Model Selection and Estimation in High-dimensional Generalized Linear Models

Francisco Rivera
Harvard College

Jiafeng (Kevin) Chen
Harvard College

December 10, 2018

1 Introduction

The workhorse for high-dimensional regressions is the ℓ_1 lasso penalty (Tibshirani, 1996). The original lasso is developed for fitting (normal) linear models with the objective[1]

$$\widehat{\boldsymbol{\beta}}_{\text{lasso}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{2n} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1 - \lambda|\beta_0|,\ \lambda > 0. \tag{1}$$

The first paper about lasso in a GLM setting is Park and Hastie (2007): Consider a scalar GLM with likelihood

$$L(y;\theta,\phi) = \exp\left\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right\},$$

where $g(\mathbb{E}[y]) = \eta = \boldsymbol{x}^\intercal\boldsymbol{\beta}$ for some scalar function g. Thus we may consider a direct extension of (1):

$$\widehat{\boldsymbol{\beta}}_{\text{GLM-lasso}} = \arg\min_{\boldsymbol{\beta}} \underbrace{-\frac{1}{n}\sum_{i=1}^{n}\left[y_i\theta(\boldsymbol{\beta})_i - b\left(\theta(\boldsymbol{\beta})_i\right)\right]}_{\ell_n(\boldsymbol{\beta})} + \lambda \|\boldsymbol{\beta}\|_1 - \lambda|\beta_0|, \tag{2}$$

where (2) reduces to (1) if the likelihood is Gaussian and g is the canonical link for the Gaussian model, which is the identity function. Note that the objective (2) is convex if

[1]We consider $\boldsymbol{y}$ an n-vector of response variables whose ith element is $\boldsymbol{y}_i$. We consider the covariate matrix $\boldsymbol{X}$ $(n \times p)$. In most settings, $\boldsymbol{X}$ include an intercept column, $\boldsymbol{x}_0 = \mathbf{1}$, that is not regularized.

we use the canonical link, since the exponential family log-likelihood is concave in θ, and $\theta = \boldsymbol{x}^\intercal \boldsymbol{\beta}$ is linear in $\boldsymbol{\beta}$ for the canonical link.

2 Model Selection with Lasso

Brute-force model selection scales poorly with respect to the number of predictors. That is, when building a model to predict $\boldsymbol{y}$ from a subset of the columns of $\boldsymbol{X} \in \mathbb{R}^{n,p}$ (since not all columns may be predictive), there are 2^p subsets of columns to regress on. Checking all of these models against each other incurs a cost exponential in p and becomes prohibitively expensive in a high-dimensional setting. In class, we tackled this problem with forward-selection and backward-deletion, a greedy algorithm that adds and removes covariates until a local optimal is reached.

We introduced the lasso (1) in Section 1. We first summarize the literature on lasso with OLS in this section. Assume that $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^\star + \boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_n)$. In this case, where possibly $p > n$, the OLS estimate $\widehat{\boldsymbol{\beta}} \in \arg\min_{\boldsymbol{\beta}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2$ may not be unique, but all such solutions have the same fitted values $\widehat{\boldsymbol{y}}_{\mathrm{OLS}} = \boldsymbol{X}\widehat{\boldsymbol{\beta}}$.

Proposition 1. For some constant C, with $r = \operatorname{rank}(\boldsymbol{X})$, then $\mathbb{E}\left[\|X\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\mathrm{OLS}}\|_2^2\right] \leq Cr\sigma^2$.

Proof. Observe that

$$\|\boldsymbol{y} - \widehat{\boldsymbol{y}}\|^2 = \|\boldsymbol{X}\boldsymbol{\beta}^\star - \boldsymbol{X}\boldsymbol{\beta}\| + \|\boldsymbol{\epsilon}\|_2^2 + 2\boldsymbol{\epsilon}^\intercal\left(\boldsymbol{X}\boldsymbol{\beta}^\star - \boldsymbol{X}\boldsymbol{\beta}\right),$$

for any $\boldsymbol{\beta}$. Note that $\|\boldsymbol{y} - \widehat{\boldsymbol{y}}_{\mathrm{OLS}}\|^2 \leq \|\boldsymbol{\epsilon}\|_2^2$. Thus

$$\begin{aligned} \|\boldsymbol{y} - \widehat{\boldsymbol{y}}_{\mathrm{OLS}}\|^2 = \|\boldsymbol{X}\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\mathrm{OLS}}\| + \|\boldsymbol{\epsilon}\|_2^2 + 2\boldsymbol{\epsilon}^\intercal\left(\boldsymbol{X}\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\mathrm{OLS}}\right) \leq \|\boldsymbol{\epsilon}\|_2^2 \\ \implies \|\boldsymbol{X}\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\mathrm{OLS}}\| \leq 2\boldsymbol{\epsilon}^\intercal \underbrace{\left(\widehat{\boldsymbol{y}}_{\mathrm{OLS}} - \boldsymbol{X}\boldsymbol{\beta}^\star\right)}_{\in C(\boldsymbol{X})}. \end{aligned}$$

