Robust sparse covariance-regularized regression for high-dimensional data with casewise and cellwise outliers

Maggie Liu
Department of Statistics
University of British Columbia
yitong.liu@stat.ubc.ca

Gabriela Cohen Freue
Department of Statistics
University of British Columbia
gcohen@stat.ubc.ca

Abstract

Modern biomedical datasets, such as those found in genomic and proteomic studies, often involve a large number of predictor variables relative to the number of observations, pointing to the need for statistical methods specifically designed to handle high-dimensional data. In particular, for a regression task, regularized methods are needed to select a sparse model, that is, one that uses only a subset of the large number of features available to predict a response. The presence of outliers in the data further complicates this task. Many existing robust and sparse regression methods are computationally expensive when the dimensionality of the data is high. Furthermore, most of these previously developed methods were developed under the assumption that outliers occur casewise, which is not always a realistic assumption in high-dimensional settings. We propose a sparse and robust regression method for high-dimensional data that is based on regularized precision matrix estimation. Our method can handle both casewise and cellwise outliers in low- and high-dimensional settings. Through simulation studies, we also compare our method to existing sparse and robust methods by evaluating computational efficiency, prediction performance, and variable selection capabilities.

Keywords: Robust regression; Cellwise outliers; Casewise outliers; High-dimensional data.

1. Introduction

Modern applications of statistical modeling frequently involve high-dimensional data, wherein the number of variables exceeds the sample size ($p > n$). Recent technological advancements have facilitated the collection of these high-dimensional datasets. For example, gene expression assays can measure tens of thousands of genes to identify biomarker genes for a disease outcome. Typically, only a small subset of measured genes are expected to be related to the outcome, so the true underlying model that we wish to estimate is sparse. In high-dimensional settings, outliers are also common, further compounding the challenge of achieving a high prediction accuracy while maintaining model sparsity.

Several robust sparse regression estimators have been proposed in the literature, such as Sparse Least Trimmed Squares (SparseLTS) (Alfons et al. 2013), the MM-LASSO (Smucler and Yohai 2017), and the Penalized Elastic Net S-Estimators (PENSE) (Cohen Freue et al. 2018). However, most existing methods were designed and tested only on casewise outliers. In high-dimensional settings, a large proportion of observations is likely to contain outlying data values in at least one predictor variable, but the proportion of predictors in a single observation that are outlying is typically small. Hence, it is more realistic to characterize outliers using the cellwise outlier paradigm, first introduced by Alqallaf et al. (2009). This cellwise outlier paradigm assumes that individual entries, rather than rows, of the data are outlying, while the remaining entries in the row are still useful for estimation and prediction.

The development of regression methods that are robust to cellwise outliers is a new and ongoing area of research. Öllerer et al. (2016) develop an S-regression method that is robust to cellwise contamination called the Shooting S. However, the Shooting S is not suitable for high-dimensional data. Bottmer et al. (2022) propose the Sparse Shooting S estimator that is robust to cellwise outliers, is feasible in $p > n$ settings, and estimates sparse coefficients. Filzmoser et al. (2020) propose a cellwise robust M (CRM) regression estimator that allows for the estimation of regression coefficients in the presence of cellwise outliers while also detecting the cells that are deviating with respect to the linear model. While the CRM method can handle cellwise outliers, it breaks down when more than 50% of the cases contain outliers, which is a common situation in the presence of cellwise contamination. Other developments in cellwise-robust estimation lie mostly in outlier detection and cellwise-robust estimation of correlations, covariance matrices, and precision matrices. Öllerer and Croux (2015) and Croux and Öllerer (2016) use rank correlations to estimate a robust precision matrix. Raymaekers and Rousseeuw (2021b) use the properties of product moments together with a transformation similar to that seen in Hampel et al. (1981) to estimate a robust correlation and covariance matrix that is positive semi-definite and satisfies other desirable statistical properties. The detecting deviating cells (DDC) method (Rousseeuw and Bossche 2018) detects and imputes deviating cells or missing data values in a multivariate dataset, and it does so by accounting for correlations between pairs of variables. DDC also uses the Hampel-like transformation developed by Raymaekers and Rousseeuw (2021b) to compute pairwise correlations efficiently, resulting in a fast algorithm for detecting and imputing outliers in high dimensions.

Accounting for the conditional independence relationships between predictor variables in a linear regression model can also improve prediction performance and unveil interrelationships between the variables that are not seen in existing regularized methods. The Scout method of Witten and Tibshirani (2009) is a new approach for regularizing linear regression that shrinks the inverse covariance matrix of the predictor variables. It aims to uncover pairs of variables in a multivariate Normal model that are conditionally independent while also estimating shrunken or sparse regression coefficients.

While the Scout method has been shown to have superior prediction performance compared to other regularized methods, it relies on non-robust sample covariance estimates, and is therefore not robust to outliers.

We propose a new generalized robust and sparse covariance-regularized regression framework called RobScout. It combines the DDC method with covariance-regularized coefficient estimation by the Scout method, and is robust to cellwise and casewise outliers. RobScout is also computationally efficient in high-dimensional settings, and is adaptable for realistic correlation structures between predictor variables.

2. The RobScout Method

Let $\mathbf{X}$ denote an $n \times p$ matrix of predictor variables, and let $\mathbf{y}$ be a length-n response vector. For each observation i, with $i = 1, \ldots, n$, let $\mathbf{x}_i$ be the length-p vector denoting the ith row of $\mathbf{X}$, and let y_i denote the ith entry of $\mathbf{y}$.

We consider a regression setup where we aim to estimate coefficients $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^{\mathsf{T}}$ from the following model:

$$y_i = \mathbf{x}_i^{\mathsf{T}} \boldsymbol{\beta} + e_i \tag{1}$$

where the e_i's are noise terms. Least Squares regression estimates $\boldsymbol{\beta}$ by minimizing the sum of squared residuals:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y_i - \mathbf{x}_i^{\mathsf{T}} \boldsymbol{\beta} \right)^2, \tag{2}$$

and when $n > p$, its closed form can be written as $\hat{\beta} = (\mathbf{X^T X})^{-1} \mathbf{X^T y}$. However, when $p > n$, $\mathbf{X^T X}$ is not invertible, so the least squares estimate is not unique. The presence of outliers further complicates the issue. In these high-dimensional cases, regularization is required to compute an estimate of the coefficients. Typically, regularized regression methods add a penalty to the sum of squared residuals in Equation 2; two examples are LASSO (Tibshirani 1996) and Ridge regression (Hoerl and Kennard 1970). However, another way to introduce parsimony into the estimated model is by covariance-regularization.

2.1. Overview of Covariance-Regularized Regression

Witten and Tibshirani (2009) introduce a covariance-regularized regression framework that penalizes the log-likelihood of the data under Normality assumptions. The method, called the Scout procedure, takes advantage of an alternative way of writing the closed form solution of Equation 2.

