A Proximal Alternating Direction Method for $\ell_{2,1}$-Norm Least Squares Problem in Multi-Task Feature Learning

Yunhai Xiao*, Soon-Yi Wu† and Bing-Sheng He‡

Abstract

The joint feature selection problem arises in many fields including computer vision, text classification and biomedical informatics. Generally, recent results show that it can be realized by solving a $\ell_{2,1}$-norm involved minimization problem. However, solving the optimization problem is a challenging task due to the non-smoothness of the regularization term. In this paper, we reformulate the problem to an equivalent constrained minimization problem by introducing an auxiliary variable. We split the corresponding augmented Lagrange function and minimize the subproblem alternatively with one variable by fixing the other one. Moreover, we linearize the subproblem and add a proximal-point term when the closed-form solutions are not easily to derived. The convergence analysis and the relatedness with other algorithms are also given. Although the $\ell_{2,1}$-norm is mainly considered, we show that the $\ell_{\infty,1}$-norm penalized learning problem can also be readily solved in our framework. The reported experiments on simulated and real data sets show that the proposed method is effective and promising. The performance comparisons illustrate that the proposed algorithm is competitive with even performs little better than the state-of-the-art solver SLEP.

Key words. multi-task feature learning, augmented Lagrangian function, alternating direction method, proximal points, real data sets

1. Introduction

Learning algorithms based on $\ell_1$-norm regularization have had a relatively long history in machine learning. The goal of single task feature learning is to learn a weight vector $x \in \mathbb{R}^n$ from a given training set $\{(a_i, b_i)\}_{i=1}^m$. Let $a_i \in \mathbb{R}^n$ denotes the $i$-th training sample; $b_i$ denotes the corresponding output, which is a real number in a regression case or $\{-1, 1\}$ in a binary classification; $m$ is the number of training samples. Undertaking the square loss, it is appropriate to minimize

$$
\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - b\|_2^2 + \mu \|x\|_1,
$$

where $A = (a_1, \ldots, a_m) \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $\mu > 0$ is a regularization parameter. The main property of the $\ell_1$-norm regularization term is its ability to recover sparse solutions [4].

*Institute of Applied Mathematics, College of Mathematics and Information Science, Henan University, Kaifeng 475004, China (yhxiaoamath@gmail.com). This author's work is supported by the Natural Science Foundation of China NSFC-11001075.
†National Center for Theoretical Sciences (South), National Cheng Kung University, Tainan 700, Taiwan (Email: soonyi@mail.ncku.edu.tw).
‡Department of Mathematics, Nanjing University, Nanjing 210093, China (Email: hebma@nju.edu.cn). This author's work is supported by Natural Science Foundation of China NSFC-10971095.
The tasks in the computer version, text classification and biomedical informatics always relate to each other. Hence, learning them together is significant efficient relative to learning each one independently [1, 3, 6]. Consequently, capturing the relatedness between each task becomes the key issue to learn. In the multi-task feature section, the given training set of \( t \) tasks is \( \{(a_i^j, b_i^j)\}_{i=1}^{m_j} \) (\( j = 1, 2, \ldots, t \)), where the \( i \)-th sample for \( j \)-th task is \( a_i^j \in \mathbb{R}^n \), the corresponding response is \( b_i^j \), and the number of training sample for the \( j \)-th task is \( m_j \). Hence, the total number of training samples is \( m = \sum_{j=1}^t m_j \). Let \( A_j = [a_1^j, \ldots, a_{m_j}^j]^\top \in \mathbb{R}^{m_j \times n} \) denote the data for the \( j \)-th task, \( A = [A_1; \ldots; A_t] \in \mathbb{R}^{m \times n} \), \( b = [b_1^1; \ldots; b_t^m]^\top \in \mathbb{R}^m \). Let \( x_j \in \mathbb{R}^n \) be the sparse feature for \( j \)-th task, and \( X = [x_1, \ldots, x_t] \in \mathbb{R}^{n \times t} \) be the joint feature to be learned. In order to select features globally, it would like to encourage several rows of \( X \) to be zero. Thus, it appreciate the following \( \ell_{2,1} \)-norm minimization problem

\[
\min_{X \in \mathbb{R}^{n \times t}} \frac{1}{2} \sum_{j=1}^t \| A_j x_j - b_j \|_2^2 + \mu \| X \|_{2,1},
\]

where \( \| X \|_{2,1} = \sum_{i=1}^n \| x_i^\top \|_2 \), in which \( x_i^\top \) is the \( i \)-th row of \( X \). The first term in (1.2) is to measure the loss incurred by \( X \) on the training sample \( A \) and \( b \), while the second is a regularization defined by the sum of \( \ell_2 \)-norm of each row. In addition, the regularization parameter \( \mu > 0 \) is used to balance both terms for minimization. The appealing property of the \( \ell_{2,1} \)-norm regularization is that it encourages multiple predictors from different tasks to share similar parameter sparsity patterns [2, 20].

