The Strong Screening Rule for SLOPE

Statistical Learning Seminar

Johan Larsson[1] Małgorzata Bogdan[1,2] Jonas Wallin[1]

[1]Department of Statistics, Lund University,

[2]Department of Mathematics, University of Wroclaw

May 8, 2020

Recap: SLOPE

The SLOPE (Bogdan et al. 2015) estimate is

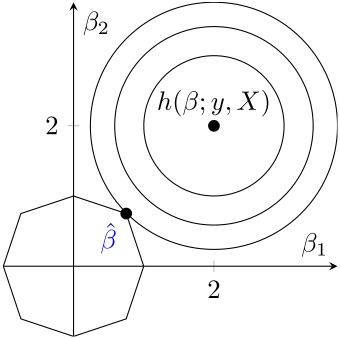
$$\hat{\beta} = \operatorname*{arg\,min}_{\beta \in \mathbb{R}^p} \{g(\beta) + J(\beta; \lambda)\}$$

where $J(\beta; \lambda) = \sum_{i=1}^{p} \lambda_i |\beta|_{(i)}$ is the **sorted ℓ_1 norm**, where

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0, \qquad |\beta|_{(1)} \geq |\beta|_{(2)} \geq \cdots \geq |\beta|_{(p)}.$$

Here we are interested in fitting a **path** of regularization penalties $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(m)}$

We will let $\hat{\beta}(\lambda^{(i)})$ correspond to the solution to SLOPE at the ith step on the path.



Predictor screening rules

motivation

many of the solutions, $\hat{\beta}$, along the regularization path will be **sparse**, which means some predictors (columns) in X will be **inactive**, especially if $p \gg n$

Predictor screening rules

motivation

many of the solutions, $\hat{\beta}$, along the regularization path will be **sparse**, which means some predictors (columns) in X will be **inactive**, especially if $p \gg n$

basic idea

what if we could, based on a relatively **cheap** test, determine which predictors will be inactive before fitting the model?

Predictor screening rules

motivation
many of the solutions, $\hat{\beta}$, along the regularization path will be **sparse**, which means some predictors (columns) in X will be **inactive**, especially if $p \gg n$

basic idea
what if we could, based on a relatively **cheap** test, determine which predictors will be inactive before fitting the model?

it turns out we can!

safe rules certifies that discarded predictors are not in model

heuristic rules may incorrectly discard some predictors, which means problem must sometimes be solved several times (in practice never more than twice)

Motivation for lasso strong rule

Assume we are solving the lasso, i.e. minimizing

$$g(\beta) + h(\beta), \qquad h(\beta) \coloneqq \lambda \sum_{i=1}^{p} |\beta_i|.$$

KKT stationarity condition is

$$\mathbf{0} \in \nabla g(\hat{\beta}) + \partial h(\hat{\beta}),$$

where $\partial h(\hat{\beta})$ is the subdifferential for the ℓ_1 norm with elements given by

$$\partial h(\hat{\beta})_i = \begin{cases} \operatorname{sign}(\hat{\beta}_i)\lambda & \hat{\beta}_i \neq 0 \\ [-\lambda, \lambda] & \hat{\beta}_i = 0, \end{cases}$$

which means that $|\nabla g(\hat{\beta})_i| < \lambda \implies \hat{\beta}_i = 0$.

Gradient estimate

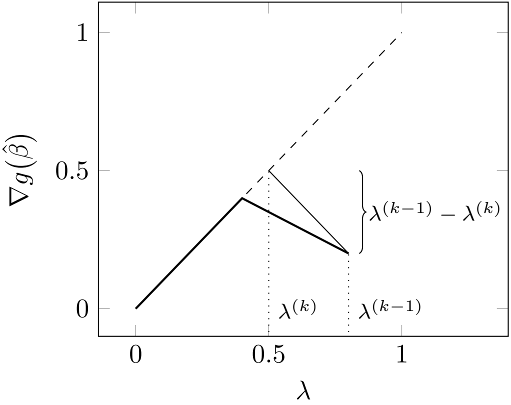
Assume that we are fitting a regularization **path** and have $\hat{\beta}(\lambda^{(k-1)})$—the solution for $\lambda^{(k-1)}$—and want to discard predictors corresponding to the problem for $\lambda^{(k)}$.

Basic idea: replace $\nabla g(\hat{\beta})$ with an estimate and apply the KKT stationarity criterion, discarding predictors that are estimated to be zero.