To outline the Scout procedure, we introduce some more notation. Let $\tilde{\mathbf{X}} = (\mathbf{X} \quad \mathbf{y})$ be an $n \times (p+1)$ matrix, where the columns of $\tilde{\mathbf{X}}$ are assumed to be centered and scaled. Let $\boldsymbol{\Sigma}$ denote the population covariance matrix of $\tilde{\mathbf{X}}$, $\mathbf{S}$ denote the sample covariance matrix of $\tilde{\mathbf{X}}$, and $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$ denote the population inverse covariance (population precision matrix) of $\tilde{\mathbf{X}}$. We write $\mathbf{S}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\Theta}$ as block matrices as follows:

$$\mathbf{S} = \begin{pmatrix} \mathbf{S}_{XX} & \mathbf{S}_{Xy} \\ \mathbf{S}_{Xy}^{\mathsf{T}} & S_{yy} \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{XX} & \boldsymbol{\Sigma}_{Xy} \\ \boldsymbol{\Sigma}_{Xy}^{\mathsf{T}} & \Sigma_{yy} \end{pmatrix}, \quad \text{and} \quad \boldsymbol{\Theta} = \begin{pmatrix} \boldsymbol{\Theta}_{XX} & \boldsymbol{\Theta}_{Xy} \\ \boldsymbol{\Theta}_{Xy}^{\mathsf{T}} & \Theta_{yy} \end{pmatrix}. \tag{3}$$

One can rewrite the closed form for Least Squares solution $\hat{\beta} = (\mathbf{X^T X})^{-1} \mathbf{X^T y}$ using only components of a precsion matrix estimate:

$$\hat{\boldsymbol{\beta}} = -\hat{\boldsymbol{\Theta}}_{Xy} / \hat{\Theta}_{yy}. \tag{4}$$

This follows from the result of inverting a block matrix, as seen in (Mardia et al. 1979). Hence, to estimate regression coefficients, it is sufficient to estimate a precision matrix for $\tilde{\mathbf{X}}$. For high-dimensional data, we can estimate a shrunken precision matrix, and consequently, shrunken coefficients, by penalizing the entries of $\mathbf{\Theta}$.

The Scout procedure by Witten and Tibshirani (2009) estimates a regularized precision matrix using L_p penalties, and uses it to compute $\hat{\boldsymbol{\beta}}$ by Equation 4 in a procedure with two regularization steps. We outline the procedure below:

1. Estimate a shrunken $\mathbf{\Theta}_{XX}$:

$$\hat{\mathbf{\Theta}}_{XX} = \underset{\mathbf{\Theta}_{XX}}{\operatorname{argmax}} \left\{ \log\left(\det\left(\mathbf{\Theta}_{XX}\right)\right) - \operatorname{tr}\left(\mathbf{S}_{XX}\mathbf{\Theta}_{XX}\right) - \lambda_1 \left\|\mathbf{\Theta}_{XX}\right\|_{p_1} \right\}. \qquad (5)$$

2. Estimate a shrunken $\mathbf{\Theta}$ conditional on the $\hat{\mathbf{\Theta}}_{XX}$ obtained in the first step:

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmax}} \left\{ \log\left(\det\left(\mathbf{\Theta}\right)\right) - \operatorname{tr}\left(\mathbf{S}\mathbf{\Theta}\right) - \frac{\lambda_2}{2} \left\|\mathbf{\Theta}\right\|_{p_2} \right\}, \qquad (6)$$

   where the top-left $p \times p$ block matrix of $\hat{\mathbf{\Theta}}$ is constrained to equal the $\hat{\mathbf{\Theta}}_{XX}$ solution from Step 1.

3. Compute $\hat{\boldsymbol{\beta}} = -\hat{\mathbf{\Theta}}_{Xy}/\hat{\Theta}_{yy}$.

4. Scale the coefficients by computing $\hat{\boldsymbol{\beta}}^{\star} = c\hat{\boldsymbol{\beta}}$ where c is the coefficient for the regression of $\mathbf{y}$ onto $\mathbf{X}\hat{\boldsymbol{\beta}}$.

This procedure is denoted by Scout(p_1, p_2), where p_1 and p_2 are the L_p penalties applied in the first and second steps, respectively. A $\bullet$ is used in place of p_1 and/or p_2 when $\lambda_1 = 0$ or $\lambda_2 = 0$. For example, Scout$(\bullet,\ 1)$ denotes the Scout procedure with no regularization in the first step, and an L_1 penalty in the second step.

In this paper, we are primarily interested in special cases where in Step 1, we have one of $\{\, p_1 = 1, p_1 = 2, \lambda_1 = 0 \,\}$ and in Step 2, we have one of $\{\, p_2 = 1, \lambda_2 = 0 \,\}$.

2.2. Robust Covariance-Regularized Regression

The Scout procedure relies on non-robust estimates of the covariance and hence precision matrices, so its coefficient estimates are negatively affected by outliers of any kind. Rousseeuw and Bossche (2018) recently proposed a method for detecting and imputing cellwise outliers in multivariate data analysis problems called detecting deviating cells (DDC). DDC computes pairwise correlations between variables in the data to predict the value of each cell, and flags cells with predicted values that deviate from their actual values. It is also the first method that can detect cellwise outliers that are not necessarily marginally outlying.

We propose RobScout, a robust extension of the Scout method. We follow the notation introduced by Witten and Tibshirani (2009) and use RobScout(p_1, p_2) to denote RobScout with L_p penalties p_1 and p_2, respectively, in the first and second precision matrix estimation steps. We outline the steps of RobScout(p_1, p_2) below:

1. **Impute outliers with DDC**: Apply the DDC algorithm to $\tilde{\mathbf{X}}$ to detect and impute cellwise outliers, as described by Rousseeuw and Bossche (2018). Let $\tilde{\mathbf{X}}_{\text{imp}}$ denote the resulting output, which is the destandardized matrix containing the imputed values of the outliers in $\tilde{\mathbf{X}}$. Note that we include the response vector in the input to DDC in order to also detect outliers in the subspace of the response paired with each predictor variable.

2. **Standardize the data**: Standardize the columns of $\tilde{\mathbf{X}}_{\text{imp}}$ using their means and standard deviations to obtain $\tilde{\mathbf{Z}}_{\text{imp}}$, which has entries

$$\tilde{z}_{ij} = \frac{\tilde{x}_{ij} - \text{mean}(\tilde{\mathbf{x}}_j)}{\text{sd}(\tilde{\mathbf{x}}_j)},$$

where $j = 1, \ldots, p+1$ and $\tilde{\mathbf{x}}_j$ denotes the jth column of $\tilde{\mathbf{X}}_{\text{imp}}$. Classical mean and standard deviation estimators are used to standardize $\tilde{\mathbf{X}}_{\text{imp}}$ since $\tilde{\mathbf{X}}_{\text{imp}}$ no longer contains the outliers detected by DDC.

3. **Estimate regression coefficients using the Scout procedure**: Apply the Scout procedure as described by Witten and Tibshirani (2009) to estimate the coefficients the regression of $\tilde{\mathbf{x}}_{p+1}$ (the imputed response vector) on $(\tilde{\mathbf{x}}_1 \cdots \tilde{\mathbf{x}}_p)$ (the imputed predictor matrix).

4. **Estimate the intercept term**: Compute $\hat{\beta}_0$ as follows:

$$\text{mean}(\tilde{\mathbf{x}}_{p+1}) - \text{sd}(\tilde{\mathbf{x}}_{p+1}) \sum_{j=1}^{p} \frac{\text{mean}\left(\tilde{\mathbf{x}}_j\right)}{\text{sd}(\tilde{\mathbf{x}}_j)} \hat{\beta}_j,$$

where $\tilde{\mathbf{x}}_j$ is the jth column of $\tilde{\mathbf{X}}_{\text{imp}}$.

