Journal of Machine Learning Research 15 (2014) 3447-3481 Submitted 6/13; Revised 5/14; Published 10/14

Alternating Linearization for Structured Regularization Problems

Xiaodong Lin LIN@BUSINESS.RUTGERS.EDU
Department of Management Science and Information Systems
Rutgers University
Piscataway, NJ 08854

Minh Pham PTUANMINH@GMAIL.COM
Statistical and Applied Mathematical Sciences Institute (SAMSI)
Durham, NC 27707

Andrzej Ruszczyński RUSZ@BUSINESS.RUTGERS.EDU
Department of Management Science and Information Systems
Rutgers University
Piscataway, NJ 08854

Editor: S. V. N. Vishwanathan

Abstract

We adapt the alternating linearization method for proximal decomposition to structured regularization problems, in particular, to the generalized lasso problems. The method is related to two well-known operator splitting methods, the Douglas–Rachford and the Peaceman–Rachford method, but it has descent properties with respect to the objective function. This is achieved by employing a special update test, which decides whether it is beneficial to make a Peaceman–Rachford step, any of the two possible Douglas–Rachford steps, or none. The convergence mechanism of the method is related to that of bundle methods of nonsmooth optimization. We also discuss implementation for very large problems, with the use of specialized algorithms and sparse data structures. Finally, we present numerical results for several synthetic and real-world examples, including a three-dimensional fused lasso problem, which illustrate the scalability, efficacy, and accuracy of the method.

Keywords: lasso, fused lasso, nonsmooth optimization, operator splitting

1. Introduction

Regularization techniques that encourage sparsity in parameter estimation have gained increasing popularity recently. The most widely used example is *lasso* (Tibshirani, 1996), where the loss function $f(\cdot)$ is penalized by the ℓ_1-norm of the unknown coefficients $\beta \in \mathbb{R}^p$, to form a modified objective function,

$$\mathcal{L}(\beta) = f(\beta) + \lambda\|\beta\|_1, \quad \lambda > 0,$$

in order to shrink irrelevant coefficients to zero. Many efficient algorithms have been proposed to solve this problem; see Fu (1998); Daubechies et al. (2004); Efron et al. (2004) and Friedman et al. (2007). Some of them are capable of handling massive data sets with tens of thousands of variables and observations.

For many practical applications, physical constraints and domain knowledge may mandate additional structural constraints on the parameters. For example, in cancer research, it may be important to consider groups of interacting genes in each pathway rather than individual genes. In image analysis, it is natural to regulate the differences between neighboring pixels in order to achieve smoothness and reduce noise. In light of these popular demands, a variety of structured penalties have been proposed to incorporate prior information regarding model parameters. One of the most important structural penalties is the *fused lasso* proposed by Tibshirani et al. (2005). It utilizes the natural ordering of input variables to achieve parsimonious parameter estimation on neighboring coefficients. Chen et al. (2010) developed the *graph induced fused lasso* that penalizes differences between coefficients associated with nodes in a graph that are connected. Rudin et al. (1992) proposed the *total variation penalty* for image denoising and deblurring, in a similar fashion to the two-dimensional fused lasso. Similar penalty functions have been successfully applied to several neuroimaging studies (Michel et al., 2011; Grosenick et al., 2011, 2013). More recently, Zhang et al. (2012) applied a generalized version of fused lasso to reconstruct gene copy number variant regions. A general structural lasso framework was proposed by Tibshirani and Taylor (2011), with the following form:

$$\mathcal{L}(\beta) = f(\beta) + \lambda\|R\beta\|_1, \quad \lambda > 0, \tag{1}$$

where R is an $m \times p$ matrix that defines the structural constraints one wants to impose on the coefficients. Many regularization problems, including high dimensional fused lasso and graph induced fused lasso, can be cast in this framework.

When the structural matrix R is relatively simple, as in the original lasso case with $R = I$, traditional path algorithms and coordinate descent techniques can be used to solve the optimization problems efficiently (Friedman et al., 2007). For more complex structural regularization, these methods cannot be directly applied. One of the key difficulties is the non-separability of the nonsmooth penalty function. Coordinate descent methods fail to converge under this circumstances (Tseng, 2001). Generic solvers, such as interior point methods, can sometimes be used; unfortunately they become increasingly inefficient for large size problems, particularly when the design matrix is ill-conditioned (Chen et al., 2012).

In the past decade, many efforts have been devoted to developing efficient optimization techniques for solving regularization problems using structured penalties. Liu et al. (2010) and Ye and Xie (2011) developed a first-order and a split Bregman scheme, respectively, for solving similar class of problems. Chen et al. (2012) proposed a modified proximal technique for the general structurally penalized problems. It is based on a first order approximation of the nonsmooth penalty function, which can become unstable when dimension is high. Meanwhile, several path algorithms have also been proposed to compute the whole regularization path for the general fused lasso problem. Hoefling (2010) developed a path algorithm for solving (1) when the matrix X^TX is nonsingular, where X is the design matrix. This technique is not applicable to cases with large dimension of β and small number of observations, such as gene expression and brain imaging analysis. Tibshirani and Taylor (2011) extended the path algorithm to include all design matrices X, by computing the regularization path of the dual problem. Although fairly general, this version of the path algorithm does not scale well with data dimension, as the knots of the piecewise linear solution path become very dense. Many of the proposed approaches are versions of the *operator splitting methods* or their dual versions, *alternating direction methods* (see, e.g., Boyd et al., 2010, Combettes and Pesquet, 2011, and the references therein). Although fairly general and universal, they frequently suffer from slow tail convergence (see He and Yuan, 2011 and the references therein).

Thus, a need arises to develop a general approach that can solve large scale structured regularization problem efficiently. For such an approach to be successful in practice, it should guarantee to converge at a fast rate, be able to handle massive data sets, and should not rely on approximating the penalty function. In this paper, we propose a framework based on the alternating linearization algorithm of Kiwiel et al. (1999), that satisfies all these requirements.

We consider the following generalization of (1):

$$\mathcal{L}(\beta) = f(\beta) + \lambda \|R\beta\|_{\lozenge}, \quad \lambda > 0, \tag{2}$$

where $\|\cdot\|_{\lozenge}$ is a norm in $\mathbb{R}^m$. Our considerations and techniques will apply to several possible choices of this norm, in particular, to the ℓ_1 norm $\|\cdot\|_1$, and to the total variation norm $\|\cdot\|_{\text{TV}}$ used in image processing.

Formally, we write the objective function as a sum of two convex functions,

$$\mathcal{L}(\beta) = f(\beta) + h(\beta), \tag{3}$$

where $f(\beta)$ is a loss function, which is assumed to be convex with respect to β, and $h(\cdot)$ is a convex penalty function. Any of the functions (or both) may be nonsmooth, but an essential requirement of our framework is that each of them can be easily minimized with respect to β, when augmented by a separable linear-quadratic term $\sum_{i=1}^{p} \left(s_i\beta_i + d_i\beta_i^2 \right)$, with some vectors $s, d \in \mathbb{R}^p$, $d > 0$. Our method bears resemblance to operator splitting and alternating direction approaches, but differs from them in the fact that it is *monotonic* with respect to the values of (3). We discuss these relations and differences later in Section 2.2, but roughly speaking, a special test applied at every iteration of the method decides which of the operator splitting iterations is the most beneficial one.

In our applications, we focus on the quadratic loss function $f(\cdot)$ and the penalty function in the form of generalized lasso (2), as the most important case, where comparison with other approaches is available. This case satisfies the requirement specified above, and allows for substantial specialization and acceleration of the general framework of alternating linearization. In fact, it will be clear from our presentation that any convex loss function $f(\cdot)$ can be handled in exactly the same way.

An important feature of our approach is that problems with the identity design matrix are solved exactly in one iteration, even for very large dimension.

The remainder of the paper is organized as follows. In Section 2, we introduce the alternating linearization method and we discuss its relations to other approaches. Section 3 briefly discusses the application to lasso problems. In Section 4 we describe the application to general structured regularization problems. Section 5 presents simulation results and real data examples, which illustrate the efficacy, accuracy, and scalability of the alternating linearization method. Concluding remarks are presented in Section 6. The appendix contains details about the algorithms used to solve the subproblems of the alternating linearization method.

2. The Alternating Linearization Method

In this section, we describe the alternating linearization (ALIN) approach to minimize (3).

2.1 Outline of the Method

The ALIN is an iterative method, which generates a sequence of approximations $\{\hat{\beta}^k\}$ converging to a solution of the original problem (3), and two auxiliary sequences: $\{\tilde{\beta}_h^k\}$ and $\{\tilde{\beta}_f^k\}$, where k is the iteration number. Each iteration of the ALIN algorithm consists of solving two subproblems: the *h-subproblem* and the *f-subproblem*, and of an *update step*, applied after any of the subproblems, or after each of them.

At the beginning we set $\tilde{\beta}_f^0 = \hat{\beta}^0$, where $\hat{\beta}^0$ is the starting point of the method. In the description below, we suppress the superscript k denoting the iteration number, to simplify notation.

The h-subproblem

We linearize $f(\cdot)$ at $\tilde{\beta}_f$, and approximate it by the function

$$\tilde{f}(\beta) = f(\tilde{\beta}_f) + s_f^T(\beta - \tilde{\beta}_f).$$

If $f(\cdot)$ is differentiable, then $s_f = \nabla f(\tilde{\beta}_f)$; for a general convex $f(\cdot)$, we select a subgradient $s_f \in \partial f(\tilde{\beta}_f)$. In the first iteration, this may be an arbitrary subgradient; at later iterations special selection rules apply, as described in (2.1) below.

The approximation is used in the optimization problem

$$\min_{\beta} \; \tilde{f}(\beta) + h(\beta) + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2, \tag{4}$$

in which the last term is defined as follows:

$$\|\beta - \hat{\beta}\|_D^2 = (\beta - \hat{\beta})^T D(\beta - \hat{\beta}),$$

with a diagonal matrix $D = \text{diag}\{d_j,\ j = 1, \dots, p\}$, $d_j > 0$, $j = 1, \dots, p$. The solution of the h-subproblem (4) is denoted by $\tilde{\beta}_h$.

We complete this stage by calculating the subgradient of $h(\cdot)$ at $\tilde{\beta}_h$, which features in the optimality condition for the minimum in (4):

$$0 \in s_f + \partial h(\tilde{\beta}_h) + D(\tilde{\beta}_h - \hat{\beta}).$$

Elementary calculation yields the right subgradient $s_h \in \partial h(\tilde{\beta}_h)$:

$$s_h = -s_f - D(\tilde{\beta}_h - \hat{\beta}). \tag{5}$$

The f-subproblem

Using the subgradient s_h we construct a linear minorant of the penalty function $h(\cdot)$ as follows:

$$\tilde{h}(\beta) = h(\tilde{\beta}_h) + s_h^T(\beta - \tilde{\beta}_h).$$

This approximation is employed in the optimization problem

$$\min_{\beta} \; f(\beta) + \tilde{h}(\beta) + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2. \tag{6}$$

The optimal solution of this problem is denoted by $\tilde{\beta}_f$. It will be used in the next iteration as the point at which the new linearization of $f(\cdot)$ will be constructed. The next subgradient of $f(\cdot)$ to be used in the h-subproblem will be

$$s_f = -s_h - D(\tilde{\beta}_f - \hat{\beta}).$$

The update step

The update step can be applied after any of the subproblems, or after both of them. It changes the current best approximation of the solution $\hat{\beta}$, if certain improvement conditions are satisfied. We describe it here for the case of applying the update step after the f-subproblem; analogous operations are carried out if the update step is applied after the h-subproblem.

At the beginning of the update step the stopping criterion is verified. If

$$f(\tilde{\beta}_f) + \tilde{h}(\tilde{\beta}_f) \geq f(\hat{\beta}) + h(\hat{\beta}) - \varepsilon, \tag{7}$$

the algorithm terminates. Here $\varepsilon > 0$ is the stopping test parameter.