Feature selection problem is primarily in the field of bio-informatics, and later is investigated widely in cooperating with sparsity regularization in [21]. Meanwhile, in the multi-task learning, the \( \ell_{2,1} \)-norm regularization is firstly introduced by Obosinsky et al. [20] and Argyriou et al. [2], and has engendered intensive research activities in recent years. Clearly, problem (1.2) is convex since it is separately convex in each of the terms. However, the non-smoothness of the \( \ell_{2,1} \)-norm makes it is challenging for optimizing. The algorithm in [13] reformulates problem (1.2) equivalently to two smooth convex optimization problems and then minimizes by Nesterov’s [18] gradient method. Another recently developed method in [2] transforms model (1.2) into a constrained optimization problem and minimizes alternately. Extensive experiments show that both methods perform quite well by using several data sets.

In this paper, we focus on the application an alternating direction method (ADM) to solve the \( \ell_{2,1} \)-norm involved minimization problem. In particular, we transform the original model into an equality constrained optimization problem by instructing an artificial vector. We split the task of minimizing the corresponding augmented Lagrangian function into two subproblems and then update the multiplier consequently. Both resulting subproblems have closed-form solutions which can be easily determined by taking the problem’s favorable structures. Under some technical assumptions, we show that the proposed algorithm converges globally. We present experimental results and do some comparisons. The performance comparison demonstrate that the proposed method is fast, efficient, and competitive with the state-of-the-art algorithm SLEP. Moreover, we also show that the proposed method can be extended to solve \( \ell_{\infty,1} \)-norm minimization problem without any technical difficulties.

We summarize the notation used in this paper. Matrices are written as uppercase letters. Vectors are written as lowercase letters. For matrix \( X \), its \( i \)-th row and \( j \)-th column are denoted by \( x_i^\top \) and \( x_{i,j} \), respectively. The Frobenius norm, the \( \ell_{2,1} \)-norm and \( \ell_{\infty,1} \)-norm of the matrix \( X \in \mathbb{R}^{n \times t} \) are respectively defined as

\[
\| X \|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^t x_{i,j}^2}, \quad \| X \|_{2,1} = \sum_{i=1}^n \sum_{j=1}^t x_{i,j}^2 = \sum_{j=1}^t \| x_j^\top \|_2, \quad \text{and} \quad \| X \|_{\infty,1} = \sum_{i=1}^n \max_{1 \leq j \leq t} | x_{i,j} | = \sum_{i=1}^n \| x_i^\top \|_\infty,
\]
where \( x_{i,j} \) is the \((i, j)\)-th component of \( X \). For any two matrices \( X, Y \in \mathbb{R}^{n \times t} \), we define \( \langle X, Y \rangle = \text{tr}(X^\top Y) \) (the standard trace inner product in \( \mathbb{R}^t \)), so that \( \| X \|_{F, r} = \sqrt{\langle X, X \rangle} \). If \( x \in \mathbb{R}^d \), we denote by \( \text{Diag}(x) \) the diagonal matrix possessing the components of vector \( x \) on the diagonal. For any symmetric \( M \in \mathbb{R}^{m \times m} \), \( \lambda_{\text{max}}(M) \) denotes the largest eigenvalues of \( M \). For vectors \( b_j \in \mathbb{R}^{m_j} \) and \( A_j \in \mathbb{R}^{m_j \times n} \), \( j = 1, \ldots, t \), we let \( b = [b_1; \ldots; b_t] = [b_1^\top, \ldots, b_t^\top]^\top \in \mathbb{R}^m \) and \( A = [A_1; \ldots; A_t] = [A_1^\top, \ldots, A_t^\top]^\top \in \mathbb{R}^{m \times n} \). We define “\( \top \)” as the transpose of a vector or a matrix. Additional notation will be introduced when it occurs.

The paper is organized as follows. In Section 2., we briefly recall the classic ADM method and present our proposed method subsequently. In section 3., we show the relationship with other methods and present the global convergence simultaneously. In section 4., we report numerical results on the simulated data and two real data sets and do performance comparisons. Finally, we conclude the paper in Section 5.