3. Optimization Algorithm

The Scout procedure involves two regularization parameters, but an efficient optimization algorithm for choosing the best pair of parameters is not currently implemented. An obvious selection method is to cross-validate over a grid of $n_{\lambda_1} \times n_{\lambda_2}$ regularization parameters, where n_{λ_1} and n_{λ_2} denote the length of the path for λ_1 and λ_2, respsectively. However, optimizing over a grid would be computationally very slow, especially when $p > n$ or when $p_1 = 1$. In this section, we propose two efficient methods by which the penalty parameters can be chosen, depending on the penalty function on the precision matrix in each step.

3.1. Regularization Paths for Scout

First, consider the case when $p_1 = 1$. Then Step 1 of the Scout procedure as shown in Section 2.1 reduces to estimating $\mathbf{\Theta}_{XX}$ by the GLASSO. Banerjee et al. (2008) show that for sufficiently large λ_1, $\hat{\mathbf{\Theta}}_{XX}$ is a diagonal matrix with elements $1/\left(\lambda_1 + \mathbf{x}_i^{\mathsf{T}}\mathbf{x}_i\right)$. In particular, $\hat{\mathbf{\Theta}}_{XX}$ is a diagonal matrix when $\lambda_1 \geq \left|\mathbf{x}_i^{\mathsf{T}}\mathbf{x}_j\right|$ for all $i \neq j$, i.e., when

$\lambda_1 \geq \left\|\mathbf{X}^\mathsf{T}\mathbf{X}\right\|_\infty$. Hence, we define $\lambda_{1\max}$ to be the smallest value of λ_1 that shrinks all off-diagonal elements of $\hat{\mathbf{\Theta}}_{XX}$ to zero. Empirically, the lower bound for $\lambda_{1\,\max}$ is much smaller than $\left\|\mathbf{X}^\mathsf{T}\mathbf{X}\right\|_\infty$, so we implement a descending search algorithm for finding a smaller $\lambda_{1\,\max}$ that reduces the search space by 20% on each iteration for at most 100 iterations.

Now, consider the case when $p_1 = 2$. Witten and Tibshirani (2009) show that the first step in the Scout has a closed form solution. Let $\mathbf{U}_{n\times p}\mathbf{D}_{p\times p}\mathbf{V}^\mathsf{T}_{p\times p}$ be the singular value decomposition of $\mathbf{X}$, and let d_i be the diagonal entries of $\mathbf{D}$ with $d_1 \geq d_2 \geq \cdots \geq d_r > d_{r+1} = \cdots = d_p = 0$ where $r = \operatorname{rank}(\mathbf{X}) \leq \min(n, p)$. Then Step 1 in the Scout procedure is equivalent to solving

$$\hat{\mathbf{\Theta}}_{XX}^{-1} - 2\lambda_1\hat{\mathbf{\Theta}}_{XX} = \mathbf{X}^\mathsf{T}\mathbf{X}. \tag{7}$$

Equation 7 has the following closed form solution:

$$\hat{\mathbf{\Theta}}_{XX}^{-1} = \mathbf{V}\left(\mathbf{D}^2 + \tilde{\mathbf{D}}^2\right)\mathbf{V}^\mathsf{T}, \tag{8}$$

where $\mathbf{D}$ is the diagonal matrix from the singular value decomposition of $\mathbf{X}$, and $\tilde{\mathbf{D}}^2$ is a $p \times p$ diagonal matrix with ith diagonal entry equal to $\frac{1}{2}\left\{-d_i^2 + \sqrt{d_i^4 + 8\lambda_1}\right\}$. Now, since $\hat{\mathbf{\Theta}}_{XX}^{-1}$ obtained in Equation 8 cannot be shrunken to a diagonal matrix by picking a sufficiently large λ_1, we consider $\lambda_{1\,\max}$ in the case of $p_1 = 2$ to be the same $\lambda_{1\,\max}$ as in $p_1 = 1$.

Next, we consider the case when $p_2 = 1$. Since the second regularization step in the Scout depends on the first, we define $\lambda_{2\,\max}$ for a particular value of λ_1 that results in a particular estimate $\hat{\mathbf{\Theta}}_{XX}$ from the first step in the Scout procedure. Setting $p_2 = 1$ corresponds to applying a LASSO penalty on the last column and last row of $\mathbf{\Theta}$, and results in a sparse $\hat{\boldsymbol{\beta}}$ estimate. Hence, we define $\lambda_{2\,\max}$ to be the smallest value of λ_2 that shrinks all entries of $\hat{\boldsymbol{\beta}}$ to zero. Witten and Tibshirani (2009) show that when $p_1 = 1$, $\lambda_1 = \lambda_{1\,\max}$, and $p_2 = 1$, the coefficient estimates are given by

$$\hat{\beta}_i = \frac{1}{\lambda_1 + 1}\operatorname{sign}\left(\mathbf{x}_i^\mathsf{T}\mathbf{y}\right)\max\left(0, \left|\mathbf{x}_i^\mathsf{T}\mathbf{y}\right| - \frac{\lambda_2}{2}\right). \tag{9}$$

In this situation, we can see that the smallest value of λ_2 that shrinks $\hat{\beta}_i$ to zero for all $i = 1, \ldots, p$ is the λ_2 value such that $\left|\mathbf{x}_i^\mathsf{T}\mathbf{y}\right| - \frac{\lambda_2}{2} = 0$ for all $i = 1, \ldots, p$, which is $\lambda_{2\,\max} = 2\left\|\mathbf{X}^\mathsf{T}\mathbf{Y}\right\|_\infty$. However, since the $\lambda_{2\,\max}$ value depends on the value of λ_1, we proceed by the same descending search procedure for empirically computing a smaller $\lambda_{2\,\max}$ value as we did with $\lambda_{1\,\max}$ when $p_1 = 1$.

When performing cross-validation, we consider a logarithmically-spaced sequence of λ_1 values from $0.1\lambda_{1\,\max}$ to $\lambda_{1\,\max}$. For λ_2, we follow the path selection strategy of Friedman et al. (2010) and consider a sequence of logarithmically-spaced values from $0.001\lambda_{2\,\max}$ to $\lambda_{2\,\max}$. Each sequence is chosen to be length 100.

3.2. A Stepwise Optimization Approach for Scout(1,1)

When $p_1 = 1$, Step 1 of the Scout procedure is equivalent to the GLASSO method for precision matrix estimation. Yuan and Lin (2007) propose a criterion like the Bayesian information criterion (BIC) for sparse graphical model selection, given in Definition 1. The chosen penalty parameter is then the one that minimizes the given BIC-like criteria.