If the the stopping test is not satisfied, we check the inequality

$$f(\tilde{\beta}_f) + h(\tilde{\beta}_f) \leq (1-\gamma)\big[f(\hat{\beta}) + h(\hat{\beta})\big] + \gamma\big[f(\tilde{\beta}_f) + \tilde{h}(\tilde{\beta}_f)\big]. \tag{8}$$

If it is satisfied, then we update $\hat{\beta} \leftarrow \tilde{\beta}_f$; otherwise $\hat{\beta}$ remains unchanged. Here the parameter $\gamma \in (0,1)$. In the implementation, we use $\gamma = 0.2$. The choice of this parameter does not influence the overall performance of the algorithm.

If the update step is applied after the h-subproblem, we use $\tilde{\beta}_h$ instead of $\tilde{\beta}_f$ in the inequalities (7) and (8).

The update step is a crucial component of the alternating linearization algorithm; it guarantees that the sequence $\{\mathcal{L}(\hat{\beta}^k)\}$ is monotonic (see Lemma 2 in Section 2.3), and it stabilizes the entire algorithm (see the remarks at the end of Section 5.2). It is a specialized form of the main distinction between *null* and *serious* steps in *bundle methods* for nonsmooth optimization. The Reader may consult the books of Bonnans et al. (2003); Hiriart-Urruty and Lemaréchal (1993); Kiwiel (1985); Ruszczyński (2006), and the references therein, for the theory of bundle methods and the significance of null and serious steps in these methods.

In Lemma 3 in Subsection 2.3 we show that under simple conditions the method is convergence linearly between successive serious steps.

2.2 Relation to Operator Splitting and Alternating Direction Methods

Our approach is intimately related to *operator splitting methods* and their dual versions, *alternating direction methods*, which are recently very popular in the area of signal processing (see, e.g., Boyd et al., 2010; Combettes and Pesquet, 2011; Fadili and Peyré, 2011). To discuss these relations, it is convenient to present our method formally, and to introduce two running *proximal centers*:

$$z_f = \hat{\beta} - D^{-1}s_f,$$
$$z_h = \hat{\beta} - D^{-1}s_h.$$

After elementary manipulations we can absorb the linear terms into the quadratic terms and summarize the alternating linearization method as follows.

Algorithm 1 Alternating Linearization

```
1: repeat
2:     $\tilde{\beta}_h \leftarrow \arg\min\left\{h(\beta) + \frac{1}{2}\|\beta - z_f\|_D^2\right\}$
3:     if (Update Test for $\tilde{\beta}_h$) then
4:         $\hat{\beta} \leftarrow \tilde{\beta}_h$
5:     end if
6:     $z_h \leftarrow \hat{\beta} + \tilde{\beta}_h - z_f$
7:     $\tilde{\beta}_f \leftarrow \arg\min\left\{f(\beta) + \frac{1}{2}\|\beta - z_h\|_D^2\right\}$
8:     if (Update Test for $\tilde{\beta}_f$) then
9:         $\hat{\beta} \leftarrow \tilde{\beta}_f$
10:    end if
11:    $z_f \leftarrow \hat{\beta} + \tilde{\beta}_f - z_h$
12: until (Stopping Test)
```

The *Update Test* in lines 3 and 8 is the corresponding version of inequality (8). The *Stopping Test* is inequality (7).

If we assume that the update steps in lines 4 and 9 are carried out after *every* h-subproblem and *every* f-subproblem, without verifying the update test (8), then the method becomes equivalent to a scaled version of the Peaceman–Rachford algorithm (originally proposed by Peaceman and Rachford (1955) for PDEs and later generalized and analyzed by Lions and Mercier (1979); see also Combettes (2009) and the references therein). If $D = \rho I$ with $\rho > 0$, then we obtain an unscaled version of this algorithm.

If we assume that the update steps are carried out after *every* h-subproblem without verifying inequality (8), but *never* after f-subproblems, then the method becomes equivalent to a scaled version of the Douglas–Rachford algorithm (introduced by Douglas and Rachford (1956), and generalized and analyzed by Lions and Mercier (1979); see also Bauschke and Combettes (2011) and the references therein). As the roles of f and h can be switched, the method in which updates are carried always after f-subproblems, but never after h-subproblems, is also equivalent to a scaled Douglas–Rachford method.

Operator splitting methods are not monotonic with respect to the values of the objective function $\mathcal{L}(\beta)$. Their convergence is based on monotonicity with respect to the distance to the optimal solution of the problem (Lions and Mercier, 1979; Eckstein and Bertsekas, 1992).

In contrast, the convergence mechanism of our method is different; it draws from some ideas of bundle methods in nonsmooth optimization (Hiriart-Urruty and Lemaréchal, 1993; Kiwiel, 1985; Ruszczyński, 2006). Its key element is the update test employed in (8). At every iteration we decide whether it is beneficial to make a Peaceman–Rachford step, any of the two possible Douglas–Rachford steps, or none. In the latter case, which we call the *null step*, $\hat{\beta}$ remains unchanged, but the trial points $\tilde{\beta}_h$ and $\tilde{\beta}_f$ are updated. These updates continue, until $\tilde{\beta}_h$ or $\tilde{\beta}_f$ become better than $\hat{\beta}$, or until optimality is detected (*cf.* the remarks at the end of section 5.2). In may be worth noticing that the recent application of the idea of alternating linearization by Goldfarb et al. (2013) removes the update test from the method of Kiwiel et al. (1999), thus effectively reducing it to an operator splitting method.

Alternating direction methods are dual versions of the operator splitting methods, applied to the following equivalent form of the problem of minimizing (3):

$$\min f(\beta_1) + h(\beta_2), \quad \text{subject to} \quad \beta_1 = \beta_2.$$

In regularized signal processing problems, when $f(\beta) = \varphi(X\beta)$ with some fixed matrix X, the convenient problem formulation is

$$\min \varphi(v) + h(\beta), \quad \text{subject to} \quad v = X\beta.$$

The dual functional,

$$L_D(\lambda) = \min_v \left\{ \varphi(v) - \lambda^T v \right\} + \min_\beta \left\{ h(\beta) + \lambda^T X\beta \right\},$$

has the form of a sum of two functions, and the operator splitting methods apply. The reader may consult the papers of Boyd et al. (2010) and Combettes and Pesquet (2011) for appropriate derivations. It is also worth mentioning that the alternating direction methods are sometimes called *split Bregman methods* in the signal processing literature (see, e.g., Goldstein and Osher, 2009; Ye and Xie, 2011, and the references therein). Recently, Qin and Goldfarb (2012) applied alternating direction methods to some structured regularization problems resulting from group lasso models.

However, to apply our alternating linearization method to the dual problem of maximizing $L_D(\lambda)$, we would have to be able to quickly compute the value of the dual functions, in order to verify the update condition (8), as discussed in detail in Kiwiel et al. (1999). The second dual function, $\min_\beta \left\{ h(\beta) + \lambda^T X\beta \right\}$ is rather difficult to evaluate, and it makes the update test time consuming. Without this test, our method reduces to the alternating direction method, which does not have descent properties, and whose tail convergence may be slow. Our experiments reported at the end of Section 5.2 confirm these observations.

2.3 Convergence

Convergence properties of the alternating linearization method follow from the general theory developed by Kiwiel et al. (1999). Indeed, after the change of variables $\xi = D^{1/2}\beta$ we see that the method is identical to Algorithm 3.1 of Kiwiel et al. (1999), with $\rho_k = 1$. The following statement is a direct consequence of (Kiwiel et al., 1999, Theorem 4.8).

Theorem 1 *Suppose that the set of minima of the function* (3) *is nonempty. Then the sequence* $\{\hat{\beta}^k\}$ *generated by the algorithm is convergent to a minimum point* β^* *of the function* (3). *Moreover, every accumulation point* (s_f^*, s_h^*) *of the sequence* $\{(s_f^k, s_h^k)\}$ *satisfies the relations:* $s_f^* \in \partial f(\beta^*)$, $s_h^* \in \partial h(\beta^*)$, *and* $s_f^* + s_h^* = 0$.

For structured regularization problems the assumption of the theorem is satisfied, because both the loss function $f(\cdot)$ and the regularizing function $h(\cdot)$ are bounded from below, and one of the purposes of the regularization term is to make the set of minima of the function $\mathcal{L}(\cdot)$ nonempty and bounded.

It may be of interest to look closer at the stopping test (7) employed in the update step.

Lemma 2 *Suppose β^* is the unique minimum point of $\mathcal{L}(\cdot) = f(\cdot) + h(\cdot)$ and let $\alpha > 0$ be such that $\mathcal{L}(\beta) - \mathcal{L}(\beta^*) \geq \alpha\|\beta - \beta^*\|_D^2$ for all β. Then the stopping criterion (7) implies that*

$$\mathcal{L}(\hat{\beta}) - \mathcal{L}(\beta^*) \leq \frac{\varepsilon}{\alpha}. \tag{9}$$

Proof As $\tilde{h}(\cdot) \leq h(\cdot)$, inequality (7) implies that

$$\begin{aligned}\min_\beta \Big\{ f(\beta) + h(\beta) + \frac{1}{2}\|\beta - \hat{\beta}\|_D^2 \Big\} &\geq \min_\beta \Big\{ f(\beta) + \tilde{h}(\beta) + \frac{1}{2}\|\beta - \hat{\beta}\|_D^2 \Big\} \\ &= f(\tilde{\beta}_f) + \tilde{h}(\tilde{\beta}_f) + \frac{1}{2}\|\tilde{\beta}_f - \hat{\beta}\|_D^2 \\ &\geq f(\hat{\beta}) + h(\hat{\beta}) - \varepsilon. \end{aligned} \tag{10}$$

The expression on the left hand side of this inequality is the Moreau–Yosida regularization of the function $\mathcal{L}(\cdot)$ evaluated at $\hat{\beta}$. By virtue of (Ruszczyński, 2006, Lemma 7.12), after setting $\tilde{x} = \beta^*$ and with the norm $\|\cdot\|_D$, the Moreau–Yosida regularization satisfies the following inequality:

$$\min_\beta \Big\{ \mathcal{L}(\beta) + \frac{1}{2}\|\beta - \hat{\beta}\|_D^2 \Big\} \leq \mathcal{L}(\hat{\beta}) - \frac{\big(\mathcal{L}(\hat{\beta}) - \mathcal{L}(\beta^*)\big)^2}{\|\hat{\beta} - \beta^*\|_D^2}.$$

Combining this inequality with (10) and simplifying, we conclude that

$$\frac{\big(\mathcal{L}(\hat{\beta}) - \mathcal{L}(\beta^*)\big)^2}{\|\hat{\beta} - \beta^*\|_D^2} \leq \varepsilon.$$

Substitution of the denominator by the upper estimate $\big(\mathcal{L}(\beta) - \mathcal{L}(\beta^*)\big)/\alpha$ yields (9). ■

If β^* is unique then $\mathcal{L}(\cdot)$ grows at least quadratically in the neighborhood of β^*. This implies that $\alpha > 0$ satisfying the assumptions of Lemma 2 exists. Our use of the norm $\|\cdot\|_D$ amounts to comparing the function $(\beta - \beta^*)^T X^T X(\beta - \beta^*)$ to its diagonal approximation $(\beta - \beta^*)^T D(\beta - \beta^*)$ for $D = \text{diag}(X^T X)$. The reader may also consult (Ruszczyński, 1995, Lemma 1) for the accuracy of the diagonal approximation when the matrix X is sparse.

By employing the estimate of Lemma 2, we can now prove linear rate of convergence of the method between serious steps.