Thus we can write $\widehat{\boldsymbol{y}}_{\text{OLS}} - \boldsymbol{X}\boldsymbol{\beta}^\star = \boldsymbol{\Phi}\boldsymbol{\nu}$, for some $n \times \text{rank}(\boldsymbol{X}) =: n \times r$ matrix $\boldsymbol{\Phi}$ of orthonormal basis of the columns of $\boldsymbol{X}$. Thus

$$\frac{\boldsymbol{\epsilon}^\intercal(\widehat{\boldsymbol{y}}_{\text{OLS}} - \boldsymbol{X}\boldsymbol{\beta}^\star)}{\|\boldsymbol{X}\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\text{OLS}}\|} = \frac{\boldsymbol{\epsilon}^\intercal\boldsymbol{\Phi}\boldsymbol{\nu}}{\|\boldsymbol{\Phi}\boldsymbol{\nu}\|_2} = (\boldsymbol{\Phi}^\intercal\boldsymbol{\epsilon})^\intercal \frac{\boldsymbol{\nu}}{\|\boldsymbol{\nu}\|} \leq \sup_{\boldsymbol{u}:\|\boldsymbol{u}\|\leq 1} (\boldsymbol{\Phi}^\intercal\boldsymbol{\epsilon})^\intercal u.$$

Using the above inequalities, we show that

$$\|\boldsymbol{X}\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\text{OLS}}\|^2 \leq 4 \left(\sup_{\boldsymbol{u}, \|\boldsymbol{u}\|\leq 1} \underbrace{(\boldsymbol{\Phi}^\intercal\boldsymbol{\epsilon})^\intercal}_{\mathcal{N}(0,\sigma^2\boldsymbol{I}_r)} \boldsymbol{u} \right)^2 \implies \mathbb{E}\left[\|\boldsymbol{X}\boldsymbol{\beta}^\star - \widehat{\boldsymbol{y}}_{\text{OLS}}\|^2\right] \leq 4\sigma^2 r,$$

by properties of the normal distribution. □

Let $\widehat{\boldsymbol{\beta}}_{\text{lasso}}$ be the linear lasso estimate as in (1). Let $\widehat{\boldsymbol{y}}_{\text{lasso}}$ be its fitted value. We can show that a bound similar to Proposition 1 for the lasso, with $O_{\mathbb{P}}(\sqrt{\log(p/n)})$ replacing the $O(r/n)$ dependence (This is the so-called *slow rate* for the lasso, see Tibshirani (2015)). Using the fact that $\widehat{\boldsymbol{\beta}}_{\text{lasso}}$ must achieve a better value than $\boldsymbol{\beta}^\star$ for the objective in (1), and an expansion similar to the one in the proof of Proposition 1, we can show the inequality

$$\frac{1}{2n}\|\boldsymbol{X}\boldsymbol{\beta}^* - \widehat{\boldsymbol{y}}_{\text{lasso}}\|_2^2 \leq \frac{1}{n} \underbrace{\boldsymbol{\epsilon}^\intercal(\widehat{\boldsymbol{y}}_{\text{lasso}} - \boldsymbol{X}\boldsymbol{\beta}^*)}_{\leq \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_\infty \left\|\widehat{\boldsymbol{\beta}}_{\text{lasso}} - \boldsymbol{\beta}^\star\right\|_1 (\text{Hölder})} + \lambda\|\boldsymbol{\beta}^\star\|_1 - \lambda\left\|\widehat{\boldsymbol{\beta}}_{\text{lasso}}\right\|_1.$$

(*Basic inequality* for the lasso)

$$\leq \lambda\left(\left\|\widehat{\boldsymbol{\beta}}_{\text{lasso}} - \boldsymbol{\beta}^\star\right\|_1 + \|\boldsymbol{\beta}^\star\|_1 - \left\|\widehat{\boldsymbol{\beta}}_{\text{lasso}}\right\|_1\right)$$

(On events $\Omega = \left\{n^{-1}\left\|\boldsymbol{X}^T\boldsymbol{\epsilon}\right\|_\infty \leq \lambda\right\}$)

$$\leq 2\lambda\|\boldsymbol{\beta}^\star\|_1.$$

We can show that $\mathbb{P}(\Omega) \geq 1 - p^{-C/2+1}$ if $\lambda = C\sigma\sqrt{\log(p)/n}$ for $C > 2$, assuming standardized columns in $\boldsymbol{X}$ (see Tibshirani, 2015). This leads to the following

