.1187	0.0996	0.4679	0.1856	0.2708	0.2784	0.4318	0.1521
	0.1	0.4942	0.4991	0.4996	0.1303	0.5000	0.4993	0.2738	0.1948	0.5000	0.3838	0.1322
	0.2	0.4870	0.4980	0.4992	0.2338	0.4857	0.4983	0.4443	0.1421	0.4724	0.3037	0.0821
	0.3	0.4897	0.4984	0.4989	0.4552	0.4813	0.4964	0.4677	0.1648	0.1693	0.1867	0.1616
200	0.0	0.4541	0.4730	0.2427	0.0823	0.0733	0.2418	0.1006	0.1913	0.2379	0.2324	0.0831
	0.1	0.4979	0.4998	0.4821	0.1053	0.4871	0.4867	0.3887	0.1233	0.4980	0.1997	0.0618
	0.2	0.4918	0.4998	0.4767	0.3033	0.4894	0.4854	0.4598	0.1054	0.4016	0.1473	0.0464
	0.3	0.4979	0.4999	0.4823	0.4501	0.4768	0.4862	0.4700	0.1835	0.1265	0.0963	0.1066
400	0.0	0.1033	0.1433	0.1004	0.0587	0.1128	0.1003	0.0246	0.0670	0.1213	0.0998	0.0239
	0.1	0.4904	0.4998	0.4429	0.1829	0.4773	0.4643	0.4162	0.0838	0.3584	0.0849	0.0193
	0.2	0.4971	0.5008	0.4471	0.4114	0.4731	0.4850	0.4485	0.1153	0.1507	0.0820	0.0340
	0.3	0.4979	0.5083	0.4697	0.4725	0.4699	0.4922	0.4499	0.1472	0.1024	0.2116	0.1229

Best performer is highlighted.
S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

Table 6: Simulation 2 - average of 100 computing times (in seconds)

p	π	Computing time in seconds										
		S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
50	0.0	0.0781	0.1631	0.1968	7.2202	1.4134	0.3028	0.2668	11.9783	2.4702	0.3133	0.2881
	0.1	0.0845	0.1644	0.2198	10.5344	1.6158	0.3213	0.2936	9.6572	2.0031	0.2907	0.2721
	0.2	0.0869	0.1661	0.2261	13.6083	2.7533	0.3243	0.3530	6.6816	1.3554	0.2685	0.2193
	0.3	0.0883	0.1691	0.2215	11.8341	2.4003	0.3169	0.3347	11.0693	2.4290	0.2740	0.2165
100	0.0	0.0643	0.1601	0.9995	11.4249	2.4414	0.9879	0.9980	19.2696	4.2499	0.9882	0.9859
	0.1	0.0691	0.1602	0.9928	13.0385	2.5656	0.9893	1.0773	17.6033	3.8457	0.9813	0.9150
	0.2	0.0733	0.1638	0.9881	20.6759	4.4796	0.9997	1.0807	12.7885	2.7091	0.9815	0.7543
	0.3	0.0927	0.1848	0.9781	19.2542	6.6422	0.9739	1.0528	12.8800	2.8942	0.8831	0.5017
200	0.0	0.0606	0.1790	3.2983	24.8661	5.5037	3.2794	3.2700	33.0951	7.3042	3.2758	3.1974
	0.1	0.0689	0.1794	3.2139	38.4748	8.3108	3.1983	3.3806	31.8369	6.9513	3.1978	2.7560
	0.2	0.0869	0.1911	3.1753	47.9747	10.9628	3.1921	3.3649	27.6680	5.9634	3.1588	2.1737
	0.3	0.1179	0.2488	3.1582	44.5559	10.3108	3.1495	3.3499	26.9792	5.8738	2.9705	1.3073
400	0.0	0.0845	0.2213	13.0325	97.6626	22.3464	13.0279	9.2987	103.9795	23.3564	13.0259	8.7507
	0.1	0.1272	0.2878	13.1755	165.5312	38.08582	13.1628	13.5352	113.4370	25.2585	13.1059	7.3319
	0.2	0.1570	0.3745	13.0621	172.9576	39.19266	13.0612	13.4129	115.9462	25.6933	12.7634	5.0736
	0.3	0.1957	0.5702	12.9983	157.6594	34.91554	12.9769	13.3564	120.2724	26.2045	9.5168	4.6150

S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

1940), dis (weighted distances to five Boston employment centers), rad (index of accessibility to radial highways), tax (full-valued property-tax rate per 10,000 dollars), ptratio (pupil-teacher ratio by town), black ($1000 \times (B_k - 0.63)^2$ where B_k is the proportion of blacks by town), and lstat (percent of lower status of the population). There are 506 observations in the dataset. We split the dataset into two parts: the first 300 observations were regarded as a training dataset and the remaining 206 observations were treated as a test dataset. To see the effect of outliers on the methods we considered, we randomly selected 30 observations from the training dataset and changed their response variables by values having large residuals (five times larger than residual standard deviation).