Lemma 3 *Suppose β^* is the unique minimum point of $\mathcal{L}(\cdot) = f(\cdot) + h(\cdot)$ and let $\alpha > 0$ be such that $\mathcal{L}(\beta) - \mathcal{L}(\beta^*) \geq \alpha\|\beta - \beta^*\|_D^2$ for all β. Then at every serious step, when the update test (8) is satisfied, we have the inequality*

$$\mathcal{L}(\tilde{\beta}_f) - \mathcal{L}(\beta^*) \leq (1 - \gamma\alpha)\big[\mathcal{L}(\hat{\beta}) - \mathcal{L}(\beta^*)\big]. \tag{11}$$

Proof If follows from inequality (8) that

$$\mathcal{L}(\tilde{\beta}_f) \leq (1-\gamma)\mathcal{L}(\hat{\beta}) + \gamma\big[f(\tilde{\beta}_f) + \tilde{h}(\tilde{\beta}_f)\big].$$

Using Lemma 2, we obtain

$$\mathcal{L}(\hat{\beta}) - \mathcal{L}(\beta^*) \leq \frac{1}{\alpha}\big[\mathcal{L}(\hat{\beta}) - f(\tilde{\beta}_f) - \tilde{h}(\tilde{\beta}_f)\big].$$

Combining these inequalities and simplifying, we conclude that

$$\begin{aligned}\mathcal{L}(\tilde{\beta}_f) &\leq (1-\gamma)\mathcal{L}(\hat{\beta}) + \gamma\big\{\alpha\mathcal{L}(\beta^*) - \alpha\mathcal{L}(\hat{\beta}) + \mathcal{L}(\hat{\beta})\big\}\\ &= \mathcal{L}(\hat{\beta}) - \gamma\alpha\big[\mathcal{L}(\hat{\beta}) - \mathcal{L}(\beta^*)\big].\end{aligned}$$

Subtracting $\mathcal{L}(\beta^*)$ from both sides, we obtain the linear rate (11). ■

If the original problem is to minimize (3) subject to the constraint that $\beta \in B$ for some convex closed set B, we can formally add the indicator function of this set,

$$\delta(\beta) = \begin{cases} 0 & \text{if } \beta \in B, \\ +\infty & \text{if } \beta \notin B, \end{cases}$$

to $f(\cdot)$ or to $h(\cdot)$ (whichever is more convenient). This will result in including the constraint in one of the subproblems, and changing the subgradients accordingly. The theory of Kiwiel et al. (1999) covers this case as well, and Theorem 1 remains valid.

3. Application to Lasso Regression

First, we demonstrate the alternating linearization algorithm (ALIN) on the classical lasso regression problem. Due to the separable nature of the penalty function, very efficient coordinate descent methods are applicable to this problem as well (Tseng, 2001), but we wish to illustrate our approach on the simplest case first.

In the lasso regression problem we have

$$f(\beta) = \tfrac{1}{2}\|y - X\beta\|_2^2, \qquad h(\beta) = \lambda\|\beta\|_1,$$

where X is the $n \times p$ design matrix, $y \in \mathbb{R}^n$ is the vector of response variables, $\beta \in \mathbb{R}^p$ is the vector of regression coefficients, and $\lambda > 0$ is a parameter of the model.

We found it essential to use $D = \operatorname{diag}(X^T X)$, that is, $d_j = X_j^T X_j$, $j = 1, \dots, p$. This choice is related to the *diagonal quadratic approximation* of the function $f(\beta) = \frac{1}{2}\|y - X\beta\|_2^2$, which was employed (for similar objectives in the context of augmented Lagrangian minimization) by Ruszczyński (1995). Indeed, in the h-subproblem in the formula (12) below, the quadratic regularization term is a quadratic form built on the diagonal of the Hessian of $f(\cdot)$.

The h-subproblem

The problem (4), after skipping constants, simplifies to the following form

$$\min_{\beta} \; s_f^T\beta + \lambda\|\beta\|_1 + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2, \tag{12}$$

with $s_f = X^T(X\tilde{\beta}_f - y)$. Writing $\tau_j = \hat{\beta} - \tilde{s}_{fj}/d_j$, we obtain the following closed form solutions of (12), which can be calculated component-wise:

$$\tilde{\beta}_{hj} = \operatorname{sgn}(\tau_j)\max\Big(0, |\tau_j| - \frac{\lambda}{d_j}\Big), \quad j = 1,\dots,p.$$

The subgradient s_h of $h(\cdot)$ at $\tilde{\beta}_h$ is calculated by (5).

The f-subproblem

The problem (6), after skipping constants, simplifies to the unconstrained quadratic programming problem

$$\min_{\beta} \; s_h^T\beta + \tfrac{1}{2}\|y - X\beta\|_2^2 + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2.$$

Its solution can be obtained by solving the following symmetric linear system in $\delta = \beta - \hat{\beta}$:

$$(X^TX + D)\delta = X^T(y - X\hat{\beta}) - s_h. \tag{13}$$

This system can be efficiently solved by the preconditioned conjugate gradient method (see, e.g., Golub and Van Loan, 1996), with the diagonal preconditioner $D = \operatorname{diag}(X^TX)$. Its application does not require the explicit form of the matrix X^TX; only matrix-vector multiplications with X and X^T are employed, and they can be implemented with sparse data structures.

The numerical accuracy of this approach is due to the improved condition index of the resulting matrix, as explained in the following lemma.

Lemma 4 *The application of the preconditioned conjugate gradient method with preconditioner D to the system* (13) *is equivalent to the application of the conjugate gradient method to a system with a symmetric positive definite matrix $\bar{H}$ whose condition index is at most $p+1$.*

Proof By construction, the preconditioned conjugate gradient method with a preconditioner D applied to a system with a matrix H is the standard conjugate gradient method applied to a system with the matrix $\bar{H} = D^{-\frac{1}{2}}HD^{-\frac{1}{2}}$. Substituting the matrix from (13), we obtain:

$$\bar{H} = D^{-\frac{1}{2}}(X^TX + D)D^{-\frac{1}{2}} = D^{-\frac{1}{2}}X^TXD^{-\frac{1}{2}} + I.$$

Define $\bar{X} = XD^{-\frac{1}{2}}$. By the construction of D, all columns $\bar{x}^j$, $j = 1,\dots,p$, of $\bar{X}$ have Euclidean length 1.

The condition index of $\bar{H}$ is equal to

$$\operatorname{cond}(\bar{H}) = \frac{\lambda_{\max}(\bar{X}^T\bar{X}) + 1}{\lambda_{\min}(\bar{X}^T\bar{X}) + 1}.$$

As the matrix $\bar{X}^T\bar{X}$ is positive semidefinite, $\lambda_{\min}(\bar{X}^T\bar{X}) \geq 0$. To estimate $\lambda_{\max}(\bar{X}^T\bar{X})$ suppose v is the corresponding eigenvector of Euclidean length 1. We obtain the chain of relations:

$$\sqrt{\lambda_{\max}(\bar{X}^T\bar{X})} = \|\bar{X}v\|_2 = \Big\| \sum_{j=1}^{p} v_j \bar{x}^j \Big\|_2 \leq \sum_{j=1}^{p} |v_j| \|\bar{x}^j\|_2 = \|v\|_1 \leq \sqrt{p}\|v\|_2 = \sqrt{p}.$$

Therefore, the condition index of $\bar{H}$ is at most $p+1$. ■

While this universal estimate can be still very large, our experience is that the preconditioned conjugate gradient method solves the system (13) in a small number of iterations even for large p (see the discussion following Figure 4).

4. Application to General Structured Regularization Problems

In the following we apply the alternating linearization algorithm to solve more general structured regularization problems including the generalized Lasso (2). Here we assume the least square loss, as in the previous subsection. The objective function can be written as follows:

$$\mathcal{L}(\beta) = f(\beta) + h(\beta) = \tfrac{1}{2}\|y - X\beta\|_2^2 + \lambda\|R\beta\|_\Diamond. \tag{14}$$

For example, for the one-dimensional fused lasso, R is the following $(p-1)\times p$ matrix:

$$R = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -1 & 1 \end{bmatrix},$$

and the norm $\|\cdot\|_\Diamond$ is the ℓ_1-norm $\|\cdot\|_1$, but our derivations are valid for any form of R, and any norm $\|\cdot\|_\Diamond$.

The h-subproblem

The h-subproblem can be equivalently formulated as follows:

$$\min_{\beta,z} \; s_f^T\beta + \lambda\|z\|_\Diamond + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2 \quad \text{subject to} \quad R\beta = z. \tag{15}$$

Owing to the use of $D = \operatorname{diag}(X^TX)$, and with $s_f = X^T(X\hat{\beta} - y)$, it is a quite accurate approximation of the original problem, especially for sparse X (Ruszczyński, 1995).

The Lagrangian of problem (15) has the form

$$L(\beta, z, \mu) = s_f^T\beta + \lambda\|z\|_\Diamond + \mu^T(R\beta - z) + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2,$$

where μ is the dual variable. Consider the dual norm $\|\cdot\|_*$, defined as follows:

$$\|\mu\|_* = \max_{\|z\|_\Diamond \leq 1} \mu^T z, \qquad \|z\|_\Diamond = \max_{\|\mu\|_* \leq 1} \mu^T z.$$

We see that the minimum of the Lagrangian with respect to z is finite if and only if $\|\mu\|_* \leq \lambda$ (Ruszczyński, 2006, Example 2.94). Under this condition, the minimum value of the z-terms is zero and we can eliminate them from the Lagrangian. We arrive to its reduced form,

$$\hat{L}(\beta, \mu) = s_f^T \beta + \mu^T R\beta + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2. \tag{16}$$

To calculate the dual function, we minimize $\hat{L}(\beta, \mu)$ over $\beta \in \mathbb{R}^p$. After elementary calculations, we obtain the solution

$$\tilde{\beta}_h = \hat{\beta} - D^{-1}(s_f + R^T \mu). \tag{17}$$

Substituting it back to (16), we arrive to the following dual problem:

$$\max_{\mu} \; -\tfrac{1}{2}\mu^T R D^{-1} R^T \mu + \mu^T R(\hat{\beta} - D^{-1}s_f) \quad \text{subject to} \quad \|\mu\|_* \leq \lambda. \tag{18}$$

This is a norm-constrained optimization problem. Its objective function is quadratic, and the specific form of the constraints depends on the norm $\|\cdot\|_\Diamond$ used in the regularizing term of (2).

The case of the ℓ_1-norm

If the norm $\|\cdot\|_\Diamond$ is the ℓ_1-norm $\|\cdot\|_1$, then the dual norm is the ℓ_∞-norm:

$$\|\mu\|_* = \|\mu\|_\infty = \max_{1\leq j\leq m} |\mu_j|.$$

In this case (18) becomes a box-constrained quadratic programming problem, for which many efficient algorithms are available. One possibility is the active-set box-constrained preconditioned conjugate gradient algorithm with spectral projected gradients, as described by Birgin and Martínez (2002); Friedlander and Martínez (1994). It should be stressed that its application does not require the explicit form of the matrix $RD^{-1}R^T$; only matrix-vector multiplications with R and R^T are employed, and they can be implemented with sparse data structures.

An even better possibility, due to the separable form of the constraints, is coordinate-wise optimization (see, e.g., Ruszczyński, 2006, Sec. 5.8.2) in the dual problem (18). In our experiments, the dual coordinate-wise optimization method strictly outperforms the box-constrained algorithm, in terms of the solution time.

The solution $\tilde{\mu}$ of the dual problem can be substituted into (17) to obtain the primal solution.

The case of a sum of ℓ_2-norms

Another important case arises when the vector $z = R\beta$ is split into I subvectors $z^1, z^2, \ldots, z^I$, and

$$\|z\|_\Diamond = \sum_{i=1}^{I} \|z^i\|_2. \tag{19}$$

This is the group lasso model, also referred to as the ℓ_1/L_2-norm regularization (see, e.g., Qin and Goldfarb, 2012). A special case of it is the *total variation norm* is discussed in Section 5.4.

We can directly verify that the dual norm has the following form:

$$\|\mu\|_* = \max_{1\le i\le I} \|\mu^i\|_2.$$

It follows that problem (18) is a block-quadratically constrained quadratic optimization problem:

$$\begin{aligned}&\max_{\mu}\ -\tfrac{1}{2}\mu^T R D^{-1} R^T \mu + \mu^T R(\hat{\beta} - D^{-1} s_f)\\ &\text{s. t. } \|\mu^i\|_2^2 \le \lambda^2, \quad i = 1, \ldots, I.\end{aligned}$$

This problem can be very efficiently solved by a cyclical block-wise optimization with respect to the subvectors $\mu^1, \mu^2, \ldots, \mu^I$. At each iteration of the method, optimization with respect to the corresponding subvector μ^j is performed, subject to one constraint $\|\mu^j\|_2^2 \le \lambda^2$. The other subvectors, μ^i, $i \ne j$ are kept fixed on their last values. After that, j is incremented (if $j < I$) or reset to 1 (if $j = I$), and the iteration continues. The method stops when no significant improvements over I steps can be observed. The dual block optimization method performs well in the applications we are interested in. General convergence theory can be found in Tseng (2001).

