Supplement to "A Sharper Computational Tool for L_2E Regression"

Xiaoqian Liu
Department of Statistics, North Carolina State University
Eric C. Chi
Department of Statistics, Rice University
and
Kenneth Lange
Departments of Computational Medicine, Human Genetics, and Statistics
University of California, Los Angeles

This supplement provides two additional simulation examples to illustrate the advantage of the proposed computational method for robust L_2E regression. In Section A, we report an example of robust convex regression to show the advantage of the proposed method in computational efficiency and estimation accuracy over the original one in Chi and Chi (2022). In Section B, we compare distance penalty to l_1 penalty for robust trend filtering under the L_2E framework. Recall that we refer to the proposed method in our paper as "MM", the original method in Chi and Chi (2022) as "PG", and the ordinary least squares as "LS."

A Robust convex regression

Convex regression shares the same squared loss with isotonic regression but imposes different constraints $\beta_{i+1} \leq \frac{1}{2}(\beta_i + \beta_{i+2})$ for $1 \leq i < n-1$. Collectively, these constraints can

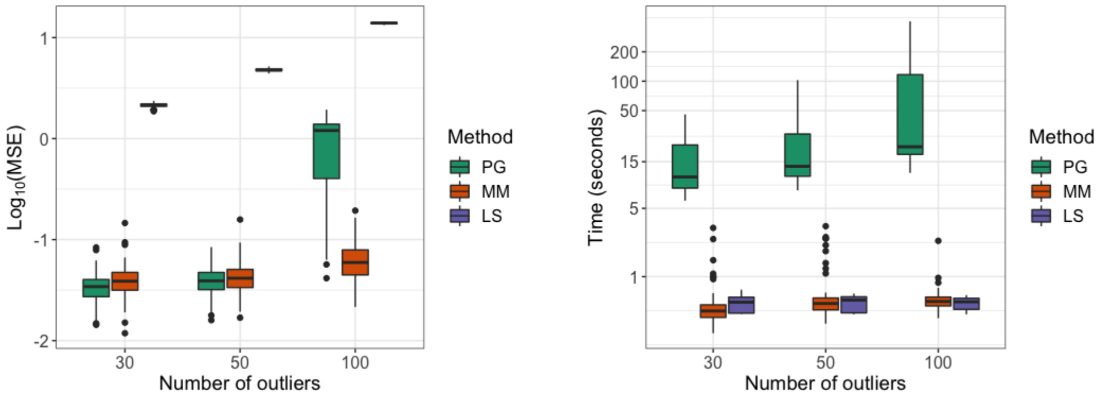

Figure A.1: Simulation results for convex regression under different numbers of outliers. Boxplots depict the MSE (left panel) and run time (right panel) over 100 replicates.

be expressed as the constraint set $S_1 = \{\boldsymbol{\beta} \in \mathbb{R}^n : \boldsymbol{D}\boldsymbol{\beta} \geq \mathbf{0}\}$, where $\boldsymbol{D}$ is the corresponding second-order difference matrix. In the L_2E version of the problem, we apply the $0/\infty$ penalty $\phi(\boldsymbol{\beta}) = \iota_{S_1}(\boldsymbol{\beta})$. The MM update of $\boldsymbol{\beta}$ can be computed by the `conreg` function in the `cobs` R package (Ng and Maechler, 2007). The MM $\boldsymbol{\beta}$ update for convex regression again requires the same computational cost as the PG $\boldsymbol{\beta}$ update (Chi and Chi, 2022).

In our numerical experiment, the components of the mean vector $\boldsymbol{\beta} \in \mathbb{R}^{500}$ conforms to the quartic function $x_i^4 + x_i$. Two kinds of perturbations are added to generate the responses $y_i = x_i^4 + x_i + s_i + \varepsilon_i$. Here the x_i are sampled evenly from $[-2, 2]$, the s_i shift the underlying quartic signal, and the ε_i are i.i.d. standard normal deviates. Outliers are introduced at consecutive responses by setting the shifts $s_i = 14$ for $i = 126, 127, \cdots, 125 + m$ where m is the number of outliers; for all other responses $s_i = 0$. We use the same initialization as in the isotonic regression example for PG and MM.