Definition 1. *Let $\mathbf{\Sigma}$ be a covariance matrix, and let $\hat{\mathbf{\Theta}}(\lambda)$ be a precision matrix estimate for $\mathbf{\Sigma}^{-1}$ where the value of the penalty is λ. Let $\hat{\theta}_{ij}$ denote the entry of $\hat{\mathbf{\Theta}}\left(\lambda\right)$ at row i and column j for $i, j = 1, \dots, p$. Let $\hat{\mathbf{\Sigma}}$ denote the maximum likelihood estimate for $\mathbf{\Sigma}$. Suppose $\hat{\mathbf{\Sigma}}$ and $\hat{\mathbf{\Theta}}(\lambda)$ were both estimated from a sample of size n. Then the Bayesian Information Criterion (BIC) for Graphical Model Selection is*

$$BIC(\lambda) = -\log\left(\det\left(\hat{\mathbf{\Theta}}(\lambda)\right)\right) + tr\left(\hat{\mathbf{\Theta}}\left(\lambda\right)\hat{\mathbf{\Sigma}}\right) + \frac{\log n}{n}\sum_{i \leq j}\hat{e}_{ij}(\lambda), \tag{10}$$

where

$$\hat{e}_{ij} = \begin{cases} 0 & \text{if } \hat{\theta}_{ij} = 0 \\ 1 & \text{otherwise} \end{cases}. \tag{11}$$

When $p_1 = 1$, instead of optimizing over a grid of $n_{\lambda_1} \times n_{\lambda_2}$, we proceed in a stepwise manner. We first select the λ_1 value which minimizes the BIC of $\hat{\mathbf{\Theta}}(\lambda_1)$. We then fix that value of $\hat{\mathbf{\Theta}}$ and select the λ_2 value which minimizes the cross-validation error. This stepwise approach is linear rather than quadratic in the number of $(n_{\lambda_1}, n_{\lambda_2})$ pairs over which it needs to search; it requires a total of $n_{\lambda_1} + n_{\lambda_2}$ model fits, whereas a full grid search would require $n_{\lambda_1} \times n_{\lambda_2}$ model fits. The full details of the stepwise approach are described in Algorithm 1.

Algorithm 1 Stepwise approach for selecting optimal λ_1 and λ_2 in cross-validation when $p_1 = 1$ and $p_2 = 1$

```
1: Compute $\lambda_1$ sequence as described in Section 3.1.
2: Select the model $\hat{\mathbf{\Theta}}(\lambda_1)$ that minimized BIC$(\lambda_1)$.
3: Using the $\hat{\mathbf{\Theta}}(\lambda_1)$ estimate obtained in Step 2, compute a sequence of $\lambda_2$ values as
   described in Section 3.1.
4: Compute the $\hat{\boldsymbol{\beta}}$ estimate for each value of $\lambda_2$ and pick the one that minimizes the
   cross-validation error.
```

3.3. An Alternating Optimization Approach for Scout(2,1)

When $p_1 = 2$, we have seen that the closed form solution for $\mathbf{\Theta}_{XX}$ satisfies Equation 7, and the closed form for $\mathbf{\Theta}_{XX}^{-1}$ is given by Equation 8. However, it is not guaranteed that the right-hand side of Equation 8 is invertible, so we may not be able to select λ_1 by minimizing the BIC of $\hat{\mathbf{\Theta}}_{XX}$ as in the previous section. Instead, we propose an alternating algorithm that is efficient, and empirically found to converge in many fewer steps than $n_{\lambda_1} + n_{\lambda_2}$. The algorithm alternates between searching for a λ_1 value for a fixed λ_2, and searching for a λ_2 value for a fixed λ_1, with the goal of minimizing the cross-validation prediction error based on the resulting $\hat{\boldsymbol{\beta}}$ estimate for each pair of λ_1

and λ_2 values considered. The full details of the algorithm are described in Algorithm 2.

Algorithm 2 Alternating optimization approach for selecting optimal λ_1 and λ_2 in cross-validation when $p_1 = 2$ and $p_1 = 1$

```
 1: Compute $\lambda_1$ sequence as described in Section 3.1.
 2: Initialize $\lambda_1 \leftarrow \lambda_{1\,\max}$
 3: Initialize diff $\leftarrow$ 1, n_iter $\leftarrow$ 0
 4: while (diff $>$ tol) OR (n_iter $\geq$ max_iter) do
 5:     Compute $\hat{\boldsymbol{\Theta}}_{XX}$ for $\lambda_1$
 6:     Compute $\lambda_2$ sequence based on $\hat{\boldsymbol{\Theta}}_{XX}$
 7:     Compute $\hat{\boldsymbol{\beta}}$ based on current $\lambda_2$ and $\lambda_2$
 8:     Set $\lambda_2$ to be the one minimizing the CV error based on $\hat{\boldsymbol{\beta}}$
 9:     Update n_iter $\leftarrow$ n_iter $+1$
10:     Find $\lambda_1$ minimizing the CV error with $\lambda_2$ fixed to be the value from line 8
11:     Compute $\hat{\boldsymbol{\beta}}$ using $\lambda_1$ and $\lambda_2$ from line 10
12:     Update diff $\leftarrow$ |current CV error $-$ previous CV error|
13:     Update n_iter $\leftarrow$ n_iter $+1$
    return $\lambda_1$, $\lambda_2$, $\hat{\boldsymbol{\beta}}$
```

4. Simulation Studies

4.1. Estimators Considered

We consider robust and non-robust estimators in our simulation study. Table 1 summarizes the estimators we consider. We also include the Oracle estimator in all settings, which uses the true coefficient values for that setting. In settings where $n > p$, we also include the OLS estimator.

Robust	**Non-robust**
RobScout(•,1)	Scout(•,1)
RobScout(1,1)	Scout(1,1)
RobScout(2,1)	Scout(2,1)
SparseLTS	LASSO
PENSE-LASSO	EN(0.5)

Table 1: Robust and non-robust methods under comparison in simulation settings. PENSE-LASSO is the Penalized Elastic Net S-Estimator (Cohen Freue et al. 2018) with LASSO Penalty, SparseLTS is the Sparse Least Trimmed Squares estimator (Alfons et al. 2013), and EN(0.5) is the Elastic Net estimator with mixing parameter 0.5.

For PENSE-LASSO, we use the R implementation available through the package `pense` (Kepplinger et al. 2023), and we use the robust τ-scale (Yohai and Zamar 1988) to measure prediction performance in 5-fold cross-validation to select the penalty parameter.

For SparseLTS, we use the R implementation available through the package `robustHD` (Alfons 2021), and use the modified Bayesian Information Criterion (BIC) described in Alfons et al. (2013) to select the penalty parameter. Furthermore, we fix the trimming proportion at 0.25 to achieve a breakdown point of 25%. For LASSO and Elastic Net, we use the `glmnet` R package (Friedman et al. 2010), and select the penalty parameter using 5-fold cross-validation.

For all RobScout and Scout methods, we use $n_{\lambda_1} = 100$ and $n_{\lambda_2} = 100$ and also use 5-fold cross-validation to select λ_1 and λ_2. We use the R implementation available through the `scout` package, along with the optimization algorithms described in Section 3. We also make modifications to certain functions in the `scout` package to improve computation time, including the use of the `huge` R package (Jiang et al. 2023) in place of the `glasso` R package (Friedman et al. 2019) for computing precision matrix estimates by the GLASSO algorithm. The details of these modifications are documented in Appendix A.

Finally, for all methods, we use the corresponding package implementation to standardize the data and estimate an intercept term. For all data generation models presented in the next section, the true intercept term is assumed to be zero.

4.2. Data Generation

Our simulation studies examine four data generation settings and three outlier generation settings. The true relationship between the predictor variables and the response is given by the following linear regression model:

$$y_i = \mathbf{x}_i^\intercal \boldsymbol{\beta} + e_i, \quad e_i \sim N(0, \sigma^2), \quad i = 1, \dots, n, \tag{12}$$

where n is the number of observations, and $\boldsymbol{\beta}$ is a length-p vector of true coefficients. In the following sections, let $\mathbf{X} = \left(\mathbf{x}_1^\intercal, \dots, \mathbf{x}_n^\intercal\right)^\intercal$ denote the $n \times p$ matrix of observations.