Again, the solution $\tilde{\mu}$ of the dual problem is substituted into (17) to obtain the primal solution.

The f-subproblem

We obtain the update $\tilde{\beta}_f$ by solving the linear equation system (13), exactly as in the lasso case.

The special case of $X = I$

If the design matrix $X = I$ in (14), then our method solves the problem in one iteration, when started from $\hat{\beta} = y$. Indeed, in this case we have $s_f = 0$, $D = I$, and the first h-subproblem becomes equivalent to the original problem (14):

$$\min_{\beta,z}\ \lambda\|z\|_\Diamond + \tfrac{1}{2}\|\beta - y\|_2^2 \quad \text{subject to} \quad R\beta = z.$$

The dual problem (18) simplifies as follows:

$$\max_{\mu}\ -\tfrac{1}{2}\mu^T R R^T \mu + \mu^T R y \quad \text{subject to} \quad \|\mu\|_* \le \lambda.$$

It can be solved by the same block-wise optimization method, as in the general case. The optimal primal solution is then $\tilde{\beta}_h = y - R^T\mu$.

5. Numerical Experiments

In this section, we present results of a number of studies on simulated and real data, involving a variety of non-differentiable penalty functions. We compare the alternating linearization algorithm (ALIN) with competing approaches in terms of iteration counts, computation time, and estimation accuracy. All these studies were performed on an AMD 2.6 GHZ, 4 GB RAM computer using MATLAB.

5.1 ℓ_1 Regularization

In this section, we compare ALIN with some competing methods for solving the ℓ_1 regularization problem:

$$\min_{\beta} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1, \quad \lambda > 0.$$

The methods that we are comparing with are: SpaRSA, a type of iterative thresholding method (Wright et al., 2009; Xiao and Zhang, 2013); FISTA, a variation of the Nesterov method, considered to be state-of-the-art among the first order methods; and SPG, a spectral gradient method (den Berg and Friedlander, 2008). We follow the procedure described by Wright et al. (2009) to generate a data set for comparisons. The elements of the matrix X are generated independently using a Gaussian distribution with mean zero and variance 10^{-2}. The dimension of X is $n = 2^{10}$ by $p = 2^{12}$, $p = 2^{13}$, and $p = 2^{14}$. The true signal, β, is a vector with 160 randomly placed ± 1 spikes and zeros elsewhere. The dependent variables are $y = X\beta + \epsilon$, where ϵ is Gaussian noise with variance 10^{-4}.

			$p = 2^{12}$		$p = 2^{13}$		$p = 2^{14}$
$\lambda = \tau$	*ALIN*	17.99	(10.68)	36.60	(15.08)	105.14	(40.88)
	FISTA	8.58	(4.00)	18.63	(9.35)	58.73	(42.01)
	SPARSA	8.18	(2.34)	8.18	(3.80)	34.23	(49.55)
	SPG	160.72	(27.48)	160.72	(47.82)	404.59	(79.62)
$\lambda = 10^{-1}\tau$	*ALIN*	9.35	(2.99)	34.01	(15.18)	74.06	(34.27)
	FISTA	16.91	(4.57)	36.43	(16.66)	131.91	(33.94)
	SPARSA	20.55	(10.08)	36.81	(19.29)	169.88	(73.01)
	SPG	136.26	(23.30)	186.94	(46.56)	460.71	(38.93)
$\lambda = 5 \times 10^{-2}\tau$	*ALIN*	6.30	(2.73)	21.83	(10.17)	65.79	(39.49)
	FISTA	18.88	(2.95)	48.37	(15.77)	158.73	(21.86)
	SPARSA	35.56	(11.32)	74.66	(24.49)	234.75	(64.67)
	SPG	140.00	(23.15)	190.43	(45.48)	473.78	(6.41)
$\lambda = 10^{-2}\tau$	*ALIN*	4.58	(2.05)	16.96	(10.99)	28.88	(16.03)
	FISTA	18.85	(3.04)	46.58	(14.91)	169.94	(19.76)
	SPARSA	33.52	(16.10)	76.69	(28.18)	214.85	(124.56)
	SPG	140.63	(22.43)	196.54	(43.96)	483.71	(4.51)
$\lambda = 10^{-3}\tau$	*ALIN*	3.68	(1.20)	6.67	(2.43)	20.16	(4.31)
	FISTA	18.88	(2.84)	45.40	(14.28)	162.85	(36.76)
	SPARSA	19.94	(12.10)	39.55	(27.45)	92.76	(102.53)
	SPG	138.73	(19.56)	201.74	(48.09)	467.91	(101.32)

Table 1: Average run time (in CPU seconds) and standard deviation (in parenthesis) comparison for combinations of dimension p and tuning parameter λ.

To make a fair comparison between the methods, we run FISTA on each instance of the problem. FISTA is set to run to "tol" $= 10^{-5}$ or $5,000$ iterations, whichever comes first. Then ALIN, SpaRSA, and SPG are set to run until the objective function values obtained are as good as that of FISTA. We set a parameter $\tau = 0.1\|X^T y\|_\infty$ and chose values of $\lambda = \tau, 10^{-1}\tau$, $5 \times 10^{-2}\tau$, $10^{-2}\tau$, and $10^{-3}\tau$. We allow SpaRSA to run its *monotone* and *continuation* feature. *Continuation* is a special feature of SpaRSA for cases when the parameter λ is small. With this feature, SpaRSA computes the solutions for bigger values of λ and uses them to find solutions for smaller values of λ. We did not let SpaRSA use its special *de-bias* feature since it involves removing zero coefficients to reduce the size of the data set. This feature makes it unfair for the other competing methods. In Table 1, we report the average time elapsed (in seconds) and the standard deviation after 20 runs.

We can see that the performance of ALIN is comparable to the other methods. In terms of running time, ALIN does better than all competing methods for all but the largest λ considered. For the large value of λ, ALIN performs worse than FISTA and SPARSA. In this case, the solution is fairly close to the starting point 0, therefore the overhead cost of the update steps and the f-subproblem slow ALIN down. When the value of λ decreases, the benefits of these steps become more evident. ALIN outperforms other methods in terms of the running time, by factors of two to three.

			$p = 2^{12}$		$p = 2^{13}$		$p = 2^{14}$
$\lambda = \tau$	*ALIN*	12.35	(0.05)	30.46	(0.05)	66.02	(0.05)
	FISTA	6.99	(0.05)	17.25	(0.05)	44.07	(0.05)
	SPARSA	3.16	(0.05)	6.14	(0.05)	12.26	(0.05)
	SPG	47.43	(0.05)	74.72	(0.05)	161.54	(0.05)
$\lambda = 10^{-1}\tau$	*ALIN*	5.45	(0.02)	19.17	(0.03)	62.13	(0.04)
	FISTA	12.08	(0.02)	25.40	(0.03)	79.48	(0.04)
	SPARSA	14.69	(0.02)	26.85	(0.03)	110.48	(0.04)
	SPG	53.54	(0.03)	93.18	(0.04)	175.77	(0.05)
$\lambda = 5 \times 10^{-2}\tau$	*ALIN*	4.97	(0.02)	18.03	(0.02)	57.55	(0.03)
	FISTA	15.58	(0.02)	32.26	(0.02)	94.91	(0.03)
	SPARSA	20.62	(0.02)	42.67	(0.02)	168.76	(0.03)
	SPG	55.82	(0.03)	94.13	(0.04)	177.17	(0.05)
$\lambda = 10^{-2}\tau$	*ALIN*	5.06	(0.01)	12.67	(0.02)	29.10	(0.04)
	FISTA	18.74	(0.01)	38.59	(0.02)	106.43	(0.04)
	SPARSA	38.76	(0.01)	79.33	(0.02)	191.51	(0.04)
	SPG	54.85	(0.03)	95.58	(0.04)	180.52	(0.05)
$\lambda = 10^{-3}\tau$	*ALIN*	3.78	(0.03)	4.98	(0.04)	9.59	(0.05)
	FISTA	21.61	(0.03)	38.50	(0.04)	106.53	(0.05)
	SPARSA	32.78	(0.03)	45.69	(0.04)	98.75	(0.05)
	SPG	56.90	(0.04)	96.95	(0.05)	183.67	(0.05)

Table 2: Average run time (in CPU seconds) and MSE (in parenthesis) comparison for combinations of dimension p and tuning parameter λ.

We further compare the efficiency of these methods by a validation experiment. All the features of the previous experiment remain the same, except that the noise variance is increased from 10^{-4} to 10^{-1} (which should favor higher regularization). We generate $n = 2^{11}$ samples, where half of them are used as the training set and the other half for testing. For each of the five λ values, we apply the four algorithms to the training set. The fitted models are then validated on the testing set to obtain out-of-sample estimation errors (measured by mean square error). The results are reported in Table 2. All the methods obtain similar MSE on the testing data and the smallest MSE is achieved at $\lambda = 10^{-2}\tau$ and $\lambda = 5 \times 10^{-2}\tau$. In both cases, ALIN clearly performs the best among the four competing methods. As shown in Figure 1, ALIN is also the fastest in terms of the overall computation cost.

It's worth noting that the implementation of FISTA was in the C programming language, while ALIN was implemented as a MATLAB script. Moreover, SpaRSA is a very efficient method specially designed for separable regularization. From our numerical studies, for medium and small values of λ, FISTA makes very small improvement over $5,000$ iterations and SPARSA has to go through many previous values of λ to reach the desired level of objective function values.

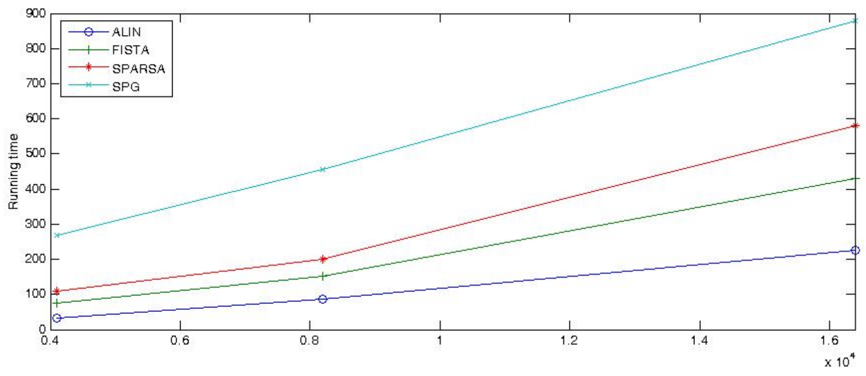

Figure 1: Run time (in CPU seconds) comparisons on the validation experiment for ALIN, FISTA, SPARSA and SPG. The sample size $n = 2^{11}$.

5.2 Fused Lasso Regularization

In this experiment, we compare the ALIN algorithm with four different approaches using data sets generated from a linear regression model $y = \sum_{j=1}^{p} x_j\beta_j + \epsilon$, with pre-specified coefficients β_j, and varying dimension p. The values of x_j are drawn from the multivariate normal distribution with zero mean and pairwise correlation of $\sigma = 0.3$. The noise ϵ is generated from the normal distribution with zero mean and variance equal to 0.01. Among the coefficients β_j, 10% equal 1, 20% equal 2, and the rest are zero. For instance, with $p = 100$, we may have

$$\beta_j = \begin{cases} 1 & \text{for } j = 11, 12, \ldots, 20, \\ 2 & \text{for } j = 21, \ldots, 40, \\ 0 & \text{otherwise.} \end{cases}$$

The regularization problem we attempt to solve is

$$\min_{\beta} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=2}^{p} |\beta_j - \beta_{j-1}|, \quad \lambda > 0.$$

Table 3 reports the run times of ALIN and the four competing algorithms: the generic quadratic programming solver (SQOPT), an implementation of Nesterov's method, SLEP, of Liu et al. (2011); Nesterov (2007), the Split Bregman method of Ye and Xie (2011) (BREGMAN), and alternating direction method of multipliers (ADMM) (Eckstein, 2012). We fix the sample size at n=1000 and vary the dimension p of the problem from 5000 to 50000. To keep the comparison fair, we first run ADMM using the stopping criteria of Boyd et al. (2010) with $tol = 1e-6$ or 30 iterations since each iteration of ADMM can be expensive. Then, the other methods are run until they obtain the same objective function value, or their stopping criteria are satisfied. Each method is repeated on 10 different randomly generated data sets for different values of the tuning parameter λ, and the average running time is reported.