### 2. Proposed algorithm

Now, we pay our attention to consider problem (1.2) with the compact form

\[
\min_{X \in \mathbb{R}^{n \times t}} \frac{1}{2} \| A(X) - b \|^2 + \mu \| X \|_{2,1},
\]

(2.1)

where \( A : \mathbb{R}^{m \times n} \to \mathbb{R}^m \) is a map defined by matric-vector multiplication based on each task, i.e., \( A(X) = [A_1 x_1; \ldots; A_t x_t] \in \mathbb{R}^m \). By adding an auxiliary vector \( r \in \mathbb{R}^m \), model (2.1) is equivalent to

\[
\min_{X \in \mathbb{R}^{n \times t}} \frac{1}{2} \| r \|^2 + \mu \| X \|_{2,1}
\]

s.t. \( A(X) - b = r \).

(2.2)

The corresponding augmented Lagrangian function for this problem is

\[
\mathcal{L}_A(X, r, \lambda) = \frac{1}{2} \| r \|^2 + \mu \| X \|_{2,1} - \lambda^\top [A(X) - b - r] + \frac{\beta}{2} \| A(X) - b - r \|^2,
\]

(2.3)

where \( \lambda \in \mathbb{R}^m \) is a multiplier and \( \beta > 0 \) is a penalty parameter. The basic idea of ADM goes back to the work of Glowinski & Marocco [9], and Gabay & Mercier [7]. ADM method takes advantage of the problem’s separable structure which minimizes (2.3) firstly with respect to \( X \) and then with respect to \( r \) serially, rather than with respect to both \( X \) and \( r \) simultaneously. More specifically, for fixed \((X_k, r_k, \lambda_k)\), ADM generates the next iterate:

\[
\begin{align*}
X_{k+1} &\leftarrow \text{arg} \min_{X} \mathcal{L}_A(X, r_k, \lambda_k), \\
r_{k+1} &\leftarrow \text{arg} \min_{r} \mathcal{L}_A(X_{k+1}, r, \lambda_k), \\
\lambda_{k+1} &\leftarrow \lambda_k - \beta [A(X_{k+1}) - b - r_{k+1}].
\end{align*}
\]

(2.4)

We clearly see that when the ADM applies to (2.3), the main computation per-iteration is solving the subproblem for \( X \) and \( r \). Firstly, it is easy to deduce that

\[
X_{k+1} \leftarrow \text{arg} \min_{X \in \mathbb{R}^{n \times t}} \mathcal{L}_A(X, r_k, \lambda_k)
\]

\[
= \text{arg} \min_{X \in \mathbb{R}^{n \times t}} \mu \| X \|_{2,1} - \lambda_k^\top [A(X) - b - r_k] + \frac{\beta}{2} \| A(X) - b - r_k \|^2
\]

\[
= \text{arg} \min_{X \in \mathbb{R}^{n \times t}} \mu \| X \|_{2,1} + \frac{\beta}{2} \| A(X) - b - r_k - \frac{\lambda_k}{\beta} \|^2.
\]

(2.5)
Unfortunately, for the general case $\mathcal{A}$ and the non-smoothness of $\ell_{2,1}$, solving (2.3) exactly is highly nontrivial. We let $\mathcal{A}^* : \mathbb{R}^m \to \mathbb{R}^{n \times t}$ be a map defined by matrix-vector multiplication based on each task, i.e., $\mathcal{A}^*(b) = [A^1_1 b_1, \ldots, A^1_t b_1] \in \mathbb{R}^{n \times t}$. At current iteration, similar with [25, 26, 22], we let

$$G_k = \mathcal{A}^*[\mathcal{A}(X_k) - b - r_k - \lambda_k / \beta]$$

denote the gradient of $\frac{1}{2} \| \mathcal{A}(X) - b - r_k - \lambda_k / \beta \|_2^2$ at $X_k$. Consequently, we approximate (2.5) by

$$\min_{X \in \mathbb{R}^{n \times t}} \mu \|X\|_{2,1} + \beta \left( \langle G_k, X - X_k \rangle + \frac{1}{2\tau} \|X - X_k\|_{F,\tau}^2 \right),$$

or equivalently, by

$$\min_{X \in \mathbb{R}^{n \times t}} \mu \|X\|_{2,1} + \frac{\beta}{2\tau} \|X - (X_k - \tau G_k)\|_{F,\tau}^2,$$

where $\| \cdot \|_{F,\tau}$ is the Frobenius norm, $\tau > 0$ is a positive parameter, and $\langle \cdot \rangle$ denotes the standard trace inner product for matrix.