Figure A.1 presents boxplots of the MSEs and run times in seconds in fitting the convex regression model with 100 replicates. The estimation accuracy of LS and PG degrades as

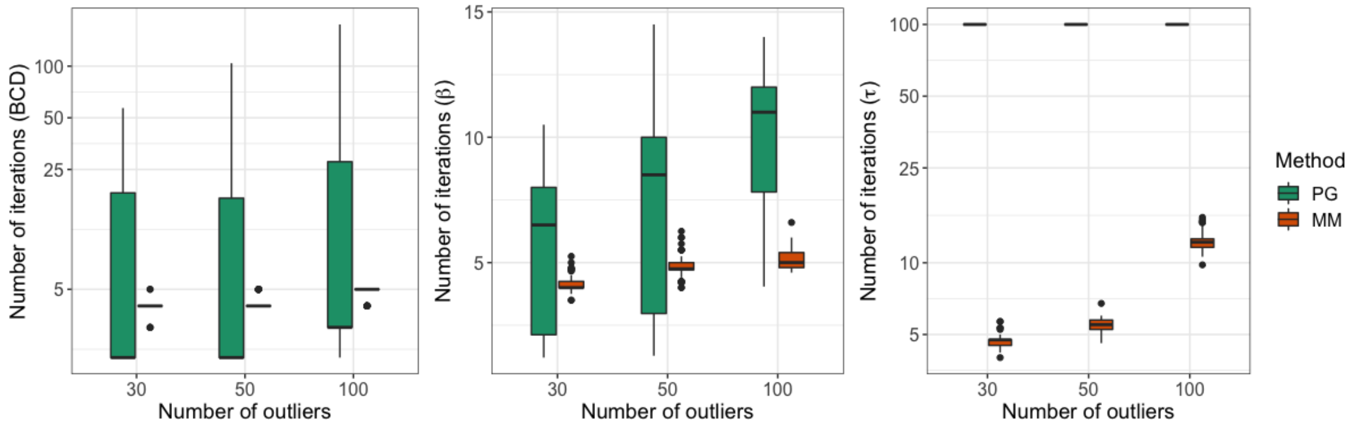

Figure A.2: Boxplots of the mean number of outer block descent iterations (left panel), the mean number of inner iterations for updating $\boldsymbol{\beta}$ per outer iteration (middle panel), and the mean number of inner iterations for updating τ per outer iteration (right panel). All plots refer to the experiment summarized in Figure A.1.

the number of outliers increases, while MM behaves stably with outliers. As the right panel of Figure A.1 shows, the run times of PG quickly increase as the number of outliers increases, while the run times of MM are immune to outliers and comparable to those of LS. Thus, MM under the L_2E loss enjoys the speed advantages of LS while protecting against outliers. To fully comprehend the difference in run times between PG and MM, Figure A.2 displays boxplots of the mean number of outer block descent iterations, the mean number of inner iterations for updating $\boldsymbol{\beta}$ per outer iteration, and the mean number of inner iterations for updating τ per outer iteration. MM takes not only fewer outer iterations than PG but also fewer average inner iterations to update $\boldsymbol{\beta}$ and τ. This contributes to the distinct speed advantage of MM over PG on convex regression.

In the second experiment, we investigate the performance of PG and MM under different contamination levels. For that purpose, we vary the contamination level by setting the shifts

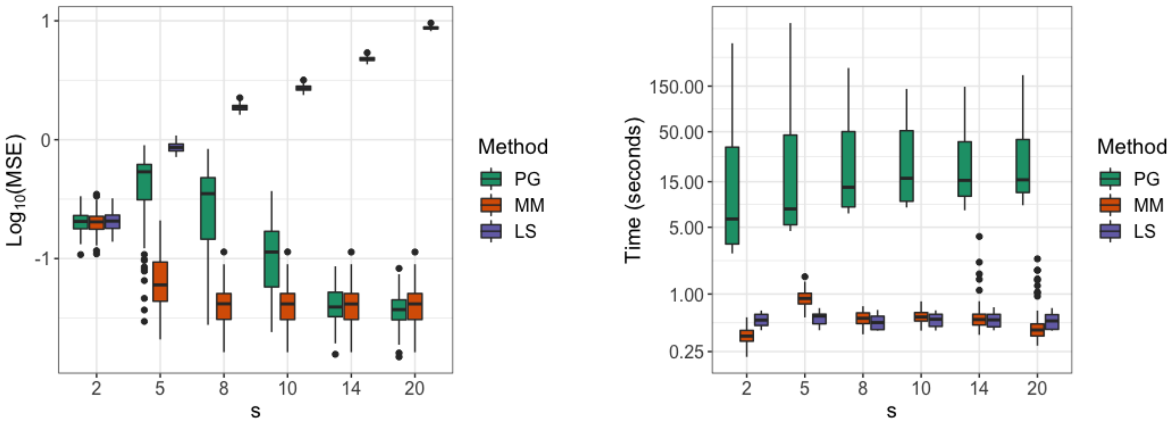

Figure A.3: Simulation results for convex regression under different contamination levels. Boxplots depict the MSE (left panel) and run time (right panel) over 100 replicates.

$s_i = \{2, 5, 8, 14, 20\}$ for $i = 126, 127, \cdots, 125 + m$ with a fixed number of outliers $m = 50$. Initialization is the same as in the previous experiment.

