A Comprehensive Study on Group Lasso Performance

James Yang

March 9, 2023

1 Group Lasso Problem

The lasso problem solves the following optimization problem:

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \; f(\beta) + \lambda \|\beta\|_1$$

where $f : \mathbb{R}^p \to \mathbb{R}$ is a convex function and $\lambda > 0$. Given data $X \in \mathbb{R}^{n \times p}$ and $y \in \mathbb{R}^n$, the most popular choices for f are the Generalized Linear Model (GLM) negative log-likelihoods under the Gaussian and Binomial family:

$$\text{Gaussian:} \quad f(\beta) := \frac{1}{2}\|y - X\beta\|_2^2 \tag{1}$$

$$\text{Binomial:} \quad f(\beta) := \sum_{i=1}^{n} \left(y_i x_i^\top \beta - \log\left(1 + e^{x_i^\top \beta}\right)\right) \tag{2}$$

The celebrated lasso objective gained popularity for its interpretability in regression settings as in the above since it tends to shrink some coefficients to zero exactly.

The group lasso problem [Yuan and Lin, 2006] is an extension of the lasso problem that attempts to capture group structure within the features. Specifically, the group lasso attempts to zero-out *groups of coefficients* rather than a single coefficient. The interpretation is that if a group of coefficients is zero, the whole group is uninformative in predicting y [Meier et al., 2008]. This overcomes a major fallback of lasso with categorical features since lasso selects individual categorical feature (factor) instead of a whole group of features (e.g. one-hot encoded factors).

James: Cite early literature of group lasso.

James: Cite GWAS and LD stuff that it makes a lot of sense to group genes.

In preparation of posing the group lasso problem, we begin with some notation. Let $1 \leq G \leq p$ be the number of groups among the p features. Let p_i be the group size of group i for each $1 \leq i \leq G$ such that $\sum_{i=1}^{G} p_i = p$. Define $\beta = [\beta_1 \, \beta_2 \, \dots \, \beta_G]^\top \in \mathbb{R}^p$ where each $\beta_i \in \mathbb{R}^{p_i}$. The group lasso problem is to minimize the following objective:

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \; f(\beta) + \lambda \sum_{j=1}^{G} \|\beta_j\|_2 \tag{3}$$

for some convex function $f : \mathbb{R}^p \to \mathbb{R}$ and $\lambda > 0$. Similar to the lasso, given data X and y, we may use the same Gaussian and Binomial log-likelihoods (1), (2), respectively, for f.

2 Optimizer Algorithm

In this section, we discuss the algorithm to solve the optimization problem (3). Let $X \in \mathbb{R}^{n \times p}$, $y \in \mathbb{R}^n$ be the feature matrix and response vector, respectively. Let G be the number of groups and p_i be the size of group i for $1 \leq i \leq G$. Further let $X = [X_1 \, X_2 \, \dots \, X_G]$ where $X_i \in \mathbb{R}^{n \times p_i}$ is the feature matrix corresponding to group i features only.

In general, if the loss function f is of the form $f(\beta) \equiv f(y, X\beta)$, the optimization problem (3) can be reparametrized in terms of $\tilde{\beta} \in \mathbb{R}^p$ where $\tilde{\beta}_i := Q_i^\top \beta_i$ where $Q_i \in \mathbb{R}^{p_i \times p_i}$ is the orthogonal matrix from the eigen-decomposition of $X_i^\top X_i = Q_i D_i Q_i^\top$ for each $i = 1, \ldots, G$. Concretely, it suffices to solve

$$\underset{\tilde{\beta} \in \mathbb{R}^p}{\text{minimize}} \; f(y, \tilde{X}\tilde{\beta}) + \lambda \sum_{j=1}^{G} \|\tilde{\beta}_j\|_2 \tag{4}$$

where $\tilde{X} = [X_1 Q_1 \; X_2 Q_2 \; \ldots \; X_G Q_G]$. Note that this was only possible because the group lasso penalty is invariant under the transformation from $\beta \to \tilde{\beta}$. Then, given a solution $\tilde{\beta}^*$ to the new problem (4), we have the solution to the original problem β^* given by $\beta_i^* := Q_i^\top \tilde{\beta}_i^*$. Hence, without loss of generality, we assume that our feature matrix X is of the form $\tilde{X}$. In particular, we assume that $X_i^\top X_i = D_i$ is diagonal for all $1 \le i \le G$. We drop all tilde for notational simplicity.