For the sake of simplicity, we let $V = X_k - \tau G_k$, and $v^i$ be the $i$-th row of $V$. The solution of (2.7) takes the following form:

$$(X_{k+1} = \) \hat{X} = \arg \min_{x \in \mathbb{R}^{n \times t}} \sum_{i=1}^n \left( \mu \|x^i\|_2 + \frac{\beta}{2\tau} \|x^i - v^i\|_2^2 \right),$$

which shows that the problem (2.7) can be decomposed into $n$ separate subproblems of dimension $t$, i.e.,

$$\min_{x^i \in \mathbb{R}^n} \mu \|x^i\|_2 + \frac{\beta}{2\tau} \|x^i - v^i\|_2^2, \quad i = 1, 2, \ldots, n.$$

It is easy to see that the optimal solution $\hat{x}^i$ must be in the direction of $v^i$ and takes the form $\hat{x}^i = sv^i$ with scalar $s \geq 0$. By constructing the Lagrangian dual form, the closed-form solutions of (2.8) can be easily obtained by (see e.g., [11, 14, 19])

$$\hat{x}^i = \left( 1 - \frac{\tau \mu}{\beta \|v^i\|_2} \right) + v^i, \quad i = 1, \ldots, n,$$

where $(\cdot)_+ = \max(\cdot, 0)$. By summarizing, the solution of (2.7) can be written explicitly as

$$X_{k+1} = \begin{pmatrix} \left( 1 - \frac{\tau \mu}{\beta \|X_k - \tau G_k\|_2} \right)^+ (X_k - \tau G_k) & 1 \\ \left( 1 - \frac{\tau \mu}{\beta \|X_k - \tau G_k\|_2} \right)^+ (X_k - \tau G_k) & 2 \\ \vdots \\ \left( 1 - \frac{\tau \mu}{\beta \|X_k - \tau G_k\|_2} \right)^+ (X_k - \tau G_k) & n \end{pmatrix}.\tag{2.9}$$

Secondly, for the given pair $(X_{k+1}, \lambda_k)$, minimizing (2.3) respect to $r$ is equivalent to

$$\min_r \frac{1}{2} \|r\|_2^2 - \lambda_k^\top [\mathcal{A}(X_{k+1}) - b - r] + \frac{\beta}{2} \|\mathcal{A}(X_{k+1}) - b - r\|_2^2.$$

The normal equations of (2.10) are given by

$$r + \lambda_k - \beta [\mathcal{A}(X_{k+1}) - b - r] = 0.$$

Therefore, the exact solution is

$$r_{k+1} = \frac{\beta}{1 + \beta} [\mathcal{A}(X_{k+1}) - b - \lambda_k / \beta].\tag{2.11}$$
In light of all derivations above, we can state the steps of the inexact version ADM method for the multi-task feature learning (IADM_MFL) as follows.

Algorithm 2.1. (IADM_MFL)

Initialization: Given $X_0$ and $\lambda_0$. Constants $\mu$, $\beta$, and $\tau$. Set $k = 0$.

Step 1. Stop if some terminated condition is satisfied. Otherwise, continue.

Step 2. Compute $X_{k+1}$ via (2.9) with fixed $r_k$ and $\lambda_k$;

Step 3. Compute $r_{k+1}$ via (2.11) with fixed $X_{k+1}$ and $\lambda_k$.

Step 4. Update $\lambda_{k+1}$ via $\lambda_{k+1} = \lambda_k - \beta[A(X_{k+1}) - b - r_{k+1}]$.

Step 5. Let $k = k + 1$. Go to Step 1.

To end this section, we briefly describe how to extend our algorithm to solve $\ell_{1,\infty}$-norm regularized learning problem and thus broaden the universality of the algorithm. Similarly, the $X$-subproblem (2.8) is replaced by

$$(X_{k+1} = \hat{X} = \arg\min_{x^1, \ldots, x^n} \sum_{i=1}^n \left( \mu \|x^i\|_{\infty} + \frac{\beta}{2\tau} \|x^i - v^i\|^2 \right)). \tag{2.12}$$

For each subproblem, the optimal $\hat{x}^i$ is determined by

$$\min_{x^i} \mu \|x^i\|_{\infty} + \frac{\beta}{2\tau} \|x^i - v^i\|^2. \tag{2.13}$$

Since the conjugate of a quadratic function is still a quadratic function and the conjugate of the $\ell_{1,\infty}$-norm is a $\ell_1$-norm barrier function, the dual problem of (2.13) is

$$\max_{\alpha^i} \frac{1}{2} \|\alpha^i - v^i\|^2, \quad \text{s.t.} \quad \|\alpha^i\|_1 \leq \frac{\mu \tau}{\beta}. \tag{2.14}$$