Table 7: Boston housing data - estimated coefficients and test RMSEs without outliers

Variables	S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
(Intercept)	−13.914	−18.325	−13.625	−16.707	−18.326	−16.273	−15.874	−15.764	−18.325	−15.014	−16.377
crim	0.955	0.000	1.073	0.000	0.101	0.350	0.000	0.000	0.000	0.119	0.000
zn	0.011	0.000	0.015	0.002	0.000	0.014	0.012	0.011	0.000	0.019	0.018
indus	0.006	0.000	0.019	0.000	0.000	−0.015	0.000	−0.023	0.000	−0.037	0.000
chas	0.607	0.413	0.579	0.500	0.506	0.525	0.000	0.468	0.413	0.362	0.000
nox	−6.453	0.000	−6.933	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
rm	9.160	9.253	9.137	9.045	9.278	8.991	9.078	8.908	9.253	8.841	9.060
age	−0.046	−0.035	−0.048	−0.042	−0.039	−0.053	−0.053	−0.050	−0.035	−0.057	−0.059
dis	−0.919	−0.529	−0.977	−0.562	−0.584	−0.822	−0.814	−0.700	−0.529	−0.870	−0.826
rad	0.117	0.000	0.162	0.049	0.000	0.176	0.181	0.158	0.000	0.240	0.257
tax	−0.013	−0.010	−0.015	−0.011	−0.011	−0.014	−0.014	−0.013	−0.010	−0.015	−0.015
ptratio	−0.632	−0.617	−0.628	−0.615	−0.620	−0.575	−0.616	−0.577	−0.617	−0.557	−0.576
black	0.016	0.011	0.017	0.010	0.012	0.013	0.012	0.011	0.011	0.012	0.012
lstat	−0.110	−0.114	−0.113	−0.107	−0.114	−0.101	−0.084	−0.070	−0.114	−0.060	−0.041
Test RMSE	15.824	7.984	17.472	7.849	8.113	9.761	7.863	7.808	7.984	8.394	8.042

RMSE = root mean squared error; S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

Table 8: Boston housing data - estimated coefficients and test RMSEs after outlier inclusion

Variables	S.CV	S.BIC	S.L	H.CV	H.BIC	H.L	H.A	B.CV	B.BIC	B.L	B.A
(Intercept)	13.198	46.611	61.568	−15.959	12.883	−14.087	−12.600	−15.960	23.007	−14.195	−14.963
crim	0.000	0.000	−9.169	0.000	−0.603	0.070	0.000	0.000	−3.529	0.000	0.000
zn	0.000	0.000	0.027	0.007	0.000	0.020	0.020	0.015	0.000	0.024	0.024
indus	0.000	0.000	−0.481	−0.020	−0.278	−0.035	0.000	−0.020	−0.360	−0.045	0.000
chas	0.000	0.000	−4.543	0.312	0.000	0.327	0.000	0.357	−0.324	0.147	0.000
nox	0.000	0.000	−15.481	0.000	0.000	−0.714	−3.601	0.000	0.000	0.000	−1.502
rm	7.873	0.000	8.700	8.975	9.339	8.877	8.960	8.678	9.825	8.502	8.728
age	0.000	0.000	0.195	−0.042	0.000	−0.051	−0.050	−0.053	0.008	−0.062	−0.063
dis	0.000	0.000	0.398	−0.610	0.000	−0.892	−0.921	−0.746	0.000	−0.985	−0.960
rad	0.000	0.000	1.882	0.112	1.214	0.260	0.284	0.189	1.446	0.301	0.321
tax	0.000	0.000	0.084	−0.007	0.036	−0.010	−0.010	−0.010	0.055	−0.013	−0.013
ptratio	−0.675	0.000	−3.067	−0.652	−1.700	−0.645	−0.692	−0.558	−2.091	−0.536	−0.563
black	0.000	0.000	−0.101	0.010	−0.007	0.012	0.012	0.013	−0.038	0.015	0.014
lstat	−0.492	0.000	−1.028	−0.129	−0.722	−0.117	−0.102	−0.085	−0.699	−0.062	−0.044
Test RMSE	23.183	29.441	98.664	7.885	38.727	8.608	8.407	7.798	42.804	8.038	8.249

RMSE = root mean squared error; S = square; H = Huber; B = bisquare; CV = cross validation; BIC = Bayes information criterion; L = robust lasso; A = robust adaptive lasso.