Computationally, it may seem like we must construct $\tilde{X}$ from X before solving (4). However, we will shortly see that the fitting algorithm can be amended to dynamically compute the necessary components of $\tilde{X}$. In particular, we only need access to $\tilde{X}_i$, D_i, and $\tilde{X}_i^\top \tilde{X}_j$ for $i = 1, \ldots, G$ and j in the active set. In comparison to other proposed fitting procedures, we believe that the extra cost to compute $\tilde{X}_i$, D_i, and $\tilde{X}_i^\top \tilde{X}_j$ is comparable to the cost of the other procedures. For example, Yuan and Lin [2006], Meier, Van De Geer, and Böhlmann [2008] rely on the fact that each group feature matrix X_i is full-rank and therefore admits a QR decomposition with an invertible R. Under this assumption, they solve (4) with $\tilde{X} = [Q_1 \, Q_2 \, \ldots \, Q_G]$ and transform the solution back to the original scale by applying R_i^{-1} to the ith group of coefficients. Note that these other approaches also apply a one-time decomposition of the group feature matrices. The benefit of this approach is that the algorithm is aided by a simple closed-form expression as part of the blockwise coordinate descent algorithm. However, there are two major downsides to this approach. The first is that, while the aforementioned authors primarily focused on the application of group lasso on categorical data such as one-hot encoded data that easily satisfy the full-rank assumption, this assumption may easily be violated in other applications. For example, . The second is that this method does not simplify computations when we add a ridge penalty as well (i.e. elastic net), since in general, it introduces a Tikhonov regularization in the transformed problem.

James: Reference to later section?

James: mention GWAS; small n, bigger group size

Similar to other approaches [Yuan and Lin, 2006, Meier et al., 2008, Tseng, 2001, Liang et al., 2022], we use blockwise coordinate descent for its excellent speed and simplicity. However, we take a completely different approach to fitting each block update. The essence of implementing a highly performant optimizer lies in solving each block update (6) as fast as possible. Aside from the orthonormalization method described above as in Yuan and Lin [2006], Meier, Van De Geer, and Böhlmann [2008], many works in literature that describe group lasso fitting procedures use the general descent method called Proximal Gradient Descent (PGD), or Iterative Soft-Thresholding Algorithm (ISTA) in this context, to solve (6) [Liang et al., 2022, Beck and Teboulle, 2009, Klosa et al., 2020, Wright et al., 2009, Loris, 2009, Hastie et al., 2016, O'Donoghue and Candés, 2015]. A Nesterov acceleration can be applied on ISTA to get Fast-ISTA (FISTA) [Beck and Teboulle, 2009]. Further, O'Donoghue and Candés [2015] showed that mild improvements can be made with an adaptive restarting strategy. Section 2.1 describes the blockwise coordinate descent algorithm. Section 2.2 describes the PGD (ISTA) procedure and its variations. Sections 2.3 and 2.4 describe our proposed method to solve (6).

2.1 Blockwise Coordinate Descent

The (convex) optimization problem (4) has a separable regularization, so blockwise coordinate descent can be applied Tseng [2001]. Hence, we iterate through each block $j = 1, \ldots, G$, keeping all other blocks fixed, and solve the following problem:

$$\underset{\beta_j \in \mathbb{R}^{p_j}}{\text{minimize}} \; f(y, X\beta) + \lambda \|\beta_j\|_2 \tag{5}$$

until convergence is reached. The final solution vector β then solves (4).

For the remainder of the paper, we only discuss the Gaussian loss (1). For a general f, we may apply Iterative Reweighted Least Squares (IRLS) by iteratively reweighting the data and solving (4) with the Gaussian loss. The Gaussian loss yields the optimization problem:

James: Cite IRLS. Yuan and Meier?

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \ \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=1}^{G} \|\beta_j\|_2$$

with the block update for group j as:

$$\underset{\beta_j \in \mathbb{R}^{p_j}}{\text{minimize}} \ \frac{1}{2}\beta_j^\top D_j \beta_j - (X_j^\top y - \sum_{i \neq j} X_j^\top X_i \beta_i)^\top \beta_j + \lambda \|\beta_j\|_2$$

For notational simplicity, define $v_j := X_j^\top y - \sum\limits_{i \neq j} X_j^\top X_i \beta_i$. Then, dropping the subscript j, each block update is of the form:

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \ \frac{1}{2}\beta^\top D \beta - v^\top \beta + \lambda \|\beta\|_2 \tag{6}$$

where $D \in \mathbb{R}^{p \times p}$ is diagonal with non-negative entries, $v \in \mathbb{R}^p$ is any vector, and $\lambda > 0$.