Proposition 2 (Slow rate for the lasso)**.** Let $\lambda = C\sigma\sqrt{\log(p)/n}$ for $C > 2$, then with probability at least $1 - p^{-C/2+1}$,

$$\frac{1}{n}\left\|\boldsymbol{X}\boldsymbol{\beta}^{*} - \widehat{\boldsymbol{y}}_{\text{lasso}}\right\|^{2} \leq 2\left\|\boldsymbol{\beta}^{\star}\right\|_{1}\sigma\sqrt{C\frac{\log p}{n}}.$$

How far is the lasso from the optimum? Suppose $k < p$ of $\boldsymbol{\beta}^{\star}$ are nonzero, then if we knew these active predictors (i.e. have an *oracle*), the oracle OLS estimator achieves an error of $k\sigma^2/n$. One can show that the error cannot get better than $\log(p)k\sigma^2/n$ without the oracle, and the best-subset estimator—replacing ℓ_1-norm in (1) with the nonconvex ℓ_0-norm, $\|\boldsymbol{\beta}\|_0 = \sum_i \mathbb{1}\left(\boldsymbol{\beta}_i \neq 0\right)$—achieves this optimum. Under certain conditions, we can show that the lasso achieves a *fast rate* of $\log(p)k\sigma^2/n$, which is the optimal rate. Van de Geer and Bühlmann (2009) provide a review of relevant conditions for achieving the fast rate. These conditions mostly forbid high correlations among the active columns in $\boldsymbol{X}$. Moreover, with even more restrictions, one could show that there exists a λ in (1) for which the corresponding $\widehat{\boldsymbol{\beta}}_{\text{lasso}}$ achieves perfect *support recovery* with high probability:

$$\text{sgn}(\widehat{\boldsymbol{\beta}}_{\text{lasso}}) = \text{sgn}(\boldsymbol{\beta}^{\star}).$$

We defer the more technical discussions to Bühlmann and Van de Geer (2011).

Chapter 6 of Bühlmann and Van de Geer (2011) extends much of the linear results to convex loss functions, covering scalar-valued GLMs with canonical links. Let $\rho_{f_{\boldsymbol{\beta}}}$ be a loss function with respect to prediction function f_{β} that maps data $\boldsymbol{Z}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$ to $\mathbb{R}$. Denote $\mathbf{P}_n\rho = n^{-1}\sum_i \rho(\boldsymbol{Z}_i)$ the empirical risk, $\mathbf{P}\rho = n^{-1}\mathbb{E}\left[\sum_i \rho(\boldsymbol{Z}_i)\right]$ the theoretical risk, and $\mathcal{E}(f) = \mathbf{P}\left(\rho_f - \rho_{f_0}\right)$ the excess risk, where f_0 is a minimizer of $\mathbf{P}\rho$ in some class $\mathbf{F} \supset \mathcal{F} = \{f_{\boldsymbol{\beta}} : \boldsymbol{\beta} \in \mathbb{R}^p\}$. Let $f^0_{\text{GLM}} = \arg\min_{\mathcal{F}} \mathbf{P}\rho$ be the GLM approximation to f_0, which is assumed to have low excess risk. In the GLM context, let $\boldsymbol{\beta}^{\star}$ be the parameters such that $f_{\boldsymbol{\beta}^{\star}} = f^0_{\text{GLM}}$. The lasso in this context produces

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{\mathbf{P}_n\rho_{f_{\boldsymbol{\beta}}} + \lambda\left\|\boldsymbol{\beta}\right\|_1\right\}.$$

Define the *empirical process* to be the gap between theoretical and empirical risks for a $\boldsymbol{\beta}$: $\left\{v_n(\boldsymbol{\beta}) = (\mathbf{P}_n - \mathbf{P})\rho_{f_{\boldsymbol{\beta}}} : \boldsymbol{\beta} \in \mathbb{R}^p\right\}$. The (*Basic inequality* for the lasso) can be generalized to

$$\mathcal{E}\left(f_{\widehat{\boldsymbol{\beta}}}\right) + \lambda \left\|\widehat{\boldsymbol{\beta}}\right\|_1 \leq -\left(v_n(\widehat{\boldsymbol{\beta}}) - v_n(\boldsymbol{\beta}^\star)\right) + \lambda \left\|\boldsymbol{\beta}^\star\right\|_1 + \mathcal{E}\left(f_{\boldsymbol{\beta}^\star}\right)$$

Using the above, we show that, with probability that converges to 1 as $\lambda \to 0$,

$$\mathcal{E}(f_{\widehat{\boldsymbol{\beta}}}) + \lambda\|\widehat{\boldsymbol{\beta}}\|_1 \leq 2\left[\mathcal{E}(f_{\boldsymbol{\beta}^\star}) + 2\lambda\left\|\boldsymbol{\beta}^\star\right\|_1\right]$$

as $\lambda \to 0$, this shows a sort of *consistency* result. For instance, if the true f is a GLM, then $\mathcal{E}(f_{\widehat{\boldsymbol{\beta}}}) \to 0 = 2\mathcal{E}(f_{\boldsymbol{\beta}^\star})$. We can also show (Bühlmann and Van de Geer, 2011, Chapter 6.7) an *oracle inequality* similar to that in the fast rate of the lasso, where under some restrictive regularity conditions, $\mathcal{E}(f_{\widehat{\boldsymbol{\beta}}})$ shrinks at the oracle rate of $O\left(k \log p/n\right)$.