Clean Data Models

We consider different dimensionalities and correlation structures of predictor variables in each underlying clean data generation model. In particular, we are interested in cases where variables have either a block correlation structure or a first-order autoregressive (AR1) structure. In all settings, a subset of the true coefficients are non-zero.

(1) The first setting is given by Example (d) in Zou and Hastie (2005). We generate $p = 40$ predictors with sample size $n = 50$, where the true regression coefficients are

$$\boldsymbol{\beta} = (\underbrace{3, \dots, 3}_{15}, \underbrace{0, \dots, 0}_{25}) \tag{13}$$

and $\sigma = 15$. The predictors are generated from the following latent variable model:

$$\begin{aligned}
X_i &= Z_1 + e_i^x, \quad Z_1 \sim N(0,1), \quad i = 1, \dots, 5\\
X_i &= Z_2 + e_i^x, \quad Z_2 \sim N(0,1), \quad i = 6, \dots, 10\\
X_i &= Z_3 + e_i^x, \quad Z_3 \sim N(0,1), \quad i = 11, \dots, 15\\
X_i &\sim N(0,1), \qquad\quad X_i \text{ i.i.d.}, \quad i = 16, \dots, 40
\end{aligned}$$

We use e_i^x to denote the error term in the generation of predictor i (in contrast to the error term in Equation 12). The e_i^x are i.i.d. $N(0, 0.01)$, $i = 1, \ldots, 15$. In this model, the first 15 variables are grouped into 3 blocks of 5 correlated variables. The remaining 25 variables are noise features.

(2) The second setting is identical to the first one, but the number of predictors is increased to $p = 200$, and the number of predictors in each correlated block is increased to 20. The true regression coefficients in this setting are:

$$\boldsymbol{\beta} = (\underbrace{3, \ldots, 3}_{60}, \underbrace{0, \ldots, 0}_{140}) \tag{14}$$

(3) The third setting is given by Example 2 in (Zou and Zhang 2009). In this setting, $n = 100$, $p = 81$, and $\sigma = 6$. The true regression coefficients are:

$$\boldsymbol{\beta} = (\underbrace{3, \ldots, 3}_{27}, \underbrace{0, \ldots, 0}_{54}) \tag{15}$$

The predictors are generated from a multivariate Normal distribution where the covariance matrix follows an AR1 structure with correlation 0.75, i.e., $\mathbf{X}$ is generated from the $N_p(\mathbf{0}, \boldsymbol{\Sigma})$ distribution, where $\Sigma_{jk} = 0.75^{|j-k|}$ for $j, k = 1, \ldots, 81$.

(4) The fourth setting has $p = 51$ predictors with $n = 200$ observations, and $\sigma = 6$. The true regression coefficients are:

$$\boldsymbol{\beta} = (3, 3, 3, 0, 0, 0, \ldots, 0, 0, 0, 3, 3, 3) \tag{16}$$

The predictors are generated from a multivariate Normal distribution where the covariance matrix follows an AR1 sturcture with correlation 0.9, i.e., $\mathbf{X}$ is generated from the $N_p(\mathbf{0}, \boldsymbol{\Sigma})$ distribution, where $\Sigma_{jk} = 0.9^{|j-k|}$ for $j, k = 1, \ldots, 51$. We note that in this setting, non-zero coefficients are placed in intervals. Due to the AR1 correlation structure, predictors that are farther apart (i.e., j and k with larger absolute differences) are less correlated. Hence, non-zero coefficients that are spaced apart are more difficult to select.

Contamination

We consider three main models of contamination: clean data, casewise contamination, and cellwise contamination. We use ϵ to denote the probability of data being contaminated. Under the clean data model, $\epsilon = 0$. Under casewise contamination, ϵ is the probability of a row of $\mathbf{X}$ being outlying, so the expected number of rows in $\mathbf{X}$ that deviate from the clean data distribution is $\lfloor \epsilon n \rfloor$. Under cellwise contamination, ϵ is the probability of a cell being outlying, so the expected number of outlying cells in each row of $\mathbf{X}$ is $\lfloor \epsilon p \rfloor$. Finally, we consider two types of cellwise contamination: one in which cells are marginally outlying, and another in which cells are outlying with respect to the subspace of the contaminated predictors in each row.

(C1) *Clean data*: The data are generated by the settings in the previous section.

(C2) *Casewise contamination*: We follow the contamination model by Maronna (2011), which is controlled by two constants k_{lev} and k_{slo}. We use k_{lev} to control the leverage of the contamination in the predictors, and k_{slo} to control the magnitude and position of contamination in the response. We introduce contamination in the first $m = \lfloor \epsilon n \rfloor$ observations, where $\epsilon \in [0, 1]$ is the proportion of contaminated observations. We fix $\epsilon = 0.1$.

We introduce leverage points as follows:

$$\mathbf{x}_i^{(\text{cont})} = \eta_i + \frac{k_{\text{lev}}}{\sqrt{\mathbf{a}^\intercal \boldsymbol{\Sigma}^{-1} \mathbf{a}}} \mathbf{a}, \quad i = 1, \dots, m \tag{17}$$

where $\eta_i \sim N_p \left(\mathbf{0}, 0.1^2 \mathbf{I}_p \right)$ and $\mathbf{a} = \tilde{\mathbf{a}} - \frac{1}{p} \tilde{\mathbf{a}}^\intercal \mathbf{1}_p$. Here, $\tilde{a}$ is a length-p vector with entries $\tilde{a}_j \sim U(-1, 1)$, $j = 1, \dots, p$.

The response is contaminated by altering the regression coefficients as follows:

$$y_i = \mathbf{x}_i^{(\text{cont})} \boldsymbol{\beta}^{(\text{cont})} \quad \text{with } \beta_j^{(\text{cont})} = \begin{cases} \beta_j \left(1 + k_{\text{slo}} \right) & \text{if } \beta_j \neq 0 \\ k_{\text{slo}} \|\boldsymbol{\beta}\|_\infty & \text{otherwise} \end{cases}, \quad i = 1, \dots, m \tag{18}$$

where $\|\boldsymbol{\beta}\|_\infty$ is the largest entry in $\boldsymbol{\beta}$. Note that if $k_{\text{slo}} = 0$, no outliers are introduced in the response.

To evaluate the effectiveness of robust methods against this contamination setting, we follow Cohen Freue et al. (2018) and fix k_{lev} at 2 while we vary k_{slo} in a sequence of 19 values. We construct a logarithmically spaced sequence of length 15 from 1 to 500 (inclusive), and another logarithmically spaced sequence of length 5 from 500 to 2000. Since 500 is included twice, we discard one occurence in the final sequence.

(C3a) *Cellwise contamination (marginally outlying cells)*: Following the contamination model by Lafit et al. (2022), we contaminate cells in $\mathbf{X}$ by replacing them with observations generated from i.i.d. Normal random variables with higher means and lower variances. That is, for each observation $\mathbf{x}_i$, we replace it with

$$\mathbf{x}_i^{(\text{cont})} = \mathbf{x}_i \left(\mathbf{I} - \mathbf{b} \right) + \mathbf{z}_i \mathbf{b} \tag{19}$$

where $\mathbf{b}$ is a diagonal matrix with entries b_j that are drawn independently from the Bernoulli(ϵ) distribution, $j = 1, \dots, p$. Let $k = \sum_{j=1}^p b_j$ be the number of contaminated observations in the row $\mathbf{x}_i$. In Equation 19, $\mathbf{z}_i$ is generated from the $N(\boldsymbol{\mu}_k, \sigma^2 \boldsymbol{\Sigma}_k)$ distribution with $\sigma = 0.2$, where $\boldsymbol{\mu}_k$ is a mean vector of length k where each entry is 10, and $\boldsymbol{\Sigma}_k$ is the true covariance of the clean data model restricted to the k contaminated predictors.