2.2 Proximal Gradient Descent (PGD)

In this section, we describe the PGD method, or ISTA, to solve the block update (6). The progression is similar to that of Hastie et al. [2016]. Define

$$f(\beta) = \frac{1}{2}\beta^\top D \beta - v^\top \beta$$

as the convex (Gaussian) loss function in (6). ISTA has guaranteed convergence so long as the step-size is chosen properly. Any step-size ν can be chosen such that $\nu \leq \frac{1}{L}$ where L is the Lipschitz constant for ∇f. Since

$$\begin{aligned} &\nabla f(\beta) = D\beta - v \\ \implies &\|\nabla f(\beta) - \nabla f(\tilde{\beta})\|_2 = \|D(\beta - \tilde{\beta})\|_2 \leq \|D\|_{\text{op}} \|\beta - \tilde{\beta}\|_2 \end{aligned}$$

we have $L \equiv \|D\|_{\text{op}} \equiv \lambda_{\max}(D)$ as the largest eigenvalue of D. In particular, since D is diagonal, $L \equiv \max\limits_i D_{ii}$.

The ISTA procedure is given in Algorithm 1. By applying a standard Nesterov acceleration [Beck and Teboulle, 2009], we have the FISTA procedure given in Algorithm 2. Additionally, we may perform adaptive restarts based on the proximal gradient updates [O'Donoghue and Candés, 2015] for even faster convergence as outlined in Algorithm 3.

Algorithm 1: ISTA

```
Data: $D$, $v$, $\lambda$, $\beta^{(0)}$
$\nu \leftarrow \left(\max_i D_{ii}\right)^{-1}$;
while not converged do
    $w \leftarrow \beta^{(k-1)} + \nu\left(v - D\beta^{(k-1)}\right)$;
    $\beta^{(k)} \leftarrow \left(1 - \frac{\nu\lambda}{\|w\|_2}\right)_+ w$;
```

Note that the computational cost of Algorithms 1 to 3 are all $O(pk)$ where k is the number of iterations until convergence. Since we must output a vector in $\mathbb{R}^p$, any optimizer must necessarily have $O(p)$ operations. Hence, the optimal optimizer of complexity $O(pk)$ must prioritize lowering the number of iterations until convergence. While each algorithm is an improvement from the previous, there is still much improvement left. Section 2.3 shows a novel approach to solving (6) with a significant decrease in iterations while maintaining $O(p)$ operations per iteration.

Algorithm 2: FISTA

```
Data: $D$, $v$, $\lambda$, $\beta^{(0)}$
$\nu \leftarrow \left(\max_i D_{ii}\right)^{-1}$;
$\eta^{(1)} \leftarrow \beta^{(0)}$;
$t_1 = 1$;
while not converged do
    $w \leftarrow \eta^{(k)} + \nu\left(v - D\eta^{(k)}\right)$;
    $\beta^{(k)} \leftarrow \left(1 - \frac{\nu\lambda}{\|w\|_2}\right)_+ w$;
    $t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$;
    $\eta^{(k+1)} \leftarrow \beta^{(k)} + \frac{t_k - 1}{t_{k+1}}(\beta^{(k)} - \beta^{(k-1)})$;
```

Algorithm 3: FISTA with Adaptive Restart

```
Data: $D$, $v$, $\lambda$, $\beta^{(0)}$
while not converged do
    Carry out Algorithm 2 with inputs $D$, $v$, $\lambda$, $\beta^{(0)}$ until
        $$\nabla f(\eta^{(k)})^\top(\beta^{(k)} - \beta^{(k-1)}) > 0 \tag{7}$$
    ;
    if (7) holds at step $k$ then
        Reset $\beta^{(0)} \leftarrow \beta^{(k)}$;
```

2.3 Newton's Method

In this section, we propose a novel and simple method to solve (6). By solving the sub-gradient problem of (6), we see that the solution $\beta^\star$ must satisfy

$$\beta = \begin{cases} \left(D + \frac{\lambda}{\|\beta\|_2} I\right)^{-1} v, & \|v\|_2 > \lambda \\ 0, & \|v\|_2 \leq \lambda \end{cases} \tag{8}$$