Judging from these results, ALIN clearly outperforms the other methods in terms of speed for most cases. The relative improvements on run time can be as much as 8-fold, depending on the experimental setting, and become more significant, when the data dimension grows. This is particularly significant in view of the fact that ALIN was implemented as a MATLAB code, as opposed to the executables in the other cases. Figure 2 presents the solutions obtained by ALIN, ADMM, SLEP and BREGMAN. The first three methods achieve results that are very close to the original β, and their final objective function values are similar. From our experience, BREGMAN performs well when the dimension p is not significantly larger than the number of observations n. When $p \gg n$, although BREGMAN has a very good running time, it tends to terminate early and does not provide accurate results. In Figure 2, we can see that the solution obtained by BREGMAN is not as good as those of the other three methods.

We further investigated how ALIN approaches the optimal objective function value compared to the other methods. Using the above simulated data set with $n = 1000$, $p = 5000$, and $\lambda = 0.5$, we run ADMM, ALIN, and SLEP until convergence. At each iteration, we calculated the difference between the optimal value $\mathcal{L}^*$ (obtained by SQOPT) and the current function value for each method. Figure 3 displays (in a logarithmic scale) the progress of these methods. It is clear that ALIN achieves the same accuracy as SLEP and ADMM in a much smaller number of iterations. Furthermore, the convergence of ALIN is monotonic, whereas that of SLEP is not. ADMM makes good improvements at the beginning but then it takes thousands of iterations to get to the desired objective function value.

In Figure 4 we show how the ADMM, ALIN and SLEP scale with the dimension of the problem. From this plot, the computation cost of ALIN is consistently lower than those of ADMM and SLEP. The efficiency of the method is due mainly to its good convergence properties, but also to the efficiency of the preconditioned conjugate gradient method for solving the subproblems. It employs sparse data structures and converges rapidly. Usually, between 10 and 20 iterations of the conjugate gradient method are sufficient to find the solution of a subproblem.

		$p = 5000$	p=10,000	p=20,000	p=50,000
$\lambda = 10^{-4}$	*ADMM*	47.45	324.93	590.22	973.48
	SQOPT	1076.00	NA	NA	NA
	SLEP	58.93	569.40	1041.91	2759.85
	ALIN	7.01	35.85	52.74	500.05
	BREGMAN	92.71	129.98	172.36	92.67
$\lambda = 10^{-3}$	*ADMM*	51.94	327.29	597.86	963.79
	SQOPT	1025.00	NA	NA	NA
	SLEP	59.65	570.25	1059.80	2750.72
	ALIN	11.45	67.69	72.04	632.32
	BREGMAN	90.86	129.08	158.72	91.88
$\lambda = 10^{-2}$	*ADMM*	52.37	322.74	599.56	1274.93
	SQOPT	1019.00	NA	NA	NA
	SLEP	59.24	576.82	1053.05	2729.88
	ALIN	27.68	100.29	225.30	810.70
	BREGMAN	86.70	129.29	153.66	90.47
$\lambda = 0.1$	*ADMM*	52.03	326.55	574.75	1528.20
	SQOPT	956.00	NA	NA	NA
	SLEP	56.58	572.34	1004.90	2679.26
	ALIN	31.42	292.73	466.84	597.30
	BREGMAN	93.93	125.65	136.92	77.98
$\lambda = 0.5$	*ADMM*	51.57	317.28	565.74	1482.76
	SQOPT	1015.00	NA	NA	NA
	SLEP	53.10	541.02	983.12	2420.88
	ALIN	22.74	211.35	473.82	908.08
	BREGMAN	97.38	129.68	151.67	71.19
$\lambda = 1$	*ADMM*	51.45	322.37	569.20	1397.77
	SQOPT	1029.00	NA	NA	NA
	SLEP	46.80	470.14	879.51	2404.60
	ALIN	18.15	169.31	477.02	983.31
	BREGMAN	85.14	136.82	170.35	61.48

Table 3: Run time (in CPU seconds) comparison for combinations of dimension p and tuning parameter λ for fused lasso problems

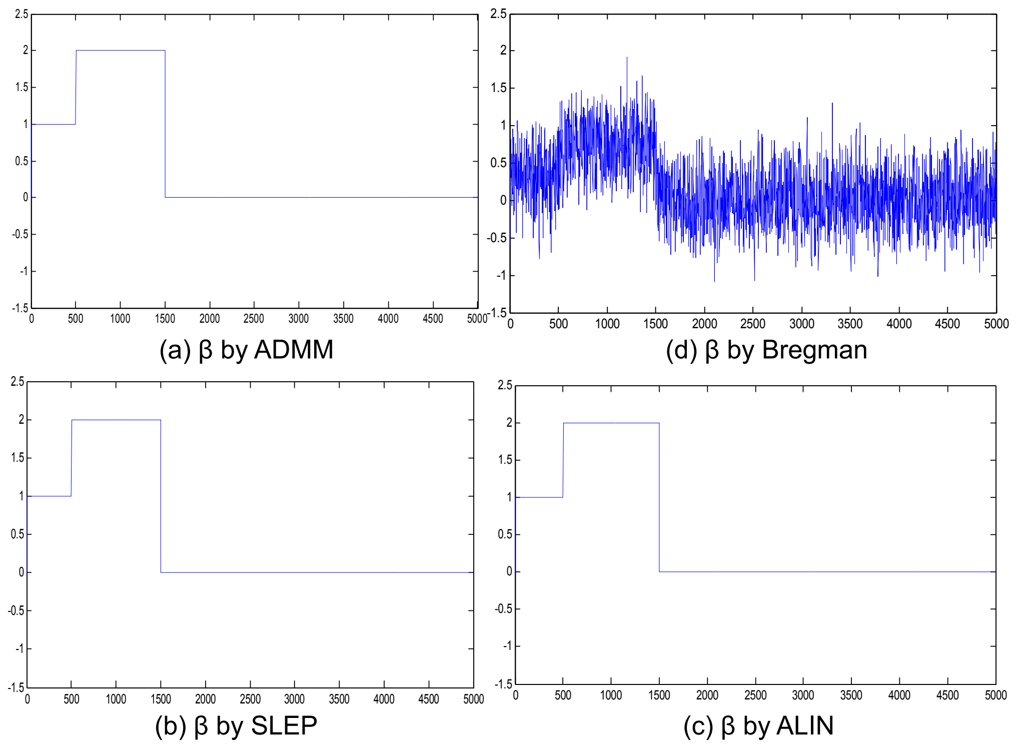

Figure 2: Results of using fused lasso penalty on a simulated data set with $n = 1000$, $p = 5000$, and $\lambda = 0.5$. Plots (a), (b), (c), and (d) correspond to results from ADMM, SLEP, ALIN, and BREGMAN, respectively.

The update test (8) is an essential element of the ALIN method. For example, in the case with $n = 1000$, $p = 5000$, and $\lambda = 0.1$, the update of $\hat{\beta}$ occurred in about 80% of the total of 70 iterations, while other iterations consisted only of improving alternating linearizations. If we allow updates of $\hat{\beta}$ at every step, the algorithm takes more than 5000 iterations to converge in this case. Similar behavior was observed in all other cases. These observations clearly demonstrate the difference between the alternating linearization method and the operator splitting methods.

5.3 CGH Data Example

In this study we present the results on analyzing the CGH data using fused lasso penalty. CGH is a technique for measuring DNA copy numbers of selected genes on the genome. The CGH array experiments return the log ratio between the number of DNA copies of the gene in the tumor cells and the number of DNA copies in the reference cells. A value greater

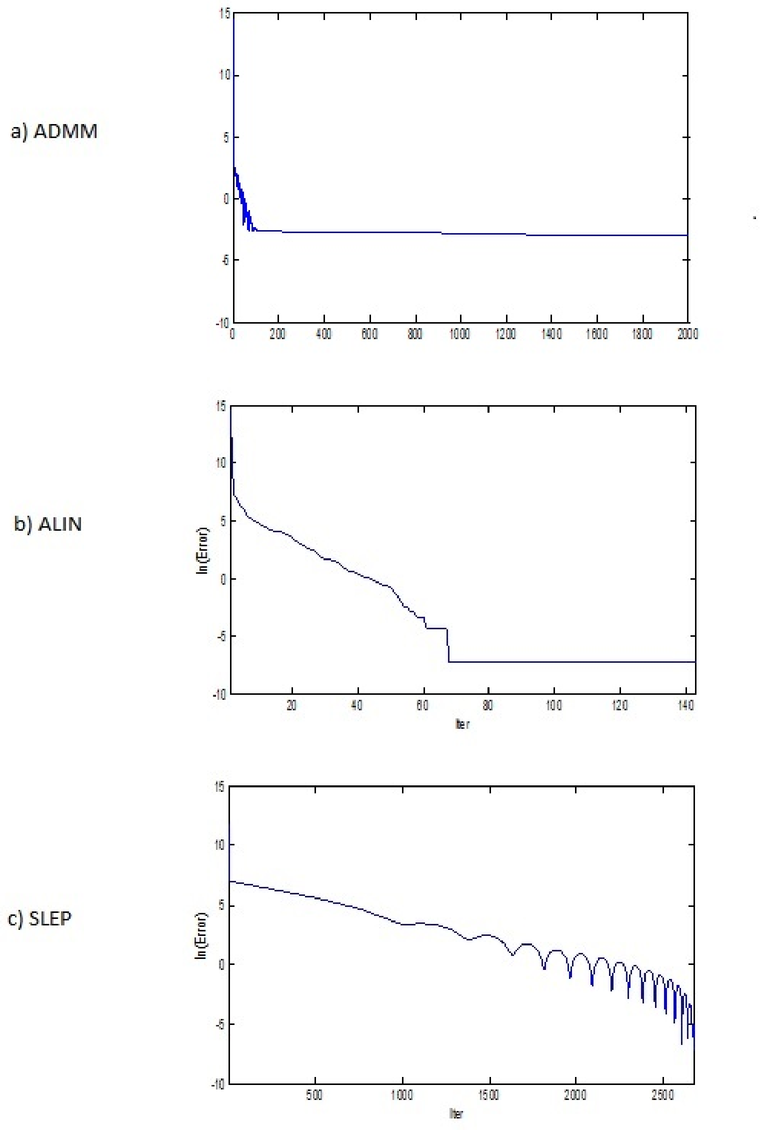

Figure 3: Simulated data set with $n = 1000$, $p = 5000$, $\lambda = 0.1$. Plots (a), (b) and (c): ln(Error) versus number of iterations for ADMM,ALIN, and SLEP respectively. Error is defined as the difference between the optimal value $\mathcal{L}^*$ (obtained by SQOPT) and those obtained by ADMM, ALIN, and SLEP respectively.

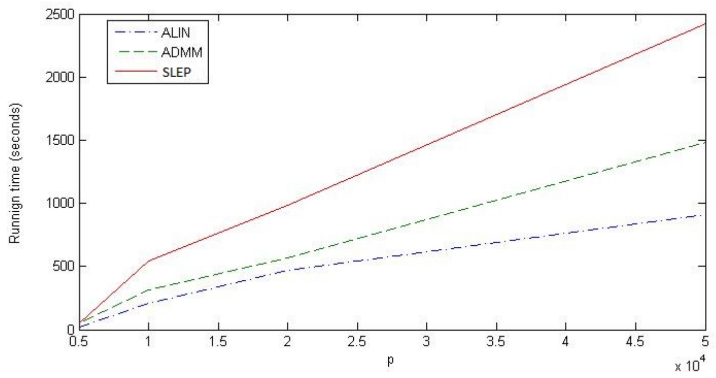

Figure 4: Running time of ADMM, SLEP and ALIN as dimension changes. The vertical axis is the run time in seconds, and the horizontal axis is the data dimension.