What estimate should we use?

The unit slope bound

A simple (and conservative) estimate turns out to be $\lambda^{(k-1)} - \lambda^{(k)}$, i.e. assume that the gradient is piece-wise linear function with slope bounded by 1.



The strong rule for the lasso

Discard the jth predictor if

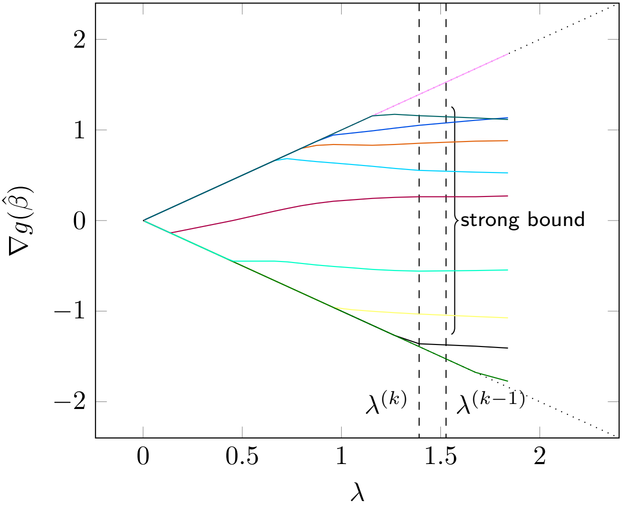
$$\underbrace{\underbrace{\left|\nabla g\left(\hat{\beta}(\lambda^{(k-1)})\right)\right|}_{\text{previous gradient}} + \underbrace{\lambda^{(k-1)} - \lambda^{(k)}}_{\text{unit slope bound}}}_{\text{gradient prediction for } k} < \lambda^{(k)}$$

$$\Longleftrightarrow$$

$$\left|\nabla g\left(\hat{\beta}(\lambda^{(k-1)})\right)\right| < 2\lambda^{(k)} - \lambda^{(k-1)}$$

Empirical results show that the strong rule leads to remarkable performance improvements in $p \gg n$ regime (and no penalty otherwise) (Tibshirani et al. 2012).

Strong rule for lasso in action



Strong rule for SLOPE

Exactly the same idea as for lasso strong rule.

The subdifferential for SLOPE is is the set of all $g \in \mathbb{R}^p$ such that

$$g_{\mathcal{A}_i} = \left\{ s \in \mathbb{R}^{\operatorname{card} \mathcal{A}_i} \;\middle|\; \begin{cases} \operatorname{cumsum}(|s|_\downarrow - \lambda_{R(s)_{\mathcal{A}_i}}) \preceq \mathbf{0} & \text{if } \beta_{\mathcal{A}_i} = \mathbf{0}, \\ \operatorname{cumsum}(|s|_\downarrow - \lambda_{R(s)_{\mathcal{A}_i}}) \preceq \mathbf{0} & \\ \quad \wedge \sum_{j \in \mathcal{A}_i} \left(|s_j| - \lambda_{R(s)_j}\right) = 0 & \text{otherwise.} \end{cases} \right\}$$

$\mathcal{A}_i$ defines a **cluster** containing indices of coefficients equal in absolute value.

$R(x)$ is an operator that returns the **ranks** of elements in $|x|$.

$|x|_\downarrow$ returns the absolute values of x sorted in non-increasing order.

Strong rule algorithm for SLOPE

Require: $c \in \mathbb{R}^p$, $\lambda \in \mathbb{R}^p$, where $\lambda_1 \geq \cdots \geq \lambda_p \geq 0$.

```
S, B ← ∅
for i ← 1, ..., p do
    B ← B ∪ {i}
    if ∑_{j∈B} (c_j − λ_j) ≥ 0 then
        S ← S ∪ B
        B ← ∅
    end if
end for
Return S
```

Set

$$c \coloneqq |\nabla g(\hat{\beta}) + \lambda^{(k-1)} - \lambda^{(k)}|_{\downarrow} \qquad \lambda \coloneqq \lambda^{(k)},$$

and run the algorithm above; the result is the predicted support for $\hat{\beta}(\lambda^{(k)})$ (subject to a permutation).