Without loss of generality, we consider the first case when $\beta^\star \neq 0$. Without knowing $\|\beta^\star\|_2$, there is no closed-form solution for $\beta^\star$. In an endeavor to find an expression for $\|\beta^\star\|_2$, we take the ℓ_2 norm on both sides to get that $\|\beta^\star\|_2$ must satisfy

$$\sum_{i=1}^{p} \frac{v_i^2}{(D_{ii}\|\beta\|_2 + \lambda)^2} = 1 \tag{9}$$

Unfortunately, there is no closed-form solution for (9) either. However, we will shortly show that there is an efficient algorithm to numerically solve (9). In turn, once we find $\|\beta^\star\|_2$, we may substitute it into (8) to get the full solution $\beta^\star$ in $O(p)$ time. We note in passing that (8) and (9) were previously discovered [Hastie et al., 2016], however, there is no mention of how to quickly solve for $\beta^\star$ or $\|\beta^\star\|_2$.

We now discuss how to solve (9). First, define $\varphi : [0, \infty) \to \mathbb{R}$ by

$$\varphi(h) := \sum_{i=1}^{p} \frac{v_i^2}{(D_{ii}h + \lambda)^2} - 1$$

so that (9) is now a root-finding problem for φ. Upon differentiating φ,

$$\frac{d\varphi(h)}{dh} = -2\sum_{i=1}^{p} \frac{v_i^2 D_{ii}}{(D_{ii}h + \lambda)^3} \leq 0$$

where the inequality is strict if not all $v_i^2 D_{ii} = 0$. Note that if all $v_i^2 D_{ii} = 0$, then we must have that $\|v\|_2 \leq \lambda$. Otherwise,

$$\varphi(h) = \sum_{i:D_{ii}=0} \frac{v_i^2}{\lambda^2} - 1 = \lambda^{-2} \sum_{i=1}^{p} v_i^2 - 1 > 0$$

In particular, φ is constant and strictly positive so it has no roots. So, if $\|v\|_2 > \lambda$, then there exists a vector β where $\varphi(\|\beta\|_2) = 0$, which is a contradiction. Hence, without loss of generality, we may assume φ' is strictly negative, so that φ is strictly decreasing. Consequently, since $\varphi(0) > 0$ by hypothesis, there exists a (unique) root. Further, it is easy to see that φ is convex since it is a sum of convex functions.

Since φ is convex, this suggests solving (9) via a one-dimensional Newton's Method. Specifically, with $h^{(0)}$ as the initial starting point, for $k \geq 1$,

$$h^{(k)} = h^{(k-1)} - \frac{\varphi(h^{(k-1)})}{\varphi'(h^{(k-1)})} \tag{10}$$

We claim that Newton's Method is guaranteed to converge for any initial point $h^{(0)}$. Indeed, for every $k \geq 1$, by convexity of φ,

$$\varphi(h^{(k)}) \geq \varphi(h^{(k-1)}) + \varphi'(h^{(k-1)})(h^{(k)} - h^{(k-1)}) = 0$$

Along with (10), this shows that $h^{(k)}$ is an increasing sequence for $k \geq 1$ and bounded above by $h^\star$, the root of φ, by monotonicity. Hence, $h^{(k)}$ converges to some limit $h^{(\infty)}$. From (10), taking limit as $k \to \infty$ and using that φ' is non-zero,

$$h^{(\infty)} = h^{(\infty)} - \frac{\varphi(h^{(\infty)})}{\varphi'(h^{(\infty)})} \implies \varphi(h^{(\infty)}) = 0$$

which shows that $h^{(\infty)}$ is the root of φ. In summary, Newton's Method gives *guaranteed convergence* to the root.

Since φ is convex, we also have quadratic convergence rate and each iteration costs $O(p)$. In principle, our setting is one of the best settings for Newton's Method to succeed. While FISTA (Algorithm 2) and the adaptive restarted version (Algorithm 3) also give the same convergence rate, Figure shows a clear improvement for the Newton's Method in practice. One very important benefit of Newton's Method is that the optimization domain remains one-dimensional *regardless of the input dimensions*. However, FISTA optimizes over $\beta \in \mathbb{R}^p$ directly, so its performance is heavily impacted by the dimension.