The dual vector $\alpha^i$ satisfies the relation $\alpha^i = v^i - x^i$, which can be attained by smooth gradient algorithm with an efficient Euclidean projection on $\ell_1$-ball. Let $u^i_j$ be the absolute value of $v^i_j$ for $j = 1, \ldots, t$, i.e., $u^i_j = |v^i_j|$. Reorder the vector $u^i$ to a decreasing sequence such that $u^i_1 \geq u^i_2 \geq \ldots \geq u^i_t$, and find $\hat{j}$ be the largest number of $j$ which satisfies

$$\hat{j} = \max \left\{ j : \frac{\mu \tau}{\beta} - \sum_{\rho=1}^j (u^\rho_j - u^\rho_j) > 0 \right\}.$$

From [5], we see that each component of explicit solution of (2.13) obeys

$$\hat{x}^i_j = \text{sign}(v^i_j) \min \left( |v^i_j|, \left( \sum_{\rho=1}^j u^\rho_j - \frac{\mu \tau}{\beta} \right) / \hat{j} \right), \quad j = 1, \ldots, t.$$

Then, it also easily derive the ADM algorithmic framework to solve $\ell_{1,\infty}$-norm involved minimization problem.

3. Related work and convergence analysis

3.1. Preliminaries

To easily show that IADM_MFL converges globally, in this section, we turn to consider the structural variable inequality problem (VIP): Finding a vector $w^* \in \Omega$ such that

$$(w - w^*)^\top Q(w^*) \geq 0, \quad \forall \ w \in \Omega, \tag{3.1}$$
where $\Omega$ is a nonempty closed convex set of $\mathbb{R}^{q+p}$, $Q$ is a map from $\mathbb{R}^{q+p}$ to itself, and

$$
w = \begin{pmatrix} x \\ y \end{pmatrix}, \quad Q(w) = \begin{pmatrix} f(x) \\ g(y) \end{pmatrix},
$$

$$
\Omega = \{(x, y) : x \in \mathcal{X}, y \in \mathcal{Y}, Mx + Ny = 0\},
$$

in which $M$ and $N$ are $l \times q$ and $l \times p$ matrices respectively; $\mathcal{X} \subset \mathbb{R}^q$ and $\mathcal{Y} \subset \mathbb{R}^p$ are nonempty closed convex sets, and $f$ and $g$ are given monotone operators. By attaching a Lagrange multiplier vector $\lambda \in \mathbb{R}^l$ to the linear constraint $Mx + Ny = 0$, (3.1) is equivalent to finding $u^* \in \mathcal{U}$ such that

$$
(u - u^*)^\top F(u^*) \geq 0, \quad \forall u \in \mathcal{U},
$$

where

$$
u = \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix}, \quad F(u) = \begin{pmatrix} f(x) - M^\top \lambda \\ g(y) - N^\top \lambda \\ Mx + Ny \end{pmatrix}, \quad \mathcal{U} = \mathcal{X} \times \mathcal{Y} \times \mathbb{R}^l.
$$

To solve (3.1), a number of decomposition methods have been developed. With the given pair $(x_k, y_k)$, the method of Gabay and Mercier [7] generates the new iteration $(x_{k+1}, y_{k+1})$ by solving two monotone sub-VIP problems respectively:

1. With fixed $y_k$ and $\lambda_k$, $x_{k+1} \in \mathcal{X}$ is obtained via

$$
(x' - x)^\top \{f(x) - M^\top [\lambda_k - \beta(Mx + Ny_k)]\} \geq 0, \quad \forall x' \in \mathcal{X}.
$$

2. With fixed $x_{k+1}$ and $\lambda_k$, $y_{k+1} \in \mathcal{Y}$ is computed via

$$
(y' - y)^\top \{g(y) - N^\top [\lambda_k - \beta(Mx_{k+1} + Ny)]\} \geq 0, \quad \forall y' \in \mathcal{Y},
$$

where $\beta > 0$ is a given penalty parameter. However, in many cases, solving such problems are challenging. The alternating direction proximal points method of He et al. [10] improves its performance by solving two strongly monotone sub-VIP problems. That is, at each iteration, instead of (3.5) and (3.6), they solve

$$
(x' - x)^\top \{f(x) - M^\top [\lambda_k - H_k(Mx + Ny_k)] + R_k(x - x_k)\} \geq 0, \quad \forall x' \in \mathcal{X},
$$

and

$$
(y' - y)^\top \{g(y) - N^\top [\lambda_k - H_k(Mx_{k+1} + Ny)] + S_k(y - y_k)\} \geq 0, \quad \forall y' \in \mathcal{Y},
$$

and update Lagrangian multiplier $\lambda$ by

$$
\lambda_{k+1} = \lambda_k - H_k(Mx_{k+1} + Ny_{k+1}),
$$

where $\{H_k\}$, $\{R_k\}$, and $\{S_k\}$ are sequences of both lower and upper bounded symmetric positive semi-definite matrices. The attractive property of this method is that solving both subproblems are relatively easy. Under some mild assumptions and parameter conditions, the global convergence is established [10].

