Combining l^1 Penalization with Higher Moment Stability Constraints in Regression Models

Austin David Brown

December 1, 2018

Motivation

- In statistics and probability theory it is common to impose moment assumptions on a random variable $X : \Omega \to \mathbb{R}^n$ such as $E(\|X\|^k) < \infty$ for $k \in \mathbb{R}$.

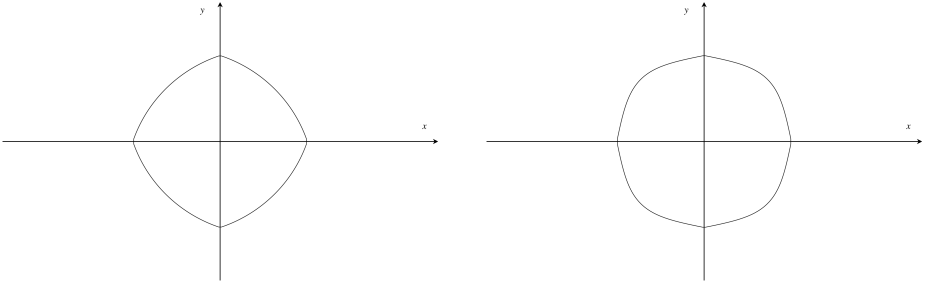
- These constraints correspond to the L^p spaces which allow control over the width and the height of such random variables. This can be interpreted as imposing "stability" on the random variable X.

- If statisticians so freely impose such constraints then we should build a tool to allow scientists and researchers to impose such constraints on their real problems.

- In this project, we build a package implements this idea. The goal is not to create the best predicting method, but instead build a tool that will allow researchers to explore stability and what a "stable" solution to their problem may look like.

Geometric Motivation

Consider for example an Elasticnet [6] penalty $Q(x) = \frac{1}{2}|x| + \frac{1}{2}|y| + \frac{1}{2}|x|^2 + \frac{1}{2}|y|^2 \leq 1$ shown on the left and a new penalty $P(x) = \frac{1}{2}|x| + \frac{1}{2}|y| + \frac{1}{2}|x|^4 + \frac{1}{2}|y|^4 \leq 1$ shown on the right.



It seems reasonable that a scientist may want the option to "bow" out the feasible set even more.

Penalizations

Define these penalizations to try to impose more "stability" for scientists and researchers (try to extend the Elasticnet idea)

$$P(\lambda) = \lambda\alpha_0 \left\|\beta\right\|_1 + \lambda\sum_{k=1}^{5}\alpha_k \left\|\beta\right\|_{2k}^{2k}$$

$$P'(\lambda) = \lambda\alpha_1 \left\|\beta\right\|_1 + \lambda\alpha_2 \left\|\beta\right\|_\infty$$

$$P''(\lambda) = \lambda\alpha_1 \left\|\beta\right\|_1 + \lambda\sum_{k=2}^{10}\alpha_k \left\|\beta\right\|_k$$

where $\lambda \in \mathbb{R}_+$ and α's are convex combinations or use separate tuning parameters instead. These are convex and P is uniformly convex!

Objective

Let

$$L_\lambda(\beta) = \frac{1}{2} \|y - X\beta\|_2^2 + \lambda P(\beta)$$

be the objective with $y \in \mathbb{R}^n$ centered, $X \in M_{n\times p}(\mathbb{R})$ centered, and $\beta \in \mathbb{R}^p$.

- I only had time to do Euclidean loss but this can be extended.
- I only had time to implement the first penalization, so consider only the first one from here on out.
- I now know how to do the others actually.

The Plan

- **glmnet** [5] uses coordinate descent, but the implementation cannot be used and a new algorithm needs to be used.

Algorithm Implementation 1

Algorithm 1: Subgradient Coordinate Method

Choose $\beta^0 \in \mathbb{R}^p$ and tolerance $\delta > 0$;
Set $k \leftarrow 0$
repeat
  Set the step size $h^k \leftarrow \frac{R}{\sqrt{1+k}}$ for some $R > 0$ or use a constant step size.
  Permute $I = \{1, \ldots, p\}$
  for $i \in I$ **do**
    $\beta_i^{k+1} \leftarrow \beta_i^k - h^i g^i$ where $g^i \in (\partial L)_i$
  end
  $k \leftarrow k + 1$
until *Until the loss difference* ΔL *is less than* δ;

Drawbacks

- Not a descent method.
- No good stopping criterion.
- The stopping criterion is also expensive at $O(n^2)$ flops.
- Convergence theory is worse.
- Rarely produces sparse solutions really small values instead.