3 Estimation

3.1 Regularization path methods

Efron et al. (2004) gives an efficient estimation procedure for the linear lasso (1) called the Least Angle Regression (LAR), which relies on the fact that the *regularization path*—$\widehat{\boldsymbol{\beta}}_i$ as a function of λ—is piecewise linear in (1). Such a structure is often unavailable in applications like (2). Park and Hastie (2007) mimicks the LAR procedure in GLM estimation. The high-level idea is to note the non-zero coefficients for different penalizations λ and constrain our search to these models. We describe their procedure in this section.

In order to employ this method, we require a method to solve (2). For now, we suppose that we have such a method and can use it as a black box. It is sufficient for us to know that this black box takes in an initial $\boldsymbol{\beta}^{\text{init}}$ and employs a descent-based method to converge toward the optimal $\boldsymbol{\beta}^*$. Section 3 follows-up by describing the inner workings of different approaches.

The lasso penalty induces a sparsity in our optimal $\boldsymbol{\beta}$. This sparsity increases as our penalization term λ becomes bigger. Indeed, if we take $\lambda \to \infty$, then our solution is driven to $\boldsymbol{\beta} \to \mathbf{0}$ because the penalization term is all that matters, and the norm of $\boldsymbol{\beta}$ is minimized at $\mathbf{0}$. The entire domain when $\boldsymbol{\beta} = \mathbf{0}$ is uninteresting, so we initialize our algorithm at $\lambda_{\max}$ which we define as the smallest λ such that there is only one non-zero coefficient.

Once we have initialize λ in our algorithm to $\lambda_{\max}$, we will repeat three steps that Park and Hastie (2007) outline,

1. Determine the step length by which to decrease λ. The intent is to pick this such that the precisely one more coefficient becomes non-zero. Introducing notation, we let λ_k be the λ value we consider in the kth step, and we have to pick $\lambda_k - \lambda_{k+1}$.

2. Using a linear approximation, predict the value of $\boldsymbol{\beta}$ for the decremented λ_{k+1}.

3. Initialize our solver of (2) with the linear estimate as $\boldsymbol{\beta}^{\text{init}}$ and get the precise solution $\boldsymbol{\beta}^{*}$ for the new λ.

doing so until either we have brought λ all the way to 0, or we determine that incremental variables are not improving the model. In order to perform the second step, a linear approximation prescribes that,

$$\boldsymbol{\beta}_{k+1} \approx \boldsymbol{\beta}_k + (\lambda_{k+1} - \lambda_k)\frac{\partial \boldsymbol{\beta}}{\partial \lambda}.$$

Denote by $\boldsymbol{X}_A$ the matrix made up by the columns of $\boldsymbol{X}$ whose corresponding $\boldsymbol{\beta}$ coefficient is non-zero, i.e. in the active set. Furthermore, let $\boldsymbol{W}$ be the diagonal matrix with i^{th} diagonal element given by,

$$\frac{1}{\mathbb{V}(y_i)}\left(\frac{\partial \mu}{\partial \eta}\right)^2$$

which changes because the derivative is evaluated at a different location based on the value of λ, so we define $\boldsymbol{W}_k$ as the value of $\boldsymbol{W}$ corresponding to λ_k. Then, we have that,

$$\boldsymbol{\beta}_{k+1} \approx \boldsymbol{\beta}_k - (\lambda_{k+1} - \lambda_k)(\boldsymbol{W}_A^\top \boldsymbol{W}_k \boldsymbol{X}_A)^{-1} \operatorname{sgn}(\beta_k).$$

Note that in the above equation, we pretend $\boldsymbol{\beta}_k$ and $\boldsymbol{\beta}_{k+1}$ only contain their non-zero components, and thus are not going to have all p dimensions. This makes our algebra simpler and we can do this because the linear approximation will not make a previously zero coefficient non-zero.

Finally, we just need to determine our step size. One simple choice could be to use a constant step size. However, in different domains of λ, the same unit change in λ could have dramatically different implications for the sparsity of the resulting $\boldsymbol{\beta}$. If we were working with a lasso in linear regression, $\boldsymbol{W}$ would not change, and we could pick the step size precisely to coincide with the point at which a coefficient of zero becomes non-zero. Park and Hastie (2007) suggest doing precisely this, with the caveat that because $\boldsymbol{W}$ does change, it will only be an approximation for the GLM case.