We consider contamination probabilities $\epsilon = 0.1, 0.25$. Moreover, under models where the true vector of coefficients is sparse, we consider different combinations of contamination fractions in the active and inactive predictors, where active predictors are those with non-zero coefficients and inactive predictors are those with zero-valued coefficients. That is, we consider every pair $(\epsilon_{\text{active}}, \epsilon_{\text{inactive}})$ where $\epsilon_{\text{active}} \in \{0.1, 0.25\}$ and $\epsilon_{\text{inactive}} \in \{0.1, 0.25\}$.

(C3b) *Cellwise contamination (correlation outliers)*: To generate cellwise outliers that are not marginally outlying, we follow the contamination framework in (Raymaekers and Rousseeuw 2021a).

In each column of the data matrix, we draw $n\epsilon$ indices to be contaminated. For each row $(x_1, \ldots, x_p)$ with at least one contaminated data cell, let k be the number of contaminated cells in that row. Let $K = \{j(1), \ldots, j(k)\}$ denote the ordered indices of those k contaminated cells in the row, and let the corresponding cells at the indices in K be the k-dimensional vector $\left(x_{j(1)}, \ldots, x_{j(k)}\right)$. We replace these cells by the following k-dimensional row vector:

$$\mathbf{v} = \frac{\gamma\sqrt{k}\mathbf{u}^\intercal}{\mathrm{MD}\left(\mathbf{u}, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}, \tag{20}$$

where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the true mean and covariance of the clean data model restricted to the indices in the set K, $\mathbf{u}$ is the eigenvector of $\boldsymbol{\Sigma}_k$ with the smallest eigenvalue, and MD is the Mahalanobis distance. This contamination model generates points that are outlying the subspace of coordinates of K, but that are not marginally oulying. The value γ controls the magnitude of outlyingness in the direction of $\mathbf{u}$, and as γ gets larger, the outliers become more marginally outlying.

In this setting, we consider $\gamma = 1, 5, 10, 15, 20$ and contamination probabilities $\epsilon = 0.01, 0.05, 0.1, 0.2$.

4.3. Evaluation Metrics

We generate a clean test set of $n^{(\text{test})} = 1000$ to evaluate model performance. Let $\mathbf{X}^{(\text{test})}$ and $\mathbf{y}^{(\text{test})}$ denote the observed predictor predictor variables and response variables in the test set, respectively. Let $\hat{\mathbf{y}}$ be the vector of predicted response values on the test set.

We evaluate a method's *prediction performance* using the Root Mean Squared Prediction Error (RMSPE) between actual and predicted response values, standardized by the true standard deviation of the error term in the underlying model, σ:

$$\frac{\mathrm{RMSPE}\left(\hat{\mathbf{y}}, \mathbf{y}^{(\text{test})}\right)}{\sigma} = \frac{1}{\sigma}\sqrt{\frac{1}{n^{(\text{test})}}\sum_{i=1}^{n^{(\text{test})}}\left(y_i^{(\text{test})} - \hat{y}_i\right)^2}. \tag{21}$$

Hence, a RMSPE/σ value of 1 indicates perfect prediction performance and is the best possible value.

We also evaluate each method's *variable selection* performance by considering the sensitivity, specificity, and the F-measure of the selected coefficients.

$$\mathrm{SENS}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right) = \frac{\#\left\{\, j : \hat{\beta}_j \neq 0 \text{ and } \beta_j \neq 0 \,\right\}}{\#\left\{\, j : \beta_i \neq 0 \,\right\}}$$

$$\mathrm{SPEC}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right) = \frac{\#\left\{\, j : \hat{\beta}_j = 0 \text{ and } \beta_j = 0 \,\right\}}{\#\left\{\, j : \beta_j = 0 \,\right\}}$$

$$\text{F-measure}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right) = \frac{2 \times \mathrm{SENS}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right) \times \mathrm{SPEC}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right)}{\mathrm{SENS}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right) + \mathrm{SPEC}\left(\hat{\boldsymbol{\beta}}, \boldsymbol{\beta}\right)}$$

An ideal method will have high variable selection sensitivity and specificity, resulting in a high F-measure as well.

4.4. Simulation Results

In this section we present results from each simulation setting. This section is organized by contamination model.

Casewise Contamination (C2)

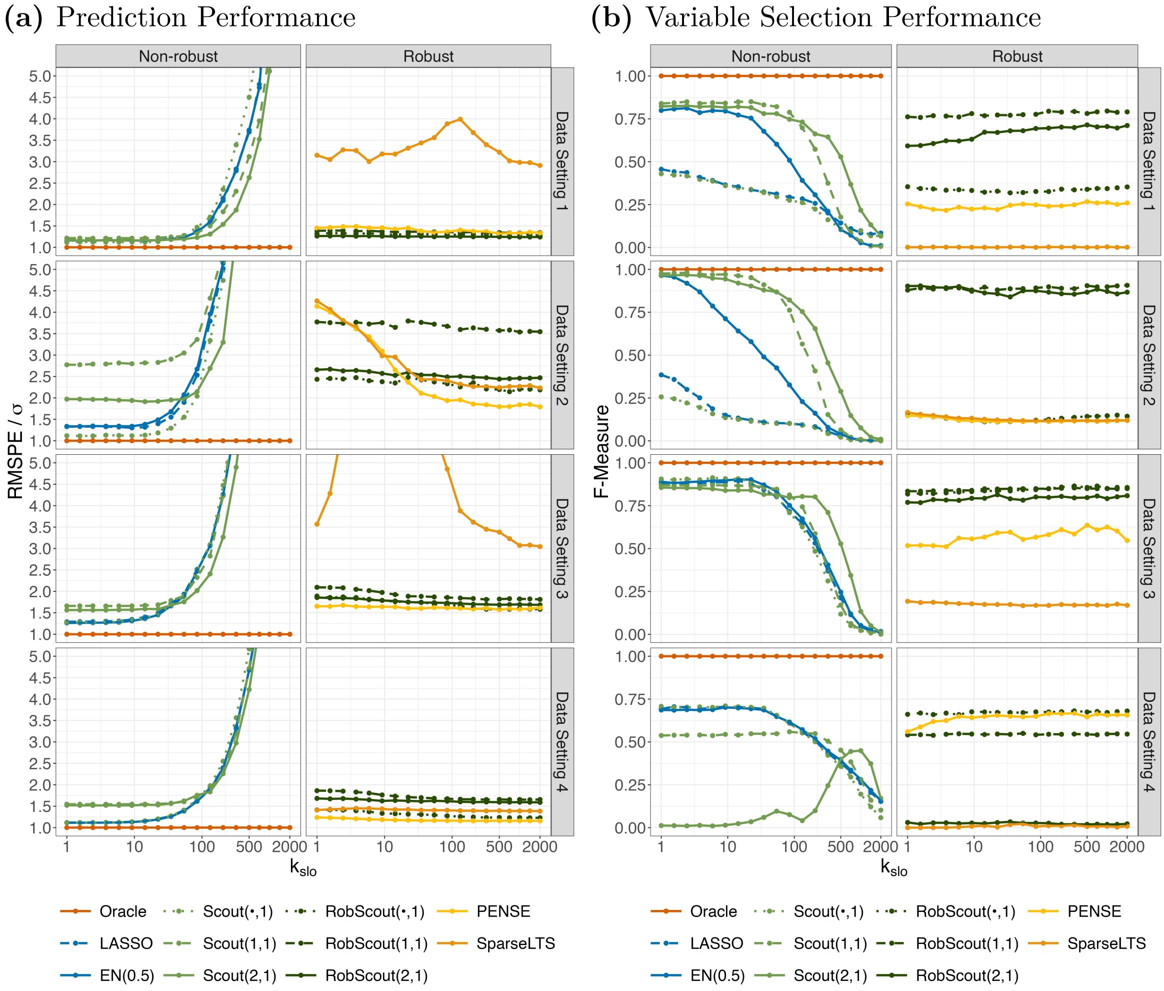
Under casewise contamination, we see that all non-robust estimators, except for the Oracle estimator, break down in their prediction performance as k_{slo} is increased from 1 to 2000. However, all RobScout estimators maintain a consistent prediction performance throughout, as shown in Figure 1a. Notably, at least one RobScout method shows similar or superior prediction performance to robust competitors. Their variable selection performances remain high even as k_{slo} is increased toward the highest value, with at least one RobScout method outperforming both PENSE and SparseLTS. This is demonstrated even in high-dimensional cases, such as in data setting 2 where $n = 50$ and $p = 200$.