### 3.2. Convergence analysis

Based on the above analysis, we reconsider the iterative form (2.4) in VIP framework. Let $\partial(\cdot)$ denote the subgradient operator of a convex function. It is easy to see that the optimal condition of problem (2.2) can
be characterized by finding $u^* = (X^*, r^*, \lambda^*) \in \mathcal{U} = \mathbb{R}^{n \times 1} \times \mathbb{R}^m \times \mathbb{R}^m$ such that

\[
\begin{aligned}
&
\begin{cases}
\langle X' - X^*, \partial(\mu \|X\|_{2,1}) - \mathcal{A}^* \lambda^* \rangle \geq 0, \\
\langle r' - r^*, (r + \lambda^*) \rangle \geq 0, \\
(\lambda' - \lambda^*)^\top [\mathcal{A}(X^*) - r^* - b] \geq 0,
\end{cases}
\forall u' = (X', r', \lambda') \in \mathcal{U}.
\end{aligned}
\tag{3.9}
\]

It is equivalent to
\[
(u' - u^*)^\top F(u^*) \geq 0, \quad \forall u' \in \mathcal{U},
\tag{3.10}
\]

where
\[
u = \begin{pmatrix} X \\ r \\ \lambda \end{pmatrix} \quad \text{and} \quad F(u) = \begin{pmatrix} \partial(\mu \|X\|_{2,1}) - \mathcal{A}^* \lambda \\ r + \lambda \\ \mathcal{A}(X) - r - b \end{pmatrix}.
\]

Similar to (3.5)-(3.6), at each iteration, the ADM of Gabay and Mercier [7] solves the following two sub-VIP respectively:
\[
\begin{aligned}
&
\begin{cases}
\langle X' - X, \partial(\mu \|X\|_{2,1}) - \mathcal{A}^* [\lambda_k - \beta (\mathcal{A}(X) - b - r_k)] \rangle \geq 0, \\
\langle r' - r_k, r + \lambda_k - \beta (\mathcal{A}(X_{k+1}) - b - r) \rangle \geq 0.
\end{cases}
\end{aligned}
\tag{3.11}
\]

Now, we reconsider the approximated $X$-subproblem in (2.7) for its optimal condition in VIP form
\[
\langle X' - X, \partial(\mu \|X\|_{2,1}) + \frac{\beta}{\tau} [X - (X_k - \tau G_k)] \rangle \geq 0,
\]

or equivalently,
\[
\langle X' - X, \partial(\mu \|X\|_{2,1}) - \mathcal{A}^* [\lambda_k - \beta (\mathcal{A}(X) - b - r_k)] + \langle \beta \frac{1}{\tau} - \mathcal{A}^* \mathcal{A}, X - X_k \rangle \rangle \geq 0.
\tag{3.12}
\]

Comparing with (3.11), it is easily derive that the approximated model (2.7) is exactly the first implementation of (3.11) adding a proximal points term $\langle \beta \frac{1}{\tau} - \mathcal{A}^* \mathcal{A}, X - X_k \rangle$. In other words, our work belongs to the framework of He et al. [10] in sense that $S_k = 0$, $R_k = \beta (\frac{1}{\tau} - \mathcal{A}^* \mathcal{A})$, and $H_k = \beta I$ ($I$ is an identity matrix). Hence, the convergence follows directly provided that $\beta (\frac{1}{\tau} - \mathcal{A}^* \mathcal{A})$ is symmetric and semi-positive definite. Therefore, from [10] Theorem 4], we have the following convergence theorem for Algorithm 2.1.

**Theorem 3.1.** For any $\beta > 0$, the sequence $\{(X_k, r_k)\}$ generated by Algorithm 2.1 from any starting point $(X_0, r_0)$ converges to a solution of (2.2) provided by $0 < \tau \leq 1/\lambda_{\max}(\mathcal{A}^* \mathcal{A})$.

In the end of this section, we show that our method is also valid by adopting a relaxation factor $\gamma$. That is, Step 4 in Algorithm 2.1 is replaced by
\[
\lambda_{k+1} = \lambda_k - \beta \gamma [\mathcal{A}(X_{k+1}) - b - r_{k+1}],
\tag{3.13}
\]

where $\gamma \in (0, \frac{\sqrt{5} + 1}{2})$ is a positive scalar. The instruction of $\gamma$ is suggested firstly by Glowinski [8] and the numerical experiments [11] have shown that it always achieves better performance. The adaptive relaxation factor technique is used in proximal ADM for solving structured VIP by Xu [24]. Hence, by the convergence theorem of [24] Theorem 4.1, we present the following theorem without proof.