James: add figure ref

It is widely known that descent methods including Newton's Method is highly sensitive to the initialization. The question that remains is: how to choose the initial starting point $h^{(0)}$? Despite the convergence guarantee irrespective of the choice of $h^{(0)}$, it is nonetheless ideal to choose an initial point close to the root for an even faster convergence. We recommend selecting any $h^{(0)}$ such that $\varphi(h^{(0)}) \geq 0$. For example, $h^{(0)} \equiv 0$ is sufficient since by definition $\varphi(0) > 0$. If $h^{(0)}$ is such that $\varphi(h^{(0)}) < 0$, then it is possible that $h^{(1)} < 0$. In fact, due to the flat tail of φ, it is extremely common for $h^{(1)}$ to be negative. In this case, after projecting back to $[0, \infty)$ (i.e. setting $h^{(1)} = 0$), we arrive at the same sequence as if we had started with $h^{(0)} = 0$. For this reason, it is almost always better to set the initial point such that $\varphi(h^{(0)}) \geq 0$. Although $h^{(0)} = 0$ is a valid choice, it may also lead to large number of iterations, especially if λ is small and D is close to being semi-definite. In Section 2.4, we describe our most performant and robust modification to the vanilla Newton's Method that reduces iterations in nearly all cases.

2.4 Newton with Adaptive Bisection Starts (Newton-ABS)

In this section, we improve the Newton's Method described in Section 2.3. Specifically, we first discuss how to find lower and upper bounds $h_\star$, $h^\star \in [0, \infty)$, respectively, such that the root lies in $[h_\star, h^\star]$. We then discuss an adaptive bisection method that preceeds the Newton's Method to find a clever initial point $h^{(0)}$.

We first begin with the derivation of $h_\star$ and $h^\star$. Note that the root-finding problem for φ is equivalent to finding the largest value $h > 0$ such that $\varphi(h) \geq 0$, or

$$\sum_{i=1}^{p} \frac{v_i^2}{(D_{ii}h + \lambda)^2} \geq 1$$

Let $a(h), b(h) \in \mathbb{R}^p$ be defined by $a_k(h) := D_{kk}h + \lambda$ and $b_k(h) := \frac{|v_k|}{a_k(h)}$ for each $k = 1, \ldots, p$. Then, by Cauchy-Schwarz,

$$\|v\|_1 := \sum_k |v_k| = \sum_k a_k(h) b_k(h) \leq \|a(h)\|_2 \|b(h)\|_2$$

Hence, if $\|a(h)\|_2 \leq \|v\|_1$, then $\varphi(h) \geq 0$. We see that $\|a(h)\|_2 \leq \|v\|_1$ if and only if

$$\sum_{i=1}^{p} (D_{ii}h + \lambda)^2 \leq \|v\|_1^2$$

This inequality can be solved for h using the quadratic formula. Let $\tilde{h}$ be the solution. Then, we have a potentially tighter lower bound $h_\star := \max(\tilde{h}, 0)$ than zero.

Next, we discuss $h^\star$. Similar to $h_\star$, we wish to find the smallest h such that

$$\sum_{i=1}^{p} \frac{v_i^2}{(D_{ii}h + \lambda)^2} \leq 1$$

Approximating this problem, since

$$\begin{aligned} \sum_{i=1}^{p} \frac{v_i^2}{(D_{ii}h + \lambda)^2} &= \sum_{i:D_{ii}=0} \frac{v_i^2}{\lambda^2} + \sum_{i:D_{ii}>0} \frac{v_i^2}{(D_{ii}h + \lambda)^2} \\ &\leq \sum_{i:D_{ii}=0} \frac{v_i^2}{\lambda^2} + h^{-2} \sum_{i:D_{ii}>0} \frac{v_i^2}{D_{ii}^2} \qquad (11) \end{aligned}$$

by setting

$$h^\star := \sqrt{\frac{\sum\limits_{i:D_{ii}>0} \frac{v_i^2}{D_{ii}^2}}{1 - \lambda^{-2} \sum\limits_{i:D_{ii}=0} v_i^2}}$$

we have that $\varphi(h^\star) \leq 0$. This shows that the root must lie in $[h_\star, h^\star]$.

In practice, φ may decay incredibly rapidly near $h \approx 0$ and have an extremely flat tail. To protect against slow convergence of Newton's Method in this case, we may use $h_\star$ and $h^\star$ to first perform a bisection method to find the initial starting point. The key idea is to use bisection first for a few iterations to avoid the region of slow changes in the Newton iterations. Then, once sufficiently close to the root, we apply the Newton's Method for the fast guaranteed convergence. Although one may use a simple bisection method of splitting the interval $[h_\star, h^\star]$ in halves until sufficiently small, we propose an *adaptive bisection method* that can significantly reduce the number of bisections.