Figure A.3 summarizes the estimation and computation results for convex regression under different contamination levels. As with the isotonic regression example, PG and MM perform comparably with the non-robust LS when the data are slightly contaminated. As the level of contamination increases, LS produces increasingly large MSE values, while MM consistently obtains small MSE values. The MSEs of PG slowly decline and get close to those of MM as the level of contamination s_i grows, suggesting that PG is less efficient in detecting outliers when the contamination level is modest. The right panel of Figure A.3 displays the computational advantage of MM over PG under varying contamination levels. Run times of MM are much shorter than those of PG and stable with the contamination level. In contrast, run times of PG increase as the contamination level increases. Figure A.4 reveals the reason for the difference in computation speed between PG and MM. As the amount of shifts increases, PG requires more inner iterations for updating $\boldsymbol{\beta}$ as well

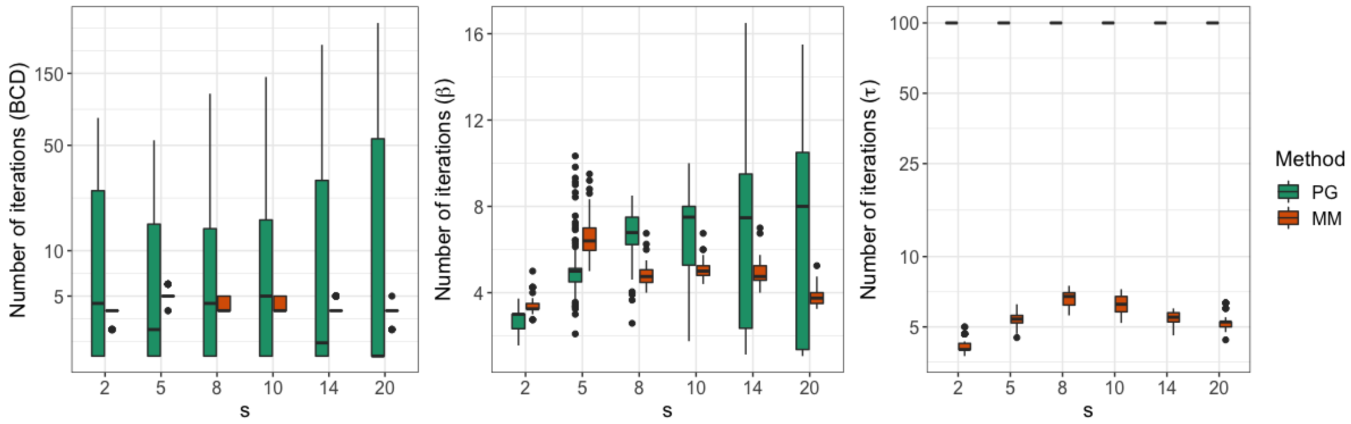

Figure A.4: Boxplots of the mean number of outer block descent iterations (left panel), the mean number of inner iterations for updating $\boldsymbol{\beta}$ per outer iteration (middle panel), and the mean number of inner iterations for updating τ per outer iteration (right panel). All plots refer to the experiments summarized in Figure A.3

as more outer block descent iterations to converge. The required numbers of both inner and outer iterations for MM are always small and relatively immune to the contamination level, leading to the fast and stable computational performance of MM.

B Robust Trend Filtering

Trend filtering is a good example to illustrate the complexities encountered in combining L_2E regression with distance penalization, sparsity recovery, and fusion constraints. The simplest version of trend filtering imposes sparsity on the differences $\beta_{i+1} - \beta_i$. The penalty can be expressed as $\phi(\boldsymbol{\beta}) = \|\boldsymbol{D}\boldsymbol{\beta}\|_0$, where $\boldsymbol{D}$ is a difference matrix and $\|\boldsymbol{\theta}\|_0$ counts the number of nonzero entries of $\boldsymbol{\theta}$. Thus, if the underlying trend is piecewise constant, then $\boldsymbol{D}$ is a first-order difference matrix. If the trend is piecewise affine, then $\boldsymbol{D}$ is a second-order difference matrix. In practice, the problem is often convexified by substituting an ℓ_1 penalty for the ℓ_0 penalty. With this change, it is possible to treat the MM update of $\boldsymbol{\beta}$ as a generalized lasso problem and use the `genlasso` function in the R package `genlasso` (Arnold and Tibshirani, 2019) to compute the solution.