**Theorem 3.2.** For any $\beta > 0$ and $\gamma \in (0, \frac{\sqrt{5} + 1}{2})$, the sequence $\{(X_k, r_k)\}$ generated by Algorithm 2.1 from any starting point $(X_0, r_0)$ with the updated formula (3.13) converges to a solution of (2.2) provided by $0 < \tau \leq 1/\lambda_{\max}(\mathcal{A}^* \mathcal{A})$. 

7
4. Numerical experiments

In this section, we report numerical results to illustrate the performance of the proposed algorithm. For this purpose, we perform two types of experiments concentrating on the simulated data and real data sets. In each test, we compare our algorithm to the state-of-the-art algorithm SLEP — Sparse Learning with Efficient Projections. We select SLEP for comparison because it has been verified that it outperforms the well-known algorithm MTL-FEAT [2]. For convenience, we use the Matlab package provided by J. Ye to do comparison. All experiments are performed under Window XP and Matlab v7.10 (2010a) running on a PC with an Intel Pentium CPU at 3.2 GHz and 1 GB of memory.

SLEP is a MATLAB package which provides a set of codes for the \( \ell_1 \)-norm, \( \ell_{1,q} \)-norm \( (q > 1) \), and trace norm regularized sparse learning. SLEP reformulates (1.2) to the constrained smooth convex optimization problem

\[
\min_{X, \nu} \frac{1}{2} \|A(X) - b\|_2^2 + \mu \sum_{i=1}^{n} \nu_i, \tag{4.1}
\]

where \( \nu = [\nu_1, \nu_2, \ldots, \nu_n]^T \) and \( \|x^i\|_2 \leq \nu_i \) for all \( i = 1, \ldots, n \), and then solves (4.1) by Nesterov’s first-order method [17, 16] due to its faster convergence rate. In running SLEP, we choose \( m\text{Flag}=1 \) and \( l\text{Flag}=1 \) which means that an adaptive line search is used.

4.1. Test on simulated data

In this subsection, we illustrate the efficiency, and stability of the proposed algorithm by using simulated data. As [2], \( \bar{x}_j \) is selected from a 5-dimensional Gaussian distribution with zero mean and covariance \( \text{diag}\{1, 0.64, 0.49, 0.36, 0.25\} \). To these 5-dimensional \( \bar{x}_j \)'s, we keep adding up to 20 irrelevant dimensions which are exactly zero. The training and test data \( A_j \) are the Gaussian matrices whose elements are generated by Matlab command \( \text{randn}(m_j, n) \). Moreover, the data matrix is re-scaled by dividing \( \lambda_{\text{max}}(A^T A) \), which guarantees the convergence of IADM_MFL given that \( \tau \in (0, 1] \). The response data \( b_j \) is computed by

\[
b_j = A_j \bar{X}_j + \omega,
\]

where \( \omega \) is zero-mean Gaussian noise with standard deviation \( 1.e - 2 \). For each test, we start at zeros points and simply terminate when

\[
\frac{\|X_k - X_{k-1}\|_{F_r}}{\|X_{k-1}\|_{F_r}} < tol, \tag{4.2}
\]

where \( tol > 0 \) is a tolerance. The quality of the solution \( X^* \) is measured by the relative error to \( \bar{X} \), i.e.,

\[
\text{RelErr} = \frac{\|X^* - \bar{X}\|_{F_r}}{\|X\|_{F_r}}.
\]

Firstly, we give some insight into the behavior of IADM_MFL with different values \( \tau \) and \( \beta \). In this test, we take \( \mu = 1e - 2, t = 200, n = 20, tol = 1e - 3 \), and \( m_j = 100 \) for all \( j = 1, 2, \ldots, t \). The results are listed in Table 1 which contains the number of iterations (Iter), the CPU time in seconds (Time), and the relative error (RelErr). Viewing from the table, we conclude that the proposed algorithm works reasonably well for all the test cases. This table also illustrates that the value of \( \beta \) does not influence the performance of the algorithm significantly. It is natural to set \( \beta = 0.01/\text{mean}(|y|) \) and \( \tau = 0.8 \) in the following series of experiments.