James: refer to a figure?

We describe the adaptive bisection method. Ideally, we would like to know if the root is closer to $h_\star$ or $h^\star$. If we believe that the root is much closer to $h_\star$, we do not have to bisect $[h_\star, h^\star]$ at the mid-point, but perhaps at a point closer to $h_\star$. Likewise, if the root were much closer to $h^\star$, we would like to bisect at a point closer to $h^\star$. Since we do not know the root, we would like to quantify a *prior* of the root being closer to $h_\star$. With this in mind, note that in (11), we approximated the problem of finding the smallest h such that $\varphi(h) \leq 0$ by solving the problem for an upper bound of φ. Then, the more accurate the approximation, the closer the approximate solution $h^\star$ is to the root. The approximation essentially came from using the fact that $D_{ii}h^\star \leq D_{ii}h^\star + \lambda$ for $D_{ii} > 0$. This motivates us to consider the worst approximation error (rate)

$$w := \max_{i:D_{ii}>0} \frac{\lambda}{D_{ii}h^\star + \lambda} = \frac{\lambda}{D_\star h^\star + \lambda} \in (0, 1)$$

where $D_\star := \min_{i:D_{ii}>0} D_{ii}$. If w is small, the approximation in (11) is tight, which implies that the root is close to $h^\star$. Hence, $1 - w$ represents the prior that the root is close to $h^\star$. We bisect at the new point $h := wh_\star + (1 - w)h^\star$. If $\varphi(h) \geq 0$, we use $h^{(0)} := h$ as the initial point for the Newton Method. Otherwise, we set $h^\star := h$ and repeat the argument. Algorithm 4 summarizes this procedure, which we call *Newton with Adaptive Bisection Starts* (Newton-ABS), in its most basic form.

Algorithm 4: Newton with Adaptive Bisection Starts (Newton-ABS)

```
Data: $D$, $v$, $\lambda$, $\epsilon$
Compute $h_\star$ that solves: $\sum_{i=1}^{p} (D_{ii}h + \lambda)^2 \leq \|v\|_1^2$;
Compute $h^\star := \sqrt{\frac{\sum_{i:D_{ii}>0} \frac{v_i^2}{D_{ii}^2}}{1-\lambda^{-2} \sum_{i:D_{ii}=0} v_i^2}}$;
$D_\star \leftarrow \min_{i:D_{ii}>0} D_{ii}$;
$w \leftarrow \frac{\lambda}{D_\star h^\star + \lambda}$;
$h \leftarrow wh_\star + (1 - w)h^\star$;
while $\varphi(h) < 0$ and $|\varphi(h)| > \epsilon$ do
    $h^\star \leftarrow h$;
    $w \leftarrow \frac{\lambda}{D_\star h^\star + \lambda}$;
    $h \leftarrow wh_\star + (1 - w)h^\star$;
if $\varphi(h) \geq 0$ then
    Apply Newton's Method starting at $h^{(0)} = h$ and get optimal $\tilde{h}$;
$\beta^\star \leftarrow (D + \lambda\tilde{h}^{-1}I)^{-1}v$;
return $\beta^\star$;
```

Algorithm 4 can be further optimized. For example, if $h^\star - h_\star$ is below some threshold (e.g. 0.1), then we may skip bisection entirely and start Newton's Method at $h^{(0)} = h_\star$. This is because the adaptive bisection may move too slowly if the range is too small. From experimentation, this happens relatively often. On a similar note, one may also enforce enough movement towards $h_\star$ by taking the max of w with a minimal probability (e.g. 0.05). This will ensure that at least some proportion of $h_\star$ is taken if the prior is too strongly suggestive that the root is close to $h^\star$. The idea is that it is always better to overshoot towards $h_\star$ such that φ is non-negative, so that Newton's Method can quickly converge down to the root, than to slowly bisect to the root.

3 Benchmark

In this section, we study a wide array of benchmarks.

3.1 Proximal Gradient Descent vs. Newton

In this section, we compare the algorithms presented in Sections 2.2 to 2.4 that solve the block update (6). Recall the objective to solve is

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \; \frac{1}{2}\beta^\top D\beta - v^\top \beta + \lambda\|\beta\|_2$$

where D is a diagonal matrix with non-negative entries, v any vector, and $\lambda > 0$.