A Much Better Algorithm

Algorithm 2: Proximal Gradient Coordinate Descent

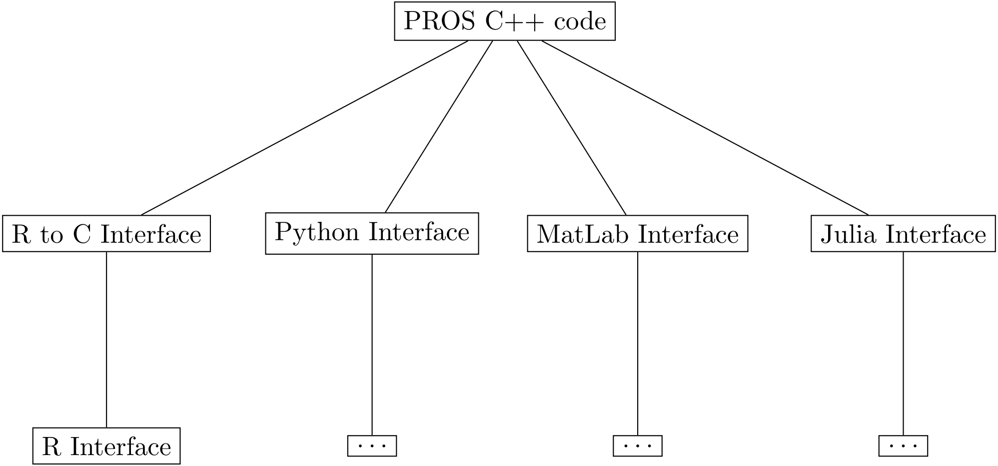
Choose $\beta^0 \in \mathbb{R}^p$ and tolerance $\delta > 0$;
Set $k \leftarrow 0$
repeat
  Set the step size $h^k > 0$ with diminishing or line search.
  Randomly permute $I = \{1, \ldots, p\}$
  for $i \in I$ **do**
    $\beta_i^{k+1} \leftarrow (\mathbf{prox}_{h^k L})_i(\beta_i^k - h^k \langle X_i, y - X\beta \rangle)$
  end
  $k \leftarrow k + 1$
until *Until the Moreau-Yoshida mapping* $M_{h_k, f} < \delta$;

Benefits

- A descent method, so can do line search.
- Good stopping criterion at $O(p^2)$ flops.
- Convergence theory is unchanged from differentiable functions.

Architecture

Written entirely in C++ for speed using Eigen [4] and can be interfaced to many other popular languages.



Simple, Familiar Interface and No Dependencies

A single fit function with prediction

```
> fit <- pros(X, y, alpha, lambda)
> predict(fit, new_X)
```

A cross-validation fit function with prediction

```
> cv <- cv.pros(X, y, alpha)
> predict(cv, new_X)
```

Ran out of time for plotting, but this would be cool.
No dependencies! I did not use RCPP.

Cross-Validation Implementation

Algorithm 3: Warm Start Cross-Validation

Choose a sequence of Langrangian dual variables $\lambda_1, \ldots, \lambda_N$, and initial value β^0.
Order $\lambda_{(1)}, \ldots, \lambda_{(N)}$ descending.
$\beta^{Warm} \leftarrow \beta^0$
for $k \in 1, \ldots, N$ **do**
  $\beta^k \leftarrow$ by Cross-Validation with $\lambda_{(k)}$ warm started with β^{Warm}.
  $\beta^{Warm} \leftarrow \beta^k$
end

Penalized Regression on Steroids

Penalized Regression on Steroids on Github

- The name doesn't fit really since I only implemented 1 penalization
- Talk about python prototype
- Show how to download
- Show the reference manual

Open Questions

How to handle users and step sizes since we want to make it easy for users to use. Backtracking line search?

References

[1] PROS. github.com/austindavidbrown/pros

[2] Neal Parikh and Stephen Boyd. 2014. Proximal Algorithms. Found. Trends Optim. 1, 3 (January 2014), 127-239. DOI=10.1561/2400000003 http://dx.doi.org/10.1561/2400000003

[3] Stephen J. Wright. 2015. Coordinate descent algorithms. Math. Program. 151, 1 (June 2015), 3-34. DOI=10.1007/s10107-015-0892-3 http://dx.doi.org/10.1007/s10107-015-0892-3

[4] Guennebaud, Gaël (2013). Eigen: A C++ linear algebra library (PDF). Eurographics/CGLibs.

[5] Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22. URL http://www.jstatsoft.org/v33/i01/.

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

[7] Yurii Nesterov. 2014. Introductory Lectures on Convex Optimization: A Basic Course (1 ed.). Springer Publishing Company, Incorporated.