Efficiency for simulated data

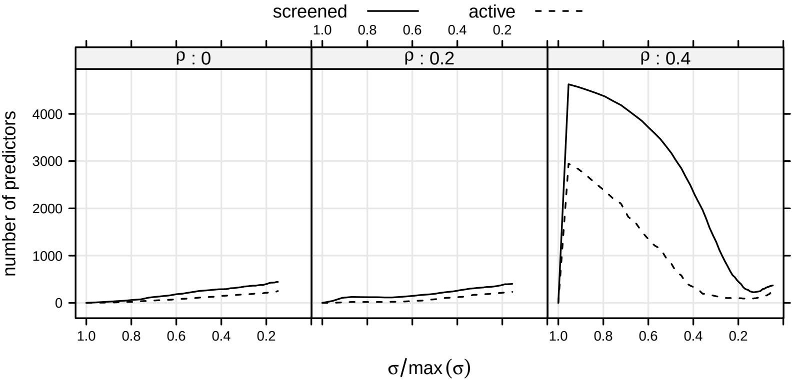


Figure 1: Gaussian design, $X \in \mathbb{R}^{200\times 5000}$, predictors pairwise correlated with correlation ρ. There were no violations of the strong rule here.

Efficiency for real data

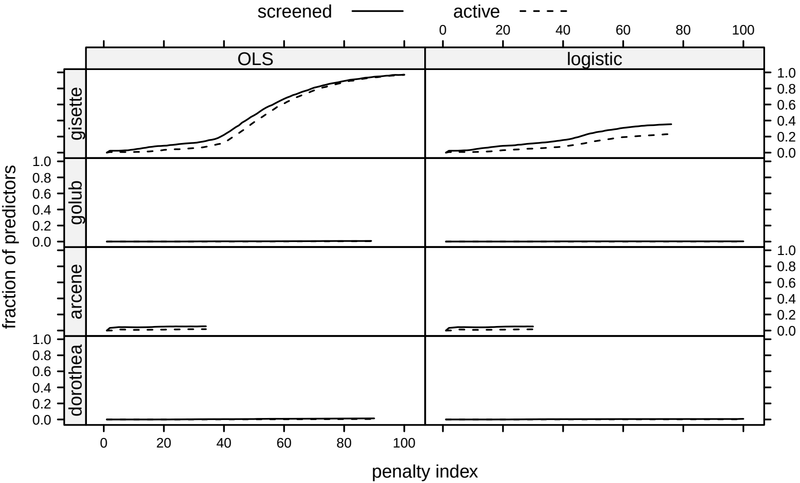


Figure 2: Efficiency for real data sets. The dimensions of the predictor matrices are 100×9920 (arcene), 800×88119 (dorothea), 6000×4955 (gisette), and 38×7129 (golub).

Violations

Violations may occur if the unit slope bound fails, which can occur if ordering permutation of absolute gradient changes, or any predictor becomes active between $\lambda^{(k-1)}$ and $\lambda^{(k)}$.

Thankfully, such violations turn out to be **rare**.

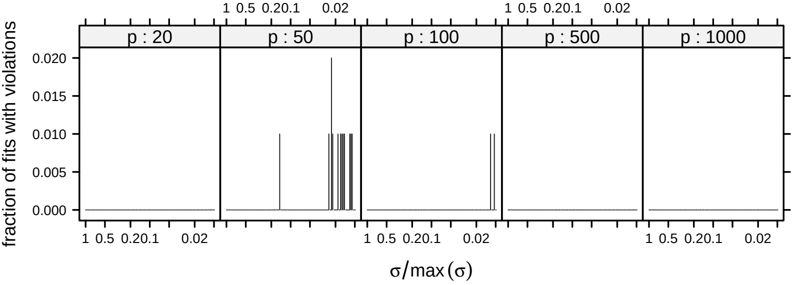


Figure 3: Violations for sorted ℓ_1 regularized least squares regression with predictors pairwise correlated with $\rho = 0.5$. $X \in \mathbb{R}^{100 \times p}$.