Figure 1 shows a set of comparisons of ISTA, FISTA, FISTA with adaptive restart (FISTA-ADA), Newton, Newton-Brent, and Newton-ABS. Newton-Brent works exactly the same as Newton-ABS except the adaptive bisection is replaced with Brent's method. For all scenarios, the diagonal of D was generated uniformly from $[0.8, 2]$. In Figure 1b, 1% of the entries were regenerated uniformly from $[1 \times 10^{-14}, 1 \times 10^{-8}]$. In Figure 1c, 20% of the entries were set exactly to zero and 10% were regenerated from $[1 \times 10^{-14}, 1 \times 10^{-8}]$. The vector v was generated from $\mathcal{N}(0, D)$. Finally, Figures 1a to 1c used $\lambda = 1 \times 10^{-1}$ while Figure 1d used $\lambda = 1 \times 10^{-4}$ with positive definite D. Each setting measures total time, relative time to Newton-ABS, number of iterations until convergence, and accuracy. All methods converged before max iterations was reached. The accuracy comparisons show that all methods properly converged, with Newton-ABS generally having smaller errors. The timings had large variance for small number of features, since most of the time was spent in the scripting language to dispatch the function calls. Overall, the Newton methods clearly outshine the PGD methods in terms of speed. The PGD methods generally use around 100 or even 1000 iterations, however Newton method is consistently using 20 to 30 iterations, Newton-Brent using 6 to 10 iterations, and Newton-ABS using 3 to 4 iterations. The relative speed difference shows that Newton-ABS is anywhere from 3 to 100 times faster than the PGD methods, 1.5 to 5 times faster than Newton, and around 2 times faster than Newton-Brent, where the differences become stark as p increases or regularization level decreases. We also note that the PGD methods and Newton are quite comparable in speed when the regularization is low. Since the blockwise coordinate descent takes a longer time to converge with low regularization and more groups will be active, the block update must especially be fast in this setting. Hence, we expect significant improvement to the overall optimizer with Newton-ABS, as its block update performance seems unaffected.

References

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. doi: 10.1137/080716542. URL https://doi.org/10.1137/080716542.

Trevor Hastie, Rob Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity*. Chapman & Hall, 2016.

J. Klosa, N. Simon, P.O. Westermark, V. Liebscher, and D. Wittenburg. Seagull: lasso, group lasso and sparse-group lasso regularization for linear regression models via proximal gradient descent. *BMC Bioinformatics*, 21(407), 2020. doi: 10.1186/s12859-020-03725-w.

Xiaoxuan Liang, Aaron Cohen, Anibal Solón Heinsfeld, Franco Pestilli, and Daniel J. McDonald. sparsegl: An r package for estimating sparse group lasso, 2022. URL https://arxiv.org/abs/2208.02942.