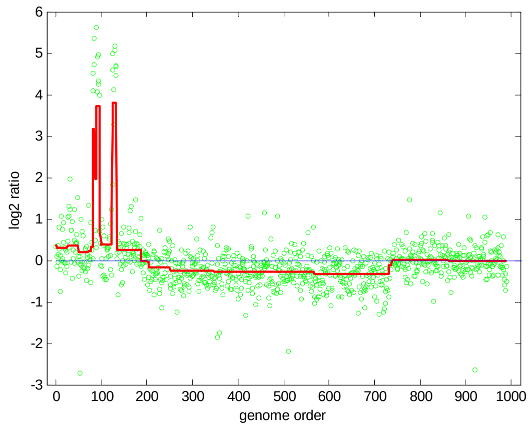

Figure 5: Fused lasso applied to CGH data, $\lambda = 3$.

than zero indicates a possible gain, while a value less than zero suggests possible losses. Tibshirani and Wang (2008) applied the fused lasso signal approximator for detecting such copy number variations. This is a simple one-dimensional signal approximation problem with the design matrix X being the identity matrix. Thus the advantage of ALIN over the other three methods is not significant, due to the overhead that ALIN has during the conjugate gradient method implemented in MATLAB. Indeed the solution time of ALIN is comparable to that of Bregman and SLEP.

Figure 5 presents the estimation results obtained by our ALIN method. The green dots shows the original CNV number, and the red line presents the fused lasso penalized estimates.

In Table 4, the running time comparisons between ADMM, SLEP, ALIN, and Bregman on the CGH data are reported. The fused lasso signal approximator problem has been solved successfully by many methods. Although the main focus of ALIN is on a general design matrix X, ALIN's performance is still comparable with other methods for this example.

	SLEP	ADMM	Alin	Bregman
λ=0.1	0.01	0.06	0.01	0.02
λ=0.5	0.01	0.05	0.01	0.02
λ=1	0.01	0.05	0.03	0.02
λ=3	0.01	0.06	0.06	0.02

Table 4: Average run time (in CPU seconds) comparison for different values of tuning parameter λ on the CGH data example.

5.4 Total Variation Based Image Reconstruction

In image recovery literature, two classes of regularizers are well known. One is the Tikhonov type operator, where the regularizing term is quadratic, and the other is the discrete total variation (TV) regularizer. The resulting objective function from the first type is relatively easy to minimize, but it tends to over-smooth the image, thus failing to preserve its sharpness (Wang et al., 2008). In the following experiment, we demonstrate the effectiveness of ALIN in solving TV-based image deblurring problems, with discrete TV (Rudin et al., 1992), as well as a comparison to the Tikhonov regularizer.

Although of similar form, higher-order fused lasso models are fundamentally different from the one-dimensional fused lasso, as the structural matrix R appearing in (2) is not full-rank and R^TR is ill-conditioned. This additional complication introduces considerable challenges in the path type algorithms (Tibshirani and Taylor, 2011), and additional computational steps need to be implemented to guarantee convergence. The ALIN algorithm does not suffer from complications due to the singularity of R, because the dual problem (18) is always well-defined and has a solution. Even if the solution is not unique, (17) is still an optimal solution of the h-subproblem, and the algorithm proceeds unaffected.

Let y be an $m\times n$ observed noisy image; one attempts to minimize the following objective function:

$$\mathcal{L}(\beta) = \tfrac{1}{2}\|y - \mathcal{A}(\beta)\|_2^2 + \lambda h(\beta), \tag{20}$$

where $h(\beta)$ is an image variation penalty, and $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ is a linear transformation. When $\mathcal{A}$ is the identity transformation, the problem is to *denoise* the image y, but we are rather interested in a significantly more challenging problem of *deblurring*, where $\mathcal{A}$ replaces each pixel with the average of its neighbors and itself (typically, a 3 by 3 block, except for the border).

The penalty can be defined as the ℓ_1-norm of the differences between neighboring pixels (ℓ_1-TV),

$$h(\beta) = \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\big(|\beta_{i,j}-\beta_{i+1,j}|+|\beta_{i,j}-\beta_{i,j+1}|\big)+\sum_{i=1}^{m-1}|\beta_{i,n}-\beta_{i+1,n}|+\sum_{j=1}^{n-1}|\beta_{m,j}-\beta_{m,j+1}|, \tag{21}$$

or as follows (ℓ_2-TV):

$$h(\beta) = \sum_{i=1}^{m-1}\sum_{j=1}^{n-1} \left(|\beta_{i,j} - \beta_{i+1,j}|^2 + |\beta_{i,j} - \beta_{i,j+1}|^2\right)^{1/2} + \sum_{i=1}^{m-1} |\beta_{i,n} - \beta_{i+1,n}| + \sum_{j=1}^{n-1} |\beta_{m,j} - \beta_{m,j+1}|. \tag{22}$$

It is clear that both cases can be cast into the general form (2), with the operator R representing the evaluation of the differences $\beta_{i,j} - \beta_{i+1,j}$ and $\beta_{i,j} - \beta_{i,j+1}$. The regularizing function (21) corresponds to the ℓ_1-norm of $R\beta$, while the function (22) corresponds to a norm of form (19). In the latter case, we have mn blocks, each of dimension two, except for the border blocks, which are one-dimensional.

In the following experiments, we apply the ℓ_1-TV to recover noisy and blurred images to their original forms. The resulting regularization problems are rather complex. Deblurring a 256 by 256 image results in solving a very large generalized lasso problem (the matrix R has dimensions of about 262000×66000). The f-subproblem is solved using the block coordinate descent method and the h-subproblem is solved using the preconditioned conjugate gradient method with "tol" $= 10^{-5}$, as discussed previously. The fact that $\mathcal{A}$ and R are sparse matrices makes the implementation very efficient, as demonstrated in the numerical study.

First, we *blur* the image, by replacing each pixel with the average of its neighbors and itself. This operation defines the kernel operator $\mathcal{A}$ used in the loss function $\frac{1}{2}\|y - \mathcal{A}(\beta)\|_2^2$. Then we add $N(0, 0.02)$ noise to each pixel. Clearly, for image deblurring, the design matrix is no longer the identity matrix, thus the problem is more complicated than the image denoising problem. The deblurring results on a standard example ("Lena") are shown in Figure 6; similar deblurring results from ALIN and FISTA are observed.

Next, we run the image deblurring on a 1 Megapixel image. We compare the result of image deblurring using the ℓ_1-TV and a quadratic Tiknonov regularization approach, which corresponds to formula (22) without the square root operations:

$$h(\beta) = \sum_{i=1}^{m-1}\sum_{j=1}^{n-1} |\beta_{i,j} - \beta_{i+1,j}|^2 + |\beta_{i,j} - \beta_{i,j+1}|^2 + \sum_{i=1}^{m-1} |\beta_{i,n} - \beta_{i+1,n}|^2 + \sum_{j=1}^{n-1} |\beta_{m,j} - \beta_{m,j+1}|^2. \tag{23}$$

The results are shown in Figure 7. It is seen that the ℓ_1-TV recovers a sharper image than the quadratic penalty. Deblurring with the two-dimensional fused lasso penalty yields an MSE of 7.2 with respect to the original image, while that of Tikhonov regularization is 9.3. Deblurring with the regularizer (21) has an almost identical effect as with (22).

There have been many efficient iterative methods proposed to solve this problem. Two outstanding general frameworks are a variation of Nesterov's gradient method (Nesterov, 2007) and the method of alternating direction (ADMM). SLEP is a variation of Nesterov method like FISTA, although it was specifically implemented for fused-lasso penalty. It is not directly applicable for total variation deblurring problem. We pick two algorithms to compare with ALIN in this numerical study: FISTA of Beck and Teboulle (2009), a very efficient first-order method for discrete total variation based image processing; and TVAL (Li et al., 2013), a method based on Augmented Lagrangian and Alternating Direction algorithm. TVAL solves a model equivalent to (20), but with a coefficient μ in front of the least-squares term, instead of λ at the regularization.

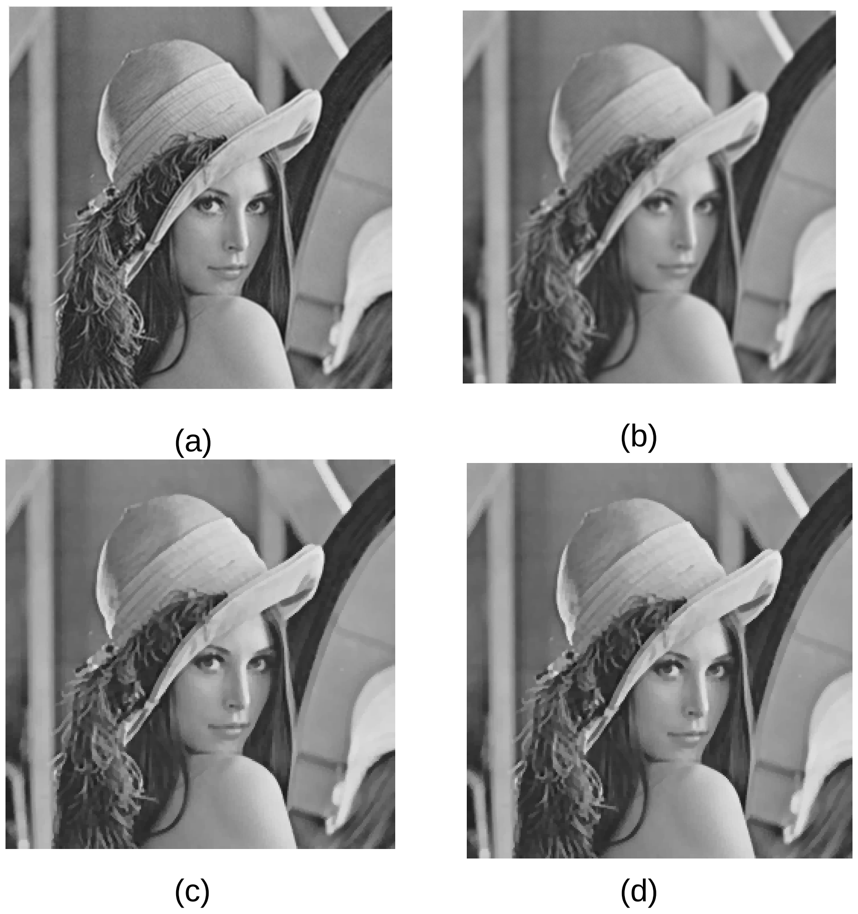

Figure 6: Results of deblurring using fused lasso penalty. Plots (a), (b), (c), and (d) correspond to the original image, the blurred image, the ALIN de-blurred image, and the FISTA de-blurred image, respectively.

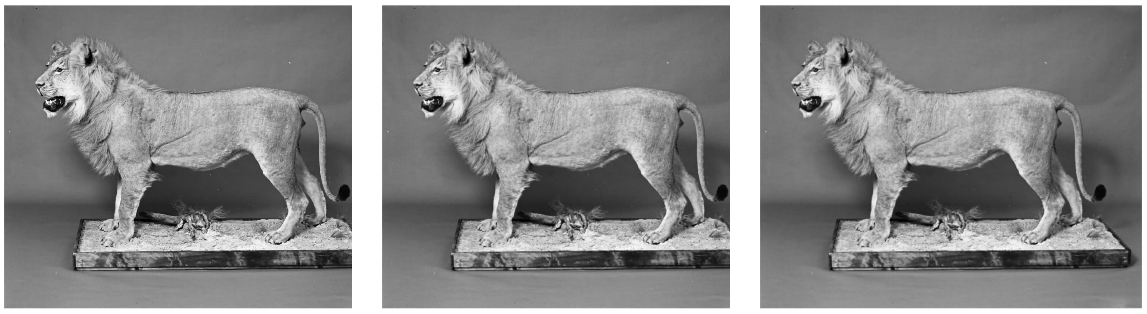

Figure 7: Image deblurring on the "lion" data. The left plot is the result from the ℓ_1-TV penalty; the middle plot is from the Tikhonov penalty. The rightmost plot is the noisy and blurred image.