3.2 Coordinate descent methods

Unfortunately, Park and Hastie (2007)'s method often scales poorly with the size of the problem. The most popular method—proposed by Friedman, Hastie, and Tibshirani (2010) and implemented in R's glmnet package—is *cyclical coordinate descent* with iteratively reweighted least squares. The idea is to approximate $\ell_n(\boldsymbol{\beta})$ in (2) with a second-order Taylor expansion, either globally for all parameters $\boldsymbol{\beta}$ (for scalar-valued GLMs) or locally with a single parameter $\boldsymbol{\beta}_j$ (for vector-valued GLMs, such as the multinomial logistic regression). Such an approximation yields a quadratic function (in the scalar GLM case)

$$\ell_Q(\boldsymbol{\beta}) = \frac{1}{n} \sum_{i=1}^{n} w_i (y_i - \boldsymbol{x}_i^\intercal \boldsymbol{\beta})^2.$$

We then solve a local penalized least-squares problem:

$$\boldsymbol{\beta} \leftarrow \arg\min_{\boldsymbol{\beta}} \ell_Q(\boldsymbol{\beta}) + \lambda \left\|\boldsymbol{\beta}\right\|_1 . \tag{3}$$

via cyclical coordinate descent, i.e. by iteratively solving

$$\boldsymbol{\beta}_j \leftarrow \arg\min_{\boldsymbol{\beta}_j} \ell_Q(\boldsymbol{\beta}) + \lambda \left\|\boldsymbol{\beta}\right\|_1 , \tag{4}$$

holding all other entries $\boldsymbol{\beta}_{-j}$ fixed. (4) has an analytical solution for the lasso penalty[2]

$$\boldsymbol{\beta}_j \leftarrow \frac{S\left(\sum_{i=1}^N w_i x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right), \lambda\right)}{\sum_{i=1}^N w_i x_{ij}^2}, \; S(t,\gamma) = \operatorname{sgn}(t)\left(|z| - \gamma\right)_+ , \; \tilde{y}_i^{(j)} = \boldsymbol{x}_i^\intercal \boldsymbol{\beta} - x_{ij}\boldsymbol{\beta}_j. \tag{5}$$

We summarize the procedure described above in Algorithm 1.

Algorithm 1 Cyclic coordinate descent algorithm for solving (2) (scalar GLM case) in Friedman, Hastie, and Tibshirani (2010)

```
Initialize $\boldsymbol{\beta}$
for $\lambda$ on regularization path do
    while $\boldsymbol{\beta}$ has not converged do
        Approximate $\ell_n(\beta)$ by $\ell_Q(\beta)$
        while cyclical descent has not converged do
            for $j$ do
                Update $\boldsymbol{\beta}_j$ according to (5)
            end for
        end while
    end while
    $\widehat{\boldsymbol{\beta}}_\lambda \leftarrow \boldsymbol{\beta}$
    Initialize $\boldsymbol{\beta}$ for next iteration to $\widehat{\boldsymbol{\beta}}_\lambda$
end for
```

[2] Friedman, Hastie, and Tibshirani (2010) show a similar expression for the *elastic net* penalty:

$$\lambda P_\alpha(\boldsymbol{\beta}) = \lambda\left(\alpha \left\|\boldsymbol{\beta}\right\|_1 + (1-\alpha)\left\|\boldsymbol{\beta}\right\|_2\right).$$

The machine learning literature slightly alters Algorithm 1 and changes (5) into

$$\boldsymbol{\beta}_j \leftarrow S\left(\boldsymbol{\beta}_j - \left(\nabla_{\boldsymbol{\beta}}\ell_n(\boldsymbol{\beta})\right)_j \kappa^{-1}, \frac{\lambda}{\kappa}\right)$$

for some *learning rate* $1/\kappa$, in keeping with gradient descent. Moreover, Shalev-Shwartz and Tewari (2011) proves a convergence guarantee for stochastic coordinate descent in this fashion, where, instead of cycling through the coordinates of $\boldsymbol{\beta}$, a coordinate is chosen uniformly at random.

Theorem 3. Let $Q(\boldsymbol{\beta})$ be the objective in (2). At iteration T of the first while-loop in a verison of Algorithm 1 with stochastic coordinate descent and gradient updates,

$$\mathbb{E}[Q(\boldsymbol{\beta}_T)] - \mathbb{E}[Q(\widehat{\boldsymbol{\beta}}_{\text{GLM-lasso}})] \leq C\frac{p\kappa}{T+1}$$

for constant C a function of the initial starting value $\boldsymbol{\beta}^{(0)}$, assuming that ℓ_n is differentiable with

$$\ell_n(\boldsymbol{\beta} + \eta \boldsymbol{e}_j) \leq \ell_n(\boldsymbol{\beta}) + \eta \left(\nabla \ell_n\right)_j + \frac{\kappa}{2}\eta^2$$

for all $\eta, \boldsymbol{\beta}, j$.[3]

Corollary 4. The runtime to achieve ϵ expected accuracy is bounded by

$$O\left(\frac{np\kappa}{\epsilon}\left\|\widehat{\boldsymbol{\beta}}_{\text{GLM-lasso}}\right\|_2^2\right).$$