Cellwise Contamination with Marginally Outlying Cells (C3a)

Cellwise Contamination with Correlation Outliers (C3b)

4.5. Runtime

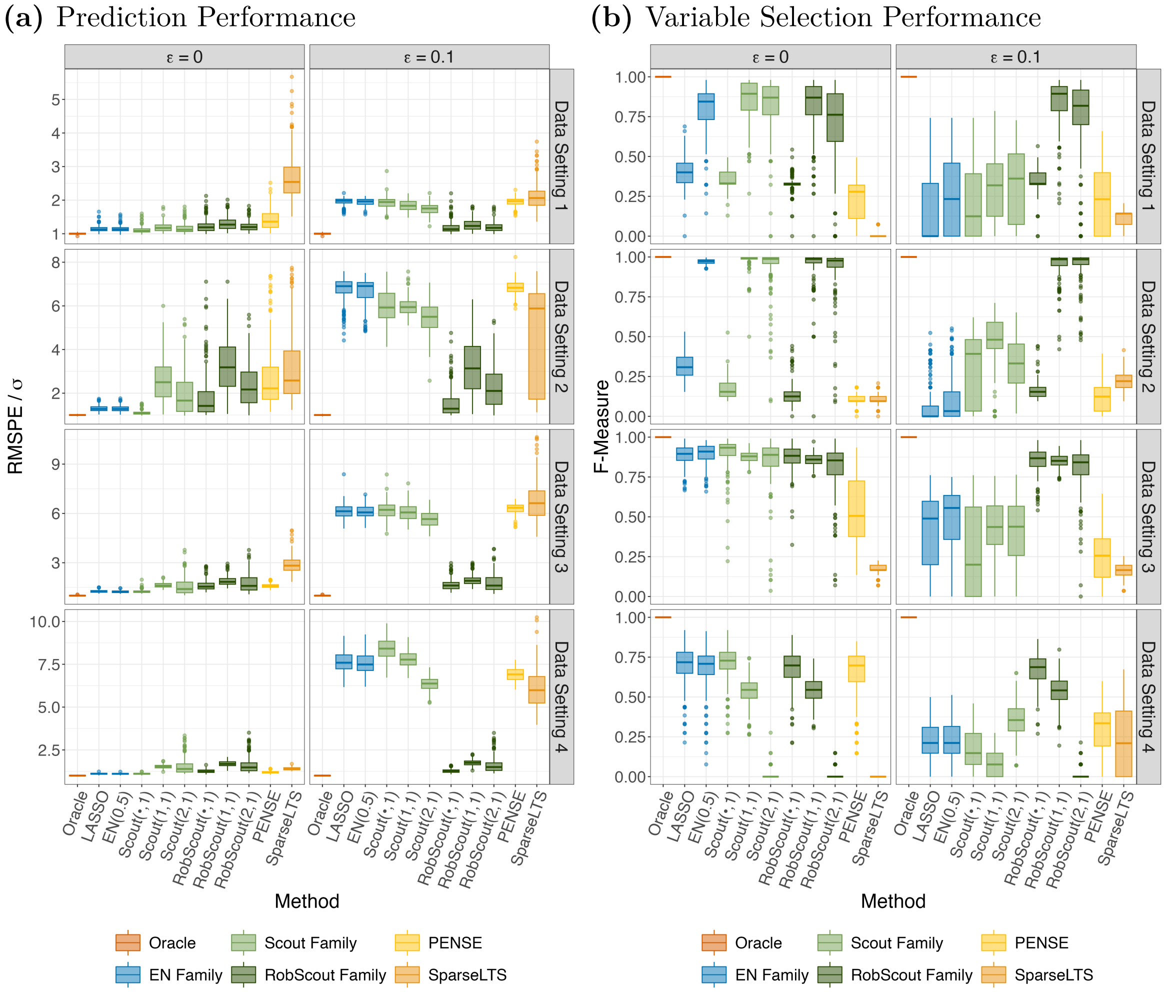
5. Discussion



(a) Prediction Performance

(b) Variable Selection Performance

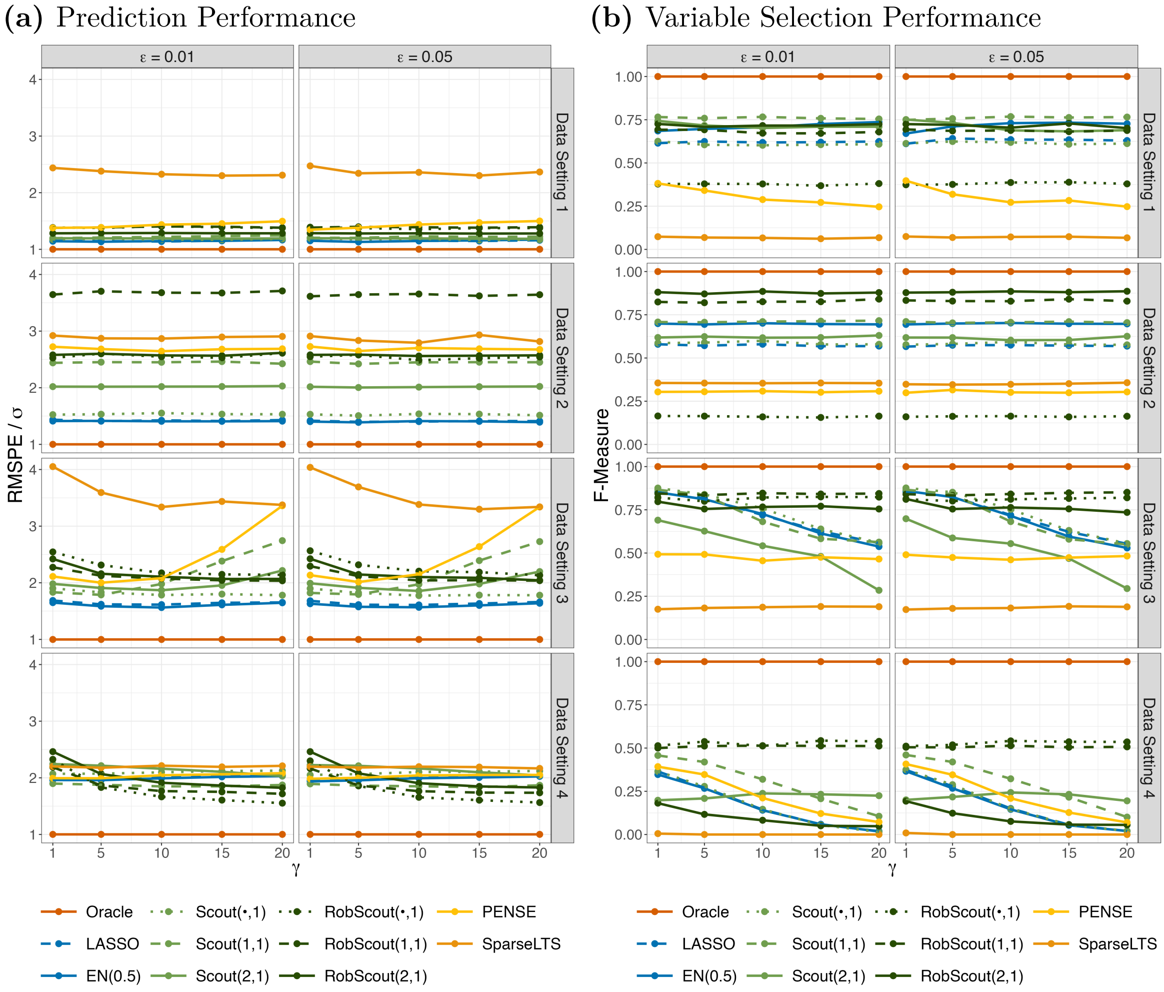
Figure 1: Prediction (left) and variable selection (right) performances of non-robust and robust methods under Contamination (C2) for k_{slo} values ranging from 1 to 2000. Subfigure row labels indicate data generation setting.



(a) Prediction Performance

(b) Variable Selection Performance

Figure 2: Prediction (left) and variable selection (right) performances of non-robust and robust methods under Contamination (C3a) for $\epsilon = 0, 0.1$. Note that $\epsilon = 0$ corresponds to no contamination, i.e., clean data (C1).



(a) Prediction Performance

(b) Variable Selection Performance

Figure 3: Prediction (left) and variable selection (right) performances of non-robust and robust methods under Contamination (C3a) for $\epsilon = 0.01, 0.05$.

Acknowledgments

References

Alfons, A. (2021). robustHD: An R package for robust regression with high-dimensional data. *Journal of Open Source Software*, 6(67):3786, DOI: 10.21105/joss.03786.

Alfons, A., Croux, C., and Gelper, S. (2013). Sparse least trimmed squares regression for analyzing high-dimensional large data sets. *The Annals of Applied Statistics*, pages 226–248.

Alqallaf, F., Van Aelst, S., Yohai, V. J., and Zamar, R. H. (2009). Propagation of outliers in multivariate data. *The Annals of Statistics*, pages 311–331.

Banerjee, O., El Ghaoui, L., and d'Aspremont, A. (2008). Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *The Journal of Machine Learning Research*, 9:485–516.

Bottmer, L., Croux, C., and Wilms, I. (2022). Sparse regression for large data sets with outliers. *European Journal of Operational Research*, 297(2):782–794.

Cohen Freue, G., Kepplinger, D., Salibian-Barrera, M., and Smucler, E. (2018). Proteomic biomarker study using novel robust penalized elastic net estimators. *Ann Appl Stat (submitted).*

Croux, C. and Öllerer, V. (2016). *Robust and sparse estimation of the inverse covariance matrix using rank correlation measures.* Springer.

Filzmoser, P., Höppner, S., Ortner, I., Serneels, S., and Verdonck, T. (2020). Cellwise robust m regression. *Computational Statistics & Data Analysis*, 147:106944.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1.

Friedman, J., Hastie, T., and Tibshirani, R. (2019). *glasso: Graphical Lasso: Estimation of Gaussian Graphical Models*, https://CRAN.R-project.org/package=glasso. R package version 1.11.