In the first comparison, we pick 10 random gray scale images with small size, typically 205×205, or approximately $40,000$ pixels. Following the same procedure as described by Beck and Teboulle (2009), the image is blurred using a 3×3 kernel and a Gaussian noise with variance 10^{-2} is added. The deblurring procedure is run with a few different values for λ. We let FISTA run for 100 iterations with the *monotone* feature, which results in a monotonic decrease of the objective function decrease, and the tolerance is set to 10^{-5}. Then we run ALIN to the same objective function value. TVAL, unfortunately, cannot reach the same objective function value. Thus we let TVAL run $10,000$ iterations or to "tol" $= 10^{-5}$, whichever comes first. To compare the quality of the restored image, we use the signal-to-noise (SNR) ratio defined as

$$SNR = 10 \log 10 \frac{\|u^0 - \tilde{u}\|^2}{\|u^0 - u\|^2},$$

where u^0 is the original image, $\tilde{u}$ is the mean intensity of the original image, and u is the restored image. In Table 5.4, we report the running time to produce the best quality restored image, where the regularization parameter $\lambda = 10^{-4}$, similar to what was suggested by (Li et al., 2013). We also report SNR and the mean squared error (MSE).

Although TVAL has superior performance in terms of running time, when compared to FISTA and ALIN, it produces an image of lower quality. With the same value of parameter λ, TVAL was not able to obtain the same objective function value as FISTA and ALIN. This makes the SNR of the TVAL-restored image lower and the error higher than those of ALIN and FISTA. ALIN and FISTA have similar performance in terms of image quality, but ALIN is more efficient than FISTA. In Figure 8, we plot the progression in terms of the objective function values for all three methods. TVAL takes only 52 iterations to terminate. In this plot, ALIN and FISTA are set to terminate in 52 iterations.

In the second comparison, we pick 10 random gray scale images with medium size, ranging from $200,000$ to $500,000$ pixels. The experiment is carried out in the same manner as the previous one. The results are reported in Table 6, and we observe the same pattern as in the previous comparison.

Method	CPU time (secs)	SNR	MSE
FISTA	9.19	11.00	4.21
ALIN	6.85	11.03	4.18
TVAL	3.03	10.56	4.57

Table 5: Run time comparison on image deblurring: small size images.

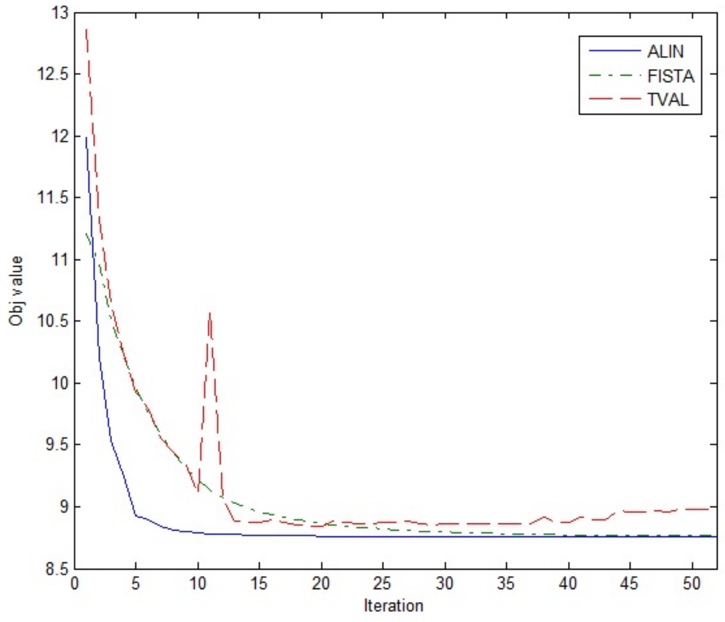

Figure 8: Progression in terms of objective function values of ALIN, FISTA, and TVAL

Method	CPU time (secs)	SNR	MSE
FISTA	65.41	12.14	5.98
ALIN	41.28	12.14	5.97
TVAL	7.18	8.30	14.36

Table 6: Run time comparison on image deblurring: medium size images.

5.5 Application to a Narrative Comprehension Study for Children

With high dimensional fused lasso penalty, the constrained optimization problem with identity design matrix is already difficult to solve, and a large body of literature has been devoted to solving this problem. When the design matrix is not full rank, the problem becomes much more difficult. In this section, we apply the three-dimensional fused lasso penalty to an regression problem where the design matrix X contains many more columns than rows.

Specifically, we perform regularized regression between the measurement of children's language ability (the response y) and voxel level brain activity during a narrative comprehension task (the design matrix X). Children develop a variety of skills and strategies for narrative comprehension during early childhood years (Karunanayaka et al., 2010). This is a complex brain function that involves various cognitive processes in multiple brain regions. We are not attempting to solve the challenging neurological problem of identifying all such brain regions for this cognitive task. Instead, the goal of this study is to demonstrate ALIN's ability for solving constrained optimization problems of this type and magnitude.

The functional MRI data are collected from 313 children with ages 5 to 18 (Schmithorst et al., 2006). The experimental paradigm is a 30-second block design with alternating stimulus and control. Children are listening to a story read by adult female speaker in each stimulus period, and pure tones of 1-second duration in each resting period. The subjects are instructed to answer ten story-related multiple-choice questions upon the completion of the MRI scan (two questions per story). The fMRI data were preprocessed and transformed into the Talairach stereotaxic space by linear affine transformation. A uniform mask is applied to all the subjects so that they have measurements on the same set of voxels.

The response variable y is the oral and written language scale (OWLS). The matrix X records the activity level for all the 8000 voxels measured. The objective function is the following:

$$\mathcal{L}(\beta) = \tfrac{1}{2}\|y - X\beta\|_2^2 + \lambda_1 h_1(\beta) + \lambda_2 h_2(\beta),$$

where

$$\begin{aligned}
h_1(\beta) = {} & \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\sum_{k=1}^{p-1}\{|\beta_{i,j,k} - \beta_{i+1,j,k}| + |\beta_{i,j,k} - \beta_{i,j+1,k}| + |\beta_{i,j,k} - \beta_{i,j,k+1}|\} \\
& + \sum_{i=1}^{m-1}\sum_{k=1}^{p-1}\{|\beta_{i,n,k} - \beta_{i+1,n,k}| + |\beta_{i,n,k} - \beta_{i,n,k+1}|\} + \sum_{j=1}^{n-1}\{|\beta_{m,j,p} - \beta_{m,j+1,p}|\} \\
& + \sum_{j=1}^{n-1}\sum_{k=1}^{p-1}\{|\beta_{m,j,k} - \beta_{n,j+1,k}| + |\beta_{m,j,k} - \beta_{m,j,k+1}|\} + \sum_{i=1}^{m-1}\{|\beta_{i,n,p} - \beta_{i+1,n,p}|\} \\
& + \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\{|\beta_{i,j,p} - \beta_{i+1,j,p}| + |\beta_{i,j,p} - \beta_{i,j+1,p}|\} + \sum_{k=1}^{p-1}\{|\beta_{m,n,k} - \beta_{m,n,k+1}|\}, \\
h_2(\beta) = {} & \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}|\beta_{i,j,k}|,
\end{aligned}$$

and $m = 31$, $n = 35$, and $p = 15$.

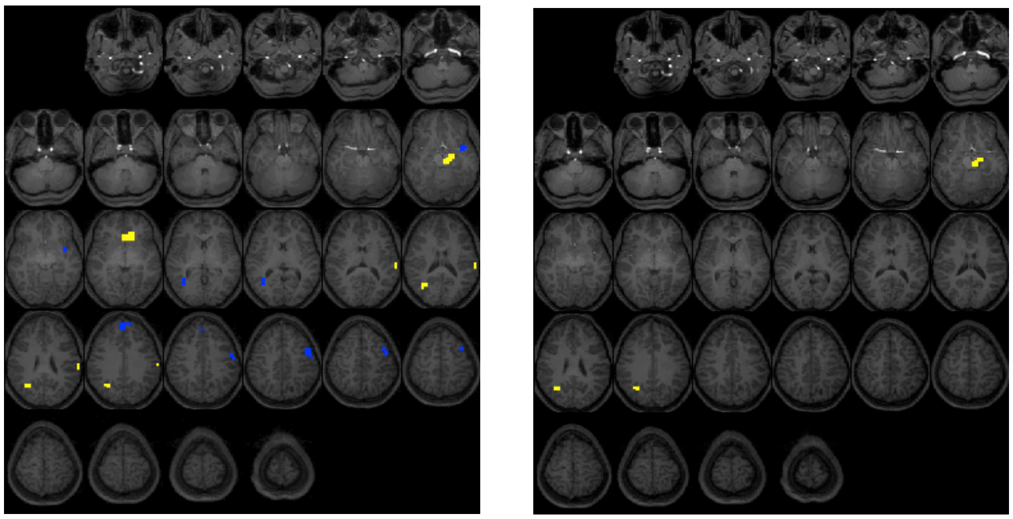

Figure 9: Results of regularization regression with combined lasso and 3-d fused lasso penalty. The tuning parameters of fused lasso is 0.2 for both figures. The tuning parameter for lasso is 0.2 for the left and 0.6 for the right.

While the main purpose of this study is to demonstrate the capability of the ALIN algorithm for solving penalized regression problems with 3-d fused lasso, there are also some interesting neurological observations. One objective of this study is to identify the voxels that are significant for explaining the performance score y. These voxels constitute active brain regions that are closely related to the OWLS. Figure 9 presents the results of fitted coefficients using combined lasso and fused lasso penalty. The highlighted regions shown in the maps are areas with more than 10 voxels (representing clusters of size 10 and above). The left plot in the figure is the optimal solution obtained using ten-fold cross validation. The optimal tuning parameters are 0.2 for both fused lasso and lasso penalties. Roughly speaking, five brain regions have been identified. The yellow area to the rightmost side of the brain is situated in the wernicke area, which is one of the two parts of the cerebral cortex linked to speech. It is involved in the understanding of written and spoken language. The only difference between the left and right plots is the value of the tuning parameter for the lasso penalty, which is 0.2 and 0.6 respectively. Clearly, the right plot shrinks more coefficients to zero, which results in a reduced number of significant regions, as compared to the left plot.

We further study this regularization problem using only lasso penalty and Tiknonov type penalty similar to (23). Figure 10 shows the fitted maps. The left plot is the case where only lasso penalty is applied. Comparing this with Figure 9, we see that the 3-d fused lasso penalty imposes smoothing constraints on the neighboring coefficients, thus allowing to identify larger areas significant for the response variable y. The simple lasso penalty imposes shrinkage on the coefficients individually, resulting in rather disjoint significant voxels. Such scatterness is much less informative for neurologists than larger areas identified

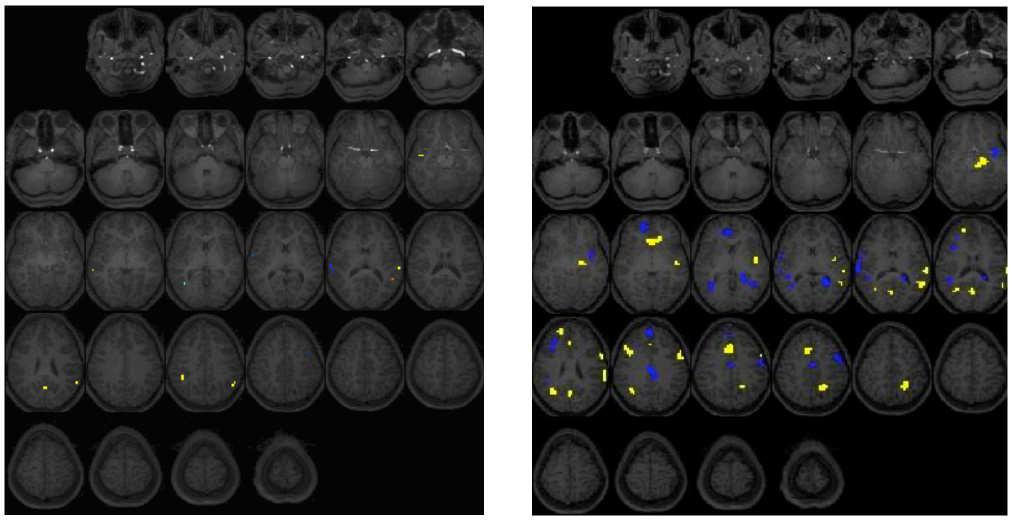

Figure 10: Results of regularization regression with lasso penalty (left plot) and Tiknonov type penalty (right plot).

by the three-dimensional fused lasso penalty. Meanwhile, the Tiknonov type regularization generates too many significant regions as shown in the right plot. This is partially due to the over smoothing of the image as discussed in the previous section.