Moreover, Bradley et al. (2011) show that a parallel version of the coordinate gradient descent procedure above where at each iteration, P (possibly duplicate) coordinates are updated in parallel. For correlated features, such parallelism is dangerous, since updating two correlated features simulataneously may over or undercompensate for the gradient direction. Bradley et al. (2011) quantifies the interference due to correlated features and shows that

[3]This condition restricts the choice of κ as a function of the loss criterion.

efficiency increases linearly in the number of parallel processes P so long as $\mathsf{P} \leq \frac{p}{\rho}$ where ρ is the largest modulus of the eigenvalues of $X^\intercal X$.

Coordinate descent methods described above can also become expensive if n is large. The standard machine learning and optimization answer to this problem is to use *stochastic gradient descent*, replacing $\nabla_{\boldsymbol{\beta}} \ell_n(\boldsymbol{\beta})$ with an unbiased estimate $\boldsymbol{g}_i = \nabla_{\boldsymbol{\beta}} \log L(y_i; \boldsymbol{\beta})$, which is the gradient evaluated on a single observation.[4] Shalev-Shwartz and Tewari (2011) consider a mirror descent algorithm in the lasso context, by running stochastic gradient descent on the dual problem and enforcing sparsity in an intelligent manner. Let $\boldsymbol{\gamma} = f(\boldsymbol{\beta})$ be the dual parameter for $\boldsymbol{\beta}$ with an invertible link f. We choose an observation i at random, compute $\boldsymbol{g}_i$, and update

$$\begin{aligned}
&\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} - \eta \boldsymbol{g}_i \\
&\boldsymbol{\gamma}' \leftarrow \boldsymbol{\gamma} - \eta\lambda \operatorname{sgn}(\boldsymbol{\gamma}) && \text{(Decrease } \|\boldsymbol{\beta}\|_1\text{)} \\
&\boldsymbol{\gamma}_j \leftarrow \boldsymbol{\gamma}'_j \mathbb{1}\left(\operatorname{sgn}(\boldsymbol{\gamma}_j) = \operatorname{sgn}(\boldsymbol{\gamma}'_j)\right) && \text{(Maintains sparsity)} \\
&\boldsymbol{\beta} \leftarrow f^{-1}(\boldsymbol{\gamma}).
\end{aligned}$$

The runtime bound for the stochastic mirror descent algorithm in Shalev-Shwartz and Tewari (2011) is

$$O\left(\frac{p \log p}{\epsilon^2} \left\|\widehat{\boldsymbol{\beta}}_{\text{GLM-lasso}}\right\|_2^2\right).$$

We pay the price of the $p \log p$ and ϵ^{-2} dependence, as opposed to p and ϵ^{-1}, in order to achieve the benefit of a n-free runtime.

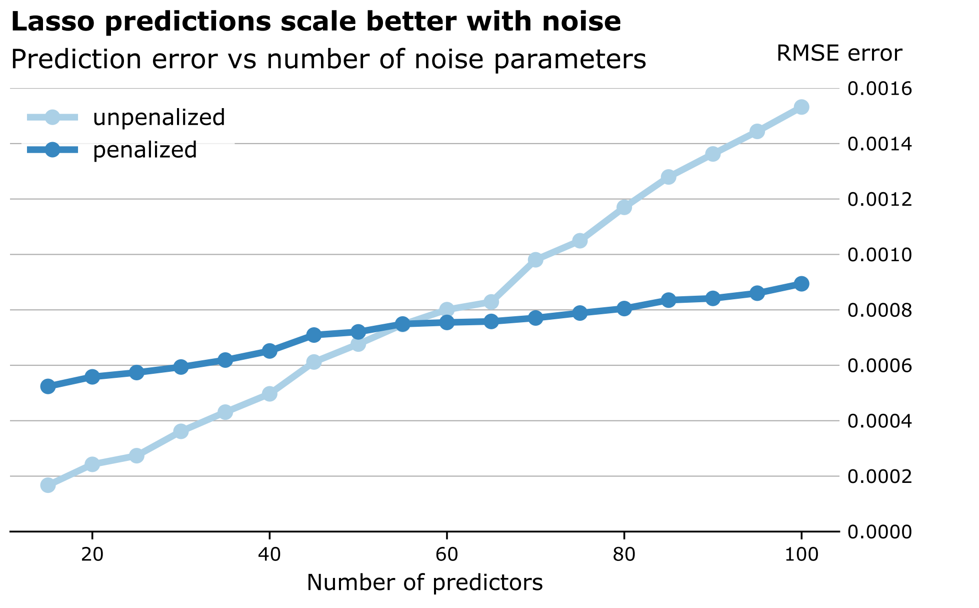
4 Numerical Simulation

Thus far we have covered the theoretical properties of the lasso in a linear setting and the algorithmic implications of extending the lasso to GLMs. Here, we explore the performance

[4] We can replace this with *batched gradient descent* as well, where the gradient estimate is averaging over a batch of observations.

of the lasso in GLMs through numerical simulation. In particular, we focus on a well-specified logistic regression in which we control the generative model and set many of the true coefficients to zero.