A brief study of trend filtering under the L$_2$E loss and the distance and ℓ_1 penalties is illuminating. The simulated data consists of six consecutive segments with $\boldsymbol{\beta}$ values 0, 6, 3, -1, 6, and 0 and jumps at 40, 60, 90, 140, and 160. The components of $\boldsymbol{\beta}$ are randomly perturbed by Gaussian deviates $\epsilon_i \sim N(0, 0.5)$ to create the observed responses y_i. Random responses are then shifted by 5 to produce outliers. Distance penalization invokes the constraint set $S_2 = \{\boldsymbol{\beta} \in \mathbb{R}^{200} : \|\mathbf{D}\boldsymbol{\beta}\|_0 \leq k\}$ where $\boldsymbol{D}$ is the first-order difference matrix and the tuning parameter k denotes the number of jumps in $\boldsymbol{\beta}$. Ideally, we should find the sparsity level $k = 5$. We examine k over the grid $\{3, 5, 7, 9, 11, 13, 15\}$ and employ five-fold cross-validation to select k. The penalty constant ρ is set to 10^8. Under the ℓ_1 penalty $\lambda\|\boldsymbol{D}\boldsymbol{\beta}\|_1$, we also choose λ by five-fold cross-validation and declare a jump whenever the magnitude of a component $\boldsymbol{D}\hat{\boldsymbol{\beta}}$ exceeds 0.1. For both penalties, we measure estimation accuracy by the MSE of the trend estimate $\hat{\boldsymbol{\beta}}$. Structure recovery is determined by the number of true and false jumps. To initialize the L$_2$E estimation, we set $\eta_0 = -\log \operatorname{MAD}(\boldsymbol{y})$ and $\boldsymbol{\beta}_0 = \tilde{\boldsymbol{y}}$, where $\tilde{\boldsymbol{y}}$ is the denoised signal from a median filter.

Figure B.5 summarizes the average performance of the ℓ_1 and the distance penalties in robust trend filtering under different numbers of outliers over 50 replicates. As expected, MSE increases as the number of outliers increases. Distance penalization always achieves a lower MSE than ℓ_1 penalization. In terms of structure recovery, distance penalization

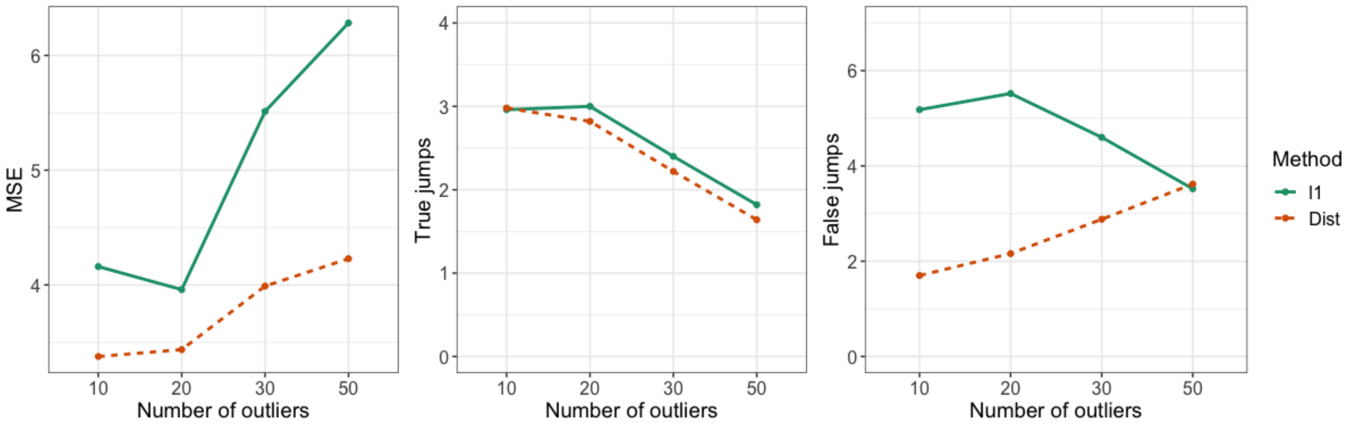

Figure B.5: Simulation results for trend filtering under different numbers of outliers. Average performance based on 50 replicates for each method.

performs comparably with l_1 penalization in capturing true jumps in the trend component, but it typically ignores undesirable false jumps, except for the case of $m = 50$. Both penalization methods miss more than half of the true jumps when the number of outliers exceeds 30, indicating that trend filtering is a difficult task when data are contaminated. This trend filtering example once again shows the flexibility of the $\mathrm{L}_2\mathrm{E}$ framework and the advantages of distance penalization.

References

Arnold, T. B. and Tibshirani, R. J. (2019), *genlasso: Path algorithm for generalized lasso problems*, R package version 1.4.

Chi, J. T. and Chi, E. C. (2022), "A User-Friendly Computational Framework for Robust Structured Regression with the L_2 Criterion," *Journal of Computational and Graphical Statistics*, 1–12.

Ng, P. and Maechler, M. (2007), "A fast and efficient implementation of qualitatively constrained quantile smoothing splines," *Statistical Modelling*, 7, 315–328.