Hampel, F. R., Rousseeuw, P. J., and Ronchetti, E. (1981). The change-of-variance curve and optimal redescending m-estimators. *Journal of the American Statistical Association*, 76(375):643–648.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

Jiang, H., Fei, X., Liu, H., Roeder, K., Lafferty, J., Wasserman, L., Li, X., and Zhao, T. (2023). *huge: High-Dimensional Undirected Graph Estimation*, https://CRAN.R-project.org/package=huge. R package version 1.3.6.

Kepplinger, D., Salibián-Barrera, M., and Cohen Freue, G. (2023). *pense: Penalized Elastic Net S/MM-Estimator of Regression*, https://CRAN.R-project.org/package=pense. R package version 2.2.0.

Lafit, G., Nogales, F., Ruiz, M., and Zamar, R. (2022). Robust graphical lasso based on multivariate winsorization. *arXiv preprint arXiv:2201.03659*.

Mardia, K., Kent, J., and Bibby, J. (1979). *Multivariate Analysis.* Academic Press Limited.

Maronna, R. A. (2011). Robust ridge regression for high-dimensional data. *Technometrics*, 53(1):44–53.

Öllerer, V., Alfons, A., and Croux, C. (2016). The shooting s-estimator for robust regression. *Computational Statistics*, 31:829–844.

Öllerer, V. and Croux, C. (2015). Robust high-dimensional precision matrix estimation. *Modern Nonparametric, Robust and Multivariate Methods*, pages 325–350.

Raymaekers, J. and Rousseeuw, P. (2021a). Handling cellwise outliers by sparse regression and robust covariance. *Journal of Data Science, Statistics, and Visualisation*, 1(3), DOI: 10.52933/jdssv.v1i3.18, https://jdssv.org/index.php/jdssv/article/view/18.

Raymaekers, J. and Rousseeuw, P. J. (2021b). Fast robust correlation for high-dimensional data. *Technometrics*, 63(2):184–198.

Rousseeuw, P. J. and Bossche, W. V. D. (2018). Detecting deviating data cells. *Technometrics*, 60(2):135–145.

Smucler, E. and Yohai, V. J. (2017). Robust and sparse estimators for linear regression models. *Computational Statistics & Data Analysis*, 111:116–130.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288.

Witten, D. M. and Tibshirani, R. (2009). Covariance-regularized regression and classification for high dimensional problems. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 71(3):615–636.

Yohai, V. J. and Zamar, R. H. (1988). High breakdown-point estimates of regression by means of the minimization of an efficient scale. *Journal of the American Statistical Association*, 83(402):406–413.

Yuan, M. and Lin, Y. (2007). Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320.

Zou, H. and Zhang, H. H. (2009). On the adaptive elastic-net with a diverging number of parameters. *Annals of Statistics*, 37(4):1733.

A. Modifications to the scout R package

B. Signal to Noise Ratio of each Data Generation Model

C. Additional Simulation Results