We start by creating a correlated matrix X with a randomly generated correlation structure. Then, we generate a binary vector y whose elements are Bernoulli random variables with probabilities determined through a logistic link function with most of the coefficients being zero. Finally, we fit a penalized logistic regression and calculate some summary statistics on its performance.



Note: This summarizes the results of a logistic regression with and without penalization for different number of predictors. The x-axis has the total number of predictors, only 10 of which are predictive. The y-axis is the RMSE between the fit expectations of y_i and the true expectation of y_i

Figure 1: Lasso prediction error

We can repeat this process for different values of p keeping the number of significant coefficients constant. This captures the insertion of noisy parameters into our problem. Then, we can plot the summary statistics against the number of predictors both for our lasso solution as well as a standard un-penalized logistic regression. Figure 1 shows us that while the insertion of covariates with no predictive ability hurts our predictive power in both the lasso and un-penalized cases, the lasso scales better as evidenced by a smaller slope. Of course, the trade-off is that the un-penalized solution does better in cases where more of the parameters are predictive (p is smaller). In our particular experiment, the break-even point is at around 50 covariates for our chosen value of λ.

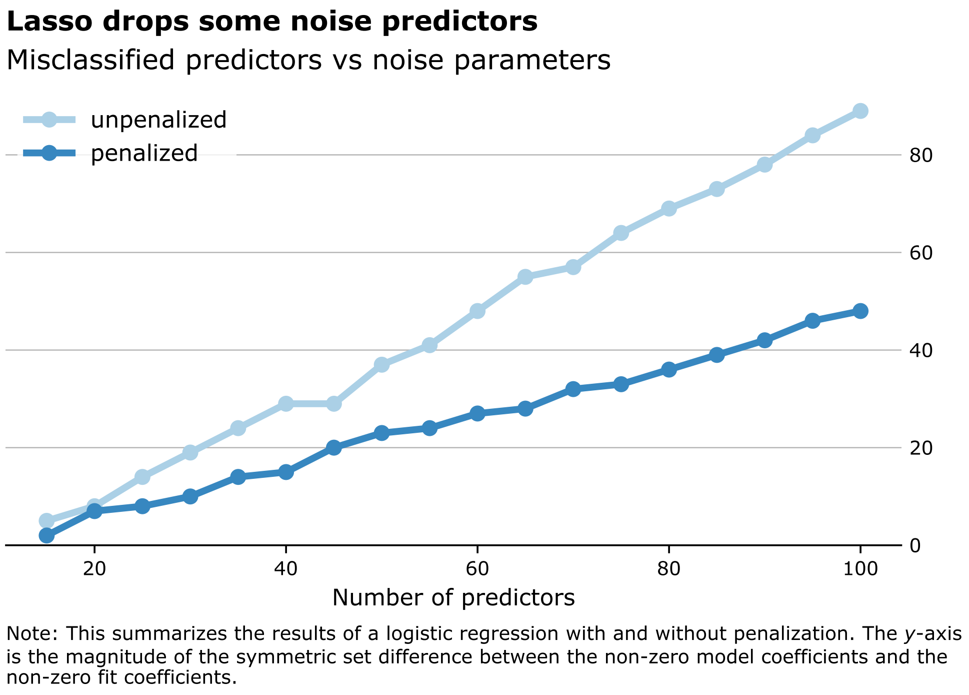


Figure 2: Lasso sparsity identification

We can also understand the implications of the lasso versus the un-penalized GLM for model selection through Figure 2. This shows us that the un-penalized GLM ascribes non-zero (which we here define as bigger in absolute value than 10^{-3}) coefficients to almost all the covariates, even when the true coefficient for most of them is 0. The lasso with our chosen value of λ brings this value down to just about half. More sparsity can be achieved by using a bigger λ, though this can also affect the predictive loss.

Finally, we can also vary the adaptive parameter λ and explore the behavior of the coefficients as we do so. Figure 3 uses R's `glmnet` package to generate the plot of $\log \lambda$ versus the coefficients in the same model as previously. This illustrates the power of the lasso for model selection: the coefficients of the non-predictive terms get brought down to zero distinctly before that of the predictive term (the first 10 terms are predictive, and are