Tables 7 and 8 present the coefficient estimates and test RNSE from the fit of the training dataset without and with outliers, respectively. Table 7, in the case of absence of outliers, shows that all methods seem to produce the similar coefficient estimates and test RMSE. However, after outlier inclusion, square loss severely deteriorates, while Huber and biweight losses are robust against outlier. Interestingly, we observed that BIC is not stable for outlier in this example.

4. Conclusion and remarks

In this study, we propose an alternative way for CV in robust lasso and robust adaptive lasso using marginal prior on coefficients under Bayesian framework. Throughout simulation studies, we demonstrate that our proposals are competitive to or better than CV in prediction, variable selection, and computing time perspectives.

In this study we limit this idea only to robust regression; however, it can be further applied to various penalized predictive models. Our approach becomes especially valuable when CV is not appropriate. For example, in classification problem, training data may suffer from a mislabeling class label called label-noise (Lee *et al.*, 2016). In such case, hold-out procedure like CV is not successful because the validation set also have a label-nose; in addition, a cross-validation score from a validation set having label-noise is not reliable for model selection. One can extend the same idea in this study to some complicated problems that include label-nose, where CV is not available. We leave this direction for future research.

References

Bishop CM (2006). *Pattern Recognition and Machine Learning*, Springer, New York.

Buntine WL and Weigend AS (1991). Bayesian back-propagation, *Complex Systems*, **5**, 603–643.

Chang L, Roberts S, and Welsh A (2017). Robust lasso regression using Tukey's biweight criterion, *Technometrics*, from: https://dx.doi.org/10.1080/00401706.2017.1305299

El Ghaoui L and Lebret H (1997). Robust solutions to least-squares problems with uncertain data, *SIAM Journal on Matrix Analysis and Applications*, **18**, 1035–1064.

Figueiredo MAT (2003). Adaptive sparseness for supervised learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**, 1150-1159.

Hastie T, Tibshirani R, and Friedman JH (2001). *The Elements of Statistical Learning: Data Mining Inference, and Prediction*, Springer, New York.

Koenker R (2005). *Quantile Regression*, Cambridge University Press, Cambridge.

Lambert-Lacroix S and Zwald L (2011). Robust regression through the Huber's criterion and adaptive lasso penalty, *Electronic Journal of Statistics*, **5**, 1015–1053.

Lee A, Caron F, Doucet A, and Holmes C (2010). *A hierarchical Bayesian framework for constructing sparsity-inducing priors* (Technical report), University of Oxford, Oxford.

Lee S, Shin H, and Lee SH (2016). Label-noise resistant logistic regression for functional data classification with an application to Alzheimer's disease study, *Biometrics*, **72**, 1325–1335.

MacKay DJC (1995). Probable networks and plausible predictions: a review of practical Bayesian methods for supervised neural networks, *Network: Computation in Neural Systems*, **6**, 469–505.

Maronna RA, Martin RD, and Yohai VJ (2006). *Robust Statistics: Theory and Methods*, Wiley, Chichester.

Murphy KP (2012). *Machine Learning: A Probabilistic Perspective*, The MIT Press, Cambridge.

Owen AB (2006). *A robust hybrid of lasso and ridge regression* (Technical report), Stanford University, Stanford.

Park T and Casella G (2008). The Bayesian lasso, *Journal of the American Statistical Association*, **103**, 681–686.

Tipping ME (2001). Sparse Bayesian learning and the relevance vector machine, *Journal of Machine Learning Research*, **1**, 211–244.

West M (1987). On scale mixtures of normal distributions, *Biometrika*, **74**, 646–648.

Zou H (2006). The adaptive lasso and its oracle properties, *Journal of the American Statistical Association*, **101**, 1418–1429.

Zou H and Li R (2008). One-step sparse estimates in nonconcave penalized likelihood models, *The Annals of Statistics*, **36**, 1509–1533.

Received July 21, 2017; Revised September 20, 2017; Accepted October 14, 2017