Performance

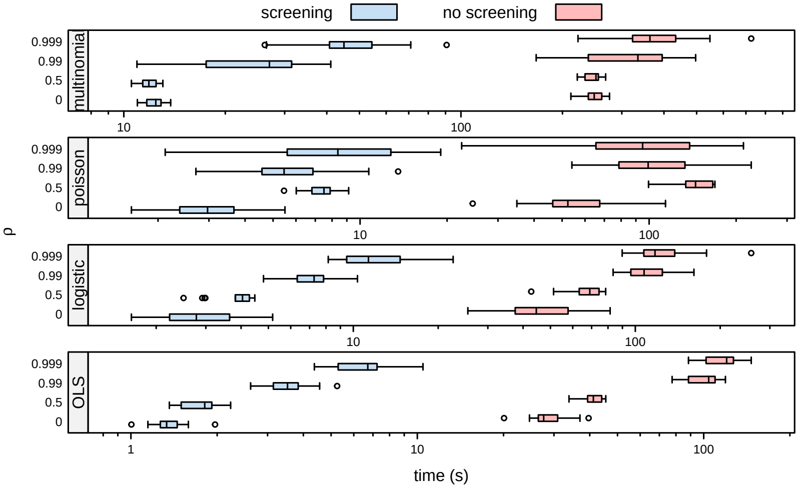


Figure 4: Performance benchmarks for various generalized linear models with $X \in \mathbb{R}^{200\times 20000}$. Predictors are autocorrelated through an $\mathrm{AR}(1)$ process with correlation ρ.

Algorithms

The original strong rule paper (Tibshirani et al. 2012) presents two strategies for using the screening rule. For SLOPE, we have two **slightly** modified versions of these algorithms

strong set algorithm

initialize $\mathcal{E}$ with strong rule set

1. fit SLOPE to predictors in $\mathcal{E}$
2. check KKT criteria against $\mathcal{E}^C$; if there are any failures, add predictors that fail the check to $\mathcal{E}$ and go back to 1

Algorithms

The original strong rule paper (Tibshirani et al. 2012) presents two strategies for using the screening rule. For SLOPE, we have two **slightly** modified versions of these algorithms

strong set algorithm

initialize $\mathcal{E}$ with strong rule set

1. fit SLOPE to predictors in $\mathcal{E}$
2. check KKT criteria against $\mathcal{E}^C$; if there are any failures, add predictors that fail the check to $\mathcal{E}$ and go back to 1

previous set algorithm

initialize $\mathcal{E}$ with ever-active predictors

1. fit SLOPE to predictors in $\mathcal{E}$
2. check KKT criteria against predictors in **strong** set
   - if there are any failures, include these predictors in $\mathcal{E}$ and go back to 1
   - if there are no failures, check KKT criteria against remaining predictors; if there are any failures, add these to $\mathcal{E}$ and go back to 1

Comparing algorithms

Strong set strategy marginally better for low–medium correlation

Previous set strategy starts to become useful for high correlation

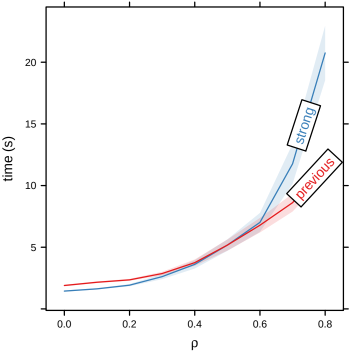


Figure 5: Performance of strong and previous set strategies for OLS problems with varying correlation between predictors.

Limitations

- the unit slope bound is generally very conservative
- does not use second-order structure in any way
- current methods for solving SLOPE (FISTA, ADMM) do not make as good use of screening rules as coordinate descent does (for the lasso)

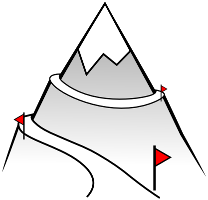
The SLOPE package for R

Strong screening rule for SLOPE has been implemented in the R package SLOPE (https://CRAN.R-project.org/package=SLOPE).

Features include

- OLS, logistic, Poisson, and multinomial models
- support for sparse and dense predictors
- cross-validation
- efficient codebase in C++

Also have a Google Summer of Code student involved in implementing proximal Newton solver for SLOPE this summer.



References I

Małgorzata Bogdan et al. "SLOPE - Adaptive Variable Selection via Convex Optimization". In: *The annals of applied statistics* 9.3 (2015), pp. 1103–1140. ISSN: 1932-6157. DOI: 10.1214/15-AOAS842.

Robert Tibshirani et al. "Strong Rules for Discarding Predictors in Lasso-Type Problems". English. In: *Journal of the Royal Statistical Society. Series B: Statistical Methodology* 74.2 (Mar. 2012), pp. 245–266. ISSN: 1369-7412. DOI: 10/c4bb85.