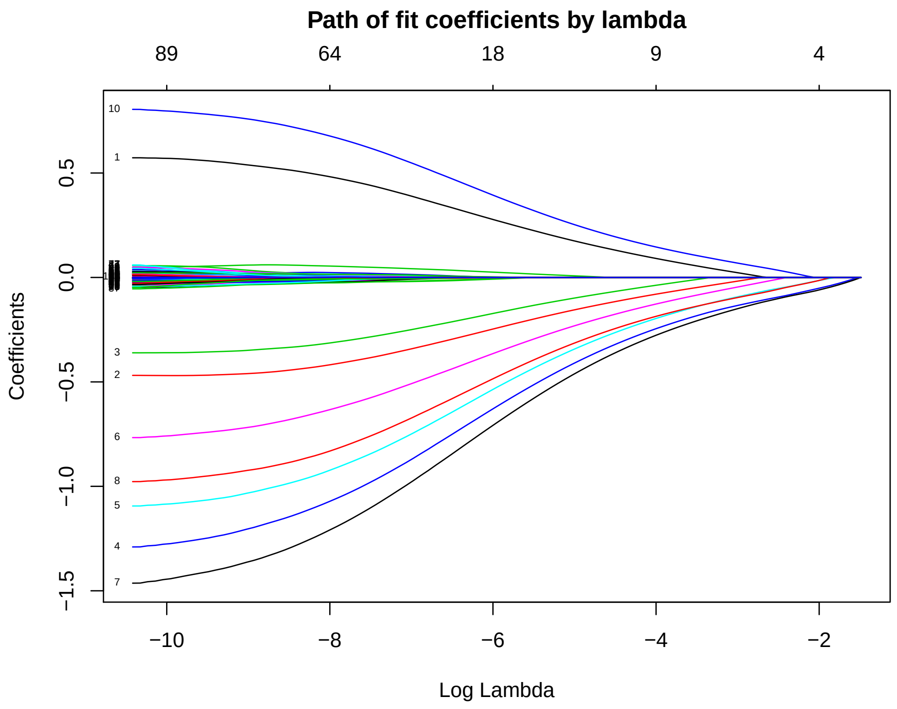


Figure 3: Lasso coefficient path

labelled on the chart).

References

Bradley, Joseph K, Aapo Kyrola, Danny Bickson, and Carlos Guestrin. 2011. "Parallel coordinate descent for l1-regularized loss minimization." Tech. rep.

Bühlmann, Peter and Sara Van de Geer. 2011. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media.

Efron, Bradley, Trevor Hastie, Iain Johnstone, Robert Tibshirani et al. 2004. "Least angle regression." *The Annals of statistics* 32 (2):407–499.

Friedman, Jerome, Trevor Hastie, and Rob Tibshirani. 2010. "Regularization paths for generalized linear models via coordinate descent." *Journal of statistical software* 33 (1):1.

Park, Mee Young and Trevor Hastie. 2007. "L1-regularization path algorithm for generalized linear models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69 (4):659–677.

Shalev-Shwartz, Shai and Ambuj Tewari. 2011. "Stochastic methods for l1-regularized loss minimization." *Journal of Machine Learning Research* 12 (Jun):1865–1892.

Tibshirani, Robert. 1996. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society. Series B (Methodological)* :267–288.

Tibshirani, Ryan. 2015. "Sparsity and the lasso."

Van de Geer, Sara A and Peter Bühlmann. 2009. "On the conditions used to prove oracle results for the Lasso." *Electronic Journal of Statistics* 3:1360–1392.

Appendix A: Code

```python
import scipy

from scipy.special import expit

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import mean_squared_error,

n, p = int(1e5), 100
num_significant = 10

A = np.random.uniform(-1, 1, size=(p,p))
cov = A.T @ A
X = np.random.multivariate_normal(np.zeros(p), cov, n)
beta = np.zeros(p)
beta[:num_significant] = np.random.normal(size=num_significant)
y = np.random.binomial(1, expit(X @ beta)).astype(float)

logistic = LogisticRegression(penalty='l1', C=.1)

logistic.fit(X, y)
fit_coeffs = logistic.coef_.flatten()

mean_squared_error(expit(X @ beta), expit(X @ fit_coeffs))

len(set(range(num_significant)) ^
    set([i for i, val in enumerate(fit_coeffs) if abs(val) > 1e-3]))
```

```r
library(MASS)
library(psych)
library(glmnet)

# generate hyperparameters

n <- 1e4
p <- 100
num.significant <- 10

A <- matrix(runif(p^2)*2-1, ncol=p)
sigma <- t(A) %*% A

# generate random data

X <- mvrnorm(n = n, Sigma = sigma, mu = rep(0, p))
beta <- rep(0, p)
beta[1:num.significant] = rnorm(num.significant)
eta <- as.vector(X %*% beta)

y <- as.numeric(runif(n) < logistic(eta))

# put together into a dataframe
# df <- as.data.frame(cbind(X, y))

l1.model <- glmnet(X, y, family = "binomial", alpha = 1)

plot.glmnet(l1.model, xvar = "lambda", label=TRUE)
title(main = "Path of fit coefficients by lambda", line = 2.5)
par(mar=c(4.5, 4.5, 5, 4))
```