For comparison, we have considered a couple of methods designed to solve this particular problem. *Genlasso* is the path algorithm designed for the Generalized Lasso in the original paper. However, it was unable to handle an instance of this magnitude.

Another method is the Augmented Lagrangian and Alternating Direction method. The problem of interest

$$\min_{\beta} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|R\beta\|_1, \quad \lambda > 0$$

can be reformulated as

$$\min_{\beta} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|z\|_1 \quad s.t : R\beta - z = 0.$$

The Augmented Lagrangian for this problem has the form:

$$\mathcal{L}(\beta, z, u) = \frac{1}{2}\|y - A\beta\|_2^2 + \lambda\|z\|_1 + \mu^T(R\beta - z) + \frac{\rho}{2}\|R\beta - z\|_2^2,$$

where μ is the vector of Lagrange multipliers, and $\rho > 0$ is the penalty coefficient. This formulation can be solved by the Alternating Direction method, in which alternating minimization with respect to β and z are followed by an update of μ. We implemented this method in MATLAB. The update step for β requires minimizing a quadratic function. The iterative method of choice to solve the sub-problems is the conjugate gradient method built in MATLAB. The performance of this method strongly depends on the choice of the penalty parameter ρ. As suggested by Wahlberg et al. (2012), we choose $\rho = \lambda$ to keep the algorithm stable for the implementation. It is known that the method has a nice decrease in the function values but slow tail convergence and an iteration of ADMM for this particular problem is rather expensive. Because of these reasons, we let the method run for 30 iterations and then let ALIN run until it reached the same objective function value. The running times for different values of λ are reported in Table 7.

Method	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.05$	$\lambda = 0.1$	$\lambda = 1$
ALIN	20.68	12.19	18.58	23.45	129.81
ADMM	72.86	94.69	68.13	74.43	267.01

Table 7: Run time (in seconds) comparison on 3D fused lasso.

We also implemented FISTA for this particular problem. For each iteration k of FISTA, the following optimization problem is solved:

$$\min_{\beta} \mathcal{Q}_L(\beta, \beta^k) = f(\beta^k) + \langle \nabla f(\beta^k), \beta - \beta^k \rangle + \frac{L}{2}\|\beta - \beta^k\|^2 + g(\beta),$$

where f is Gaussian loss function and $g(\beta) = \|R\beta\|_1$. The parameter L is the Lipschitz constant of ∇f. This problem is similar to the f-subproblem of ALIN, so we utilize our own block-coordinate descent method to solve this problem. Since the Lipschitz constant L cannot be computed efficiently, we use back-tracking to find the proper L. Back-tracking is a popular method to find the right step size for FISTA iterations, however it can slow down the algorithm significantly. We observe that in each iteration, back-tracking will have to solve the sub-problem 20 to 30 times to find a good approximation to the Lipschitz constant of ∇f. Normally, this quantity is approximated by the maximum eigenvalue of $X^T X$. However, in the $p \gg n$ setting, it is not computationally efficient to estimate eigenvalues. When FISTA is applied to this data set with 3-d fused lasso penalty, it needs around $10,000$ iterations to reach the same objective function value as ADMM, and the running time is about one hour. This is also in line with what we observe from the implementation of FISTA for 1-d fused lasso penalty in the package SPAMS (Jenatton et al., 2010).

6. Conclusion

The alternating linearization method is a specialized nonsmooth optimization method for solving structured nonsmooth optimization problems. It combines the ideas of bundle methods and operator splitting methods, to define a descent algorithm in terms of the values of the function that is minimized. We have adapted the alternating linearization method to structured regularization problems by introducing the idea of diagonal quadratic approximation and developing specialized methods for solving subproblems. As a result, a new general method for a variety of regularization problems has been obtained, which has the following theoretical features:

- It deals with nonsmoothness directly, not via approximations,
- It is monotonic with respect to the values of the function that is minimized,
- Its convergence is guaranteed theoretically.

Our numerical experiments on a variety of structured regularization problems illustrate the applicability of the alternating linearization method and indicate its practically important virtues: speed, scalability, and accuracy. While other specialized methods compare to ALIN in some cases, none of them exhibits equal speed and accuracy for the broad range of problems discussed in the paper.

Its efficacy and accuracy follow from the use of the diagonal quadratic approximation and from a special test, which chooses in an implicit way the best operator splitting step to be performed. The current approximate solution is updated only if it leads to a significant decrease of the value of the objective function.

Its scalability is due to the use of highly specialized algorithms for solving its two subproblems. The algorithms do not require any explicit matrix formation or inversion, but only matrix–vector multiplications, and can be efficiently implemented with sparse data structures.

Our study of image denoising and deblurring in Section 5.4, as well as the narrative comprehension for children in Section 5.5 are illustrations of broad applicability of the alternating linearization method.

Acknowledgements

The authors are grateful to the Associate Editor and four anonymous Referees, whose insightful comments helped to make significant improvements in the results reported in this manuscript. The work of the third author was partially supported by the National Science Foundation (award CMMI-0965689).

References

H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

E. G. Birgin and J. M. Martínez. Large-scale active-set box-constrained optimization method with spectral projected gradients. *Comput. Optim. Appl.*, 23(1):101–125, 2002.

J.-F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Numerical Optimization. Theoretical and Practical Aspects.* Springer-Verlag, Berlin, 2003.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

X. Chen, S. Kim, Q. Lin, J. Carbonell, and E. Xing. Graph-structured multi-task regression and an efficient optimization method for general fused lasso. Technical report 1005.3579v1, arXiv, 2010.

X. Chen, Q. Lin, S. Kim, J. Carbonell, and E. Xing. Smoothing proximal gradient method for general structured sparse regression. *Ann. Appl. Stat.*, 6(2):719–752, 2012.

P. L. Combettes. Iterative construction of the resolvent of a sum of maximal monotone operators. *J. Convex Anal.*, 16(3-4):727–748, 2009.

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, Springer Optimization and Its Applications, pages 185–212. Springer, 2011.

I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math*, 57(11):1413–1457, 2004.

E. Van den Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM J. on Scientific Computing.*, 2008.

J. Douglas and H. H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Trans. Amer. Math. Soc.*, 82:421–439, 1956.

J. Eckstein. Augmented Lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results. *RUTCOR Research Report, Rutgers University*, RRR 32-2012, 2012.

J. Eckstein and D. P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Programming*, 55(3, Ser. A):293–318, 1992.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–451, 2004.

J.M. Fadili and G. Peyré. Total variation projection with first order schemes. *IEEE Transactions on Image Processing*, 20(3):657 –669, 2011.

A. Friedlander and J. M. Martínez. On the maximization of a concave quadratic function with box constraints. *SIAM J. Optim.*, 4(1):177–192, 1994.

J. Friedman, T. Hastie, H. Hoefling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

W. Fu. Penalized regressions: the bridge vs. the lasso. *Journal of Computational and Graphical Statistics*, 7(3):397–416, 1998.

D. Goldfarb, S. Ma, and K. Scheinberg. Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming*, (141):349–382, 2013.

T. Goldstein and S. Osher. The split Bregman method for $L1$-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–343, 2009.

G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

L. Grosenick, B. Klingenberg, B. Knutson, and J. Taylor. A family of interpretable multivariate models for regression and classification of whole-brain fmri data. *arXiv/1110.4139*, 2011.

L. Grosenick, B. Klingenberg, K. Katovich, B. Knutson, and J. Taylor. Interpretable whole-brain prediction analysis with graphnet. *Neuroimage*, 2013.

B. He and X. Yuan. On the $O(1/t)$ convergence rate of alternating direction method. Technical report, 2011.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms. II*, volume 306 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences].* Springer-Verlag, Berlin, 1993.

H. Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4), 2010.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. *International Conference on Machine Learning*, 2010.

P. Karunanayaka, V. J. Schmithorst, J. Vannest, J. P. Szaflarski, E. Plante, and S. K. Holland. A group independent component analysis of covert verb generation in children: A functional magnetic resonance imaging study. *Neuroimage*, 51:472–487, 2010.

K. Kiwiel, C. Rosa, and A. Ruszczyński. Proximal decomposition via alternating linearization. *SIAM Journal on Optimization*, 9:153–172, 1999.

K. C. Kiwiel. *Methods of Descent for Nondifferentiable Optimization*, volume 1133 of *Lecture Notes in Mathematics.* Springer-Verlag, Berlin, 1985.

C. Li, W. Yin, H. Jiang, and Y. Zhang. An efficient augmented Lagrangian method with applications to total variation minimization. *Computational Optimization and Applications*, 56(3):507–530, 2013.

P.-L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM J. Numer. Anal.*, 16(6):964–979, 1979.

J. Liu, L. Yuan, and J. Ye. An efficient algorithm for a class of fused lasso problems. *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2010.

J. Liu, L. Yuan, and J. Ye. SLEP: Sparse learning with efficient projections. Technical report, Computer Science Center, Arizona State University, 2011.

V. Michel, A. Gramfort, G. Varoquaux, E. Eger, and B. Thirion. Total variation regularization for fMRI-based prediction of behavior. *IEEE Transactions on Medical Imaging*, 30(7):1328–1340, 2011.

Yu. Nesterov. Gradient methods for minimizing composite objective function. Discussion paper 2007/76, ECORE, 2007.

D. W. Peaceman and H. H. Rachford. The numerical solution of parabolic and elliptic differential equations. *J. Soc. Indust. Appl. Math.*, 3:28–41, 1955.

Z. Qin and D. Goldfarb. Structured sparsity via alternating direction methods. *Journal of Machine Learning Research*, pages 1435–1468, 2012.

L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena.*, 1992.

A. Ruszczyński. On convergence of an augmented Lagrangian decomposition method for sparse convex optimization. *Math. Oper. Res.*, 20(3):634–656, 1995.

A. Ruszczyński. *Nonlinear Optimization.* Princeton University Press, Princeton, NJ, 2006.

V. Schmithorst, S. Holland, and E. Plante. Cognitive modules utilized for narrative comprehension in children: a functional magnetic resonance imaging study. *Neuroimage*, 29: 254–266, 2006.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of The Royal Statistical Society Series B*, 58(1):267–288, 1996.

R. Tibshirani and J. Taylor. The solution path of the generalized lasso. *Annals of Statistics*, 39(3), 2011.

R. Tibshirani and P. Wang. Spatial smoothing and hot spot detection for CGH data using the fused lasso. *Biostatistics*, 9(1):18–29, 2008.

R. Tibshirani, M. Saunders, J. Zhu, and S. Rosset. Sparsity and smoothness via the fused lasso. *Journal of The Royal Statistical Society Series B*, 67:91–108, 2005.

P. Tseng. Convergence of block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:474–494, 2001.

B. Wahlberg, S. Boyd, M. Annergren, and Y. Wang. An ADMM algorithm for a class of total variation regularized estimation problems. *arXiv:1203.1828*, 2012.

Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7), 2009.

L. Xiao and T. Zhang. A proximal-gradient homotopy method for the sparse least-squared problem. *SIAM Journal on Optimization.*, 23(2):1062–1091, 2013.

G.-B. Ye and X. Xie. Split Bregman method for large scale fused Lasso. *Comput. Statist. Data Anal.*, 55(4):1552–1569, 2011.

Z. Zhang, K. Lange, and C. Sabatti. Reconstructing DNA copy number by joint segmentation of multiple sequences. *BMC Bioinformatics*, 13(205), 2012.