*Available at [http://www.public.asu.edu/~jye02/Software/SLEP/index.htm](http://www.public.asu.edu/~jye02/Software/SLEP/index.htm)
Table 1: Numerical Results with different $\beta$ and $\tau$

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>$\beta = 1 / \text{mean}(y)$</th>
<th>$\beta = 1 / \text{mean}(y)$</th>
<th>$\beta = .01 / \text{mean}(y)$</th>
<th>$\beta = .001 / \text{mean}(y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>.6</td>
<td>62</td>
<td>5.58</td>
<td>3.74e-3</td>
<td>50</td>
</tr>
<tr>
<td>.7</td>
<td>54</td>
<td>4.36</td>
<td>3.63e-3</td>
<td>43</td>
</tr>
<tr>
<td>.8</td>
<td>46</td>
<td>3.55</td>
<td>4.21e-3</td>
<td>37</td>
</tr>
<tr>
<td>.9</td>
<td>42</td>
<td>4.30</td>
<td>3.46e-3</td>
<td>34</td>
</tr>
<tr>
<td>1</td>
<td>38</td>
<td>3.27</td>
<td>3.09e-3</td>
<td>31</td>
</tr>
</tbody>
</table>

Secondly, the number of tasks $t$ may also control the amount of performance of the method within a certain extent. In this test, we fixed $\mu = 1.e-1$, $\tau = 0.8$, $\beta = 0.01 / \text{mean}(|y|)$, $m_j = 100$ ($j = 1, \ldots, t$) with different $t = 10, 20, 50, 100, 200$. We present, in Figure 1, the impact of $t$ on the quality of the learned features and the number of iterations when the number input dimensionality increases. From the left plot, we clearly see that the quality of the underlying features generally increase when more tasks are available. Moreover, the relative error of the learned solutions increases moderately with the number of dimensionality input. The right plot shows that adding more tasks leads to less number of iterations.

As shown by Figure 2, both algorithms exhibit similar performance in sense that they derive the desired results with roughly the same accuracy and function values over $\mu < 1e-3$. However, as $\mu$ increases, the relative error and the function values obtained by SLEP become larger; meanwhile, the number of iterations becomes unstable. In summary, these plots verify that IADM_MFL is more stable and not sensitive to the $\mu$ values so much.

4.2. Test on real data sets

In this subsection, we evaluate the proposed method IADM_MFL and compare it with SLEP by using two real data sets. In this class of test, we choose $\tau = 0.8$ and $\beta = 0.01 / \text{mean}(|y|)$. DMOZ is a text categorization data set available at [http://www.dmoz.org/](http://www.dmoz.org/) in which each of the 10 tasks corresponds to one of the subcategories...
of the Arts category. For this data set, we randomly sample 50% of the data from each task for training. We run both algorithms simultaneously to learn the joint feature among the task. When running both codes, we set all the parameter values as the proceeding subsection except for $\mu = 1e-4$. To illustrate the convergence behavior of both solvers, we draw two figures to show the decreasing function values in the first 50 steps as the iterations and CPU time increase. It is clear that each algorithm generates decreasing sequences and eventually attains nearly equal function values at the end. The preliminary numerical comparisons indicate that both algorithms are comparable.

In the second test, we further test the performance of IADM_MFL on the School data set. This set consists of the examination scores of 15362 students from 139 secondary schools in London from 1985 to 1987. Each sample contains 28 attributes. The School data set has 139 tasks corresponding to predicting student performance in each school. The data set has been used widely on the multi-task learning, for instance in [2, 13]. Again, as [2], we randomly take 75% of the example from each task for training, and the rest 25% for testing. We run both codes with given $\mu = 1e-5$ and 200 iterations, and examine the objective function values, the training error rate and the testing error rate behavior when each algorithm proceeds. The fidelity of training and testing data is measured by root mean squared error. The convergence behavior of both algorithms are reported in Figure 4.

As can be seen from the first row of Figure 4, the function values, training error, and testing error derived by IADM_MFL are slightly larger at the very beginning, while become smaller in the rest iteration process. The second row plots shows that IADM_MFL is the faster. In conclusion, this test illustrates that our
proposed method performs little better than the state-of-the-art algorithm SLEP.

5. Conclusions

In this paper, we proposed, analyzed, and tested a proximal alternating direction algorithm for $\ell_{2,1}$-norm least square Regression Problem. The problem always appears in many fields such as in computer version, text classification and biomedical informatics. The proposed algorithm is easily performable in sense that both resulting subproblems with respect to each variable can be determined explicitly. The proposed algorithm is closely related to the another proximal points ADM for structural VIP. Thus, the convergence theorem follows directly. The performance comparisons showed that the proposed method is competitive with even performs litter better than SLEP — the up-to-date state-of-the-art algorithm. Our algorithm is only an extension of the well-known proximal alternating direction algorithm to solve $\ell_{2,1}$-norm regularized minimization problems. Although this motivation is simple, the numerical experiments illustrate that the proposed algorithm is very promising and comparable. Thus, it provides a new approach to solve the joint feather selection problem in multi-task learning. Actually, this is the main numerical contribution of our paper. The capability of the algorithm to solve the $\ell_{\infty,1}$-norm penalized learning problems is out