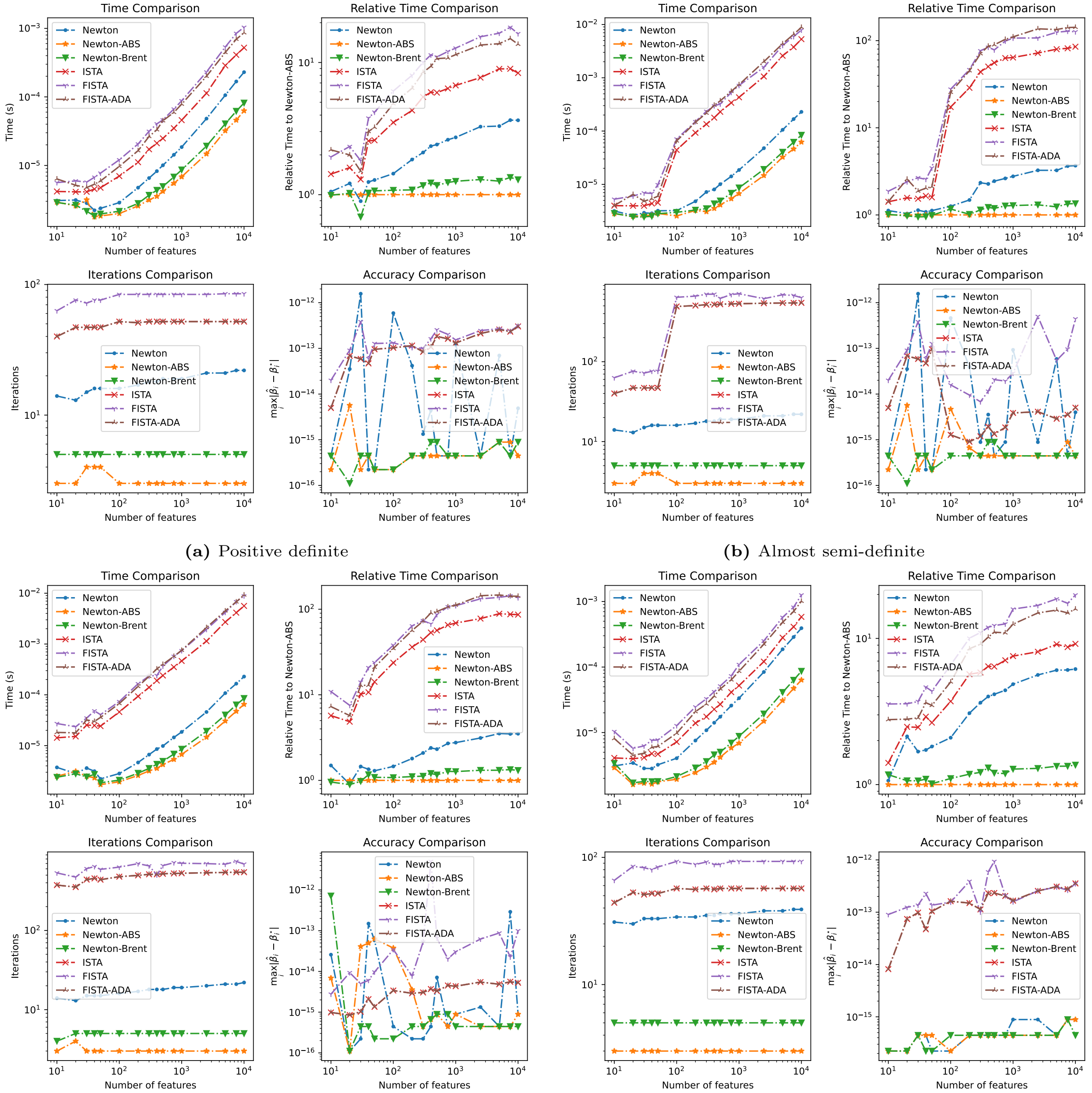
Ignace Loris. On the performance of algorithms for the minimization of ℓ_1-penalized functionals. *Inverse Problems*, 25(3):035008, jan 2009. doi: 10.1088/0266-5611/25/3/035008. URL https://dx.doi.org/10.1088/0266-5611/25/3/035008.

Lukas Meier, Sara Van De Geer, and Peter Böhlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society B*, 70(1):53–71, 2008. doi: 10.1111/j.1467-9868.2007.00627.x.

Brendan O'Donoghue and Emmanuel Candés. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, 15:715–732, 2015. doi: https://doi-org.stanford.idm.oclc.org/10.1007/s10208-013-9150-3.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:475–494, 2001. doi: https://doi-org.stanford.idm.oclc.org/10.1023/A:1017501703105.

Stephen J. Wright, Robert D. Nowak, and MÁrio A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009. doi: 10.1109/TSP.2009.2016892.



(a) Positive definite

(b) Almost semi-definite

(c) Very semi-definite

(d) Positive definite and low regularization

Figure 1: Figures 1a to 1d show a comparison of time, relative time, number of iterations, and accuracy across different algorithms. Figures 1a to 1c fixed $\lambda = 1 \times 10^{-1}$ while Figure 1a fixed $\lambda = 1 \times 10^{-4}$. The difference across Figures 1a to 1c is the degree of singularity of D. Figure 1a used a positive definite D, Figure 1b used a positive definite D with some eigenvalues close to 0 (but not exactly), and Figure 1c used a positive semi-definite D with some eigenvalues exactly at 0, some close to 0, and some sufficiently positive. The accuracy comparisons show that all methods properly converged, with Newton-ABS generally having smaller errors. Overall, the Newton methods clearly outshine the PGD methods in terms of speed with Newton-ABS in the obvious lead.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society B*, 68(1):49–67, 2006. doi: 10.1111/j.1467-9868.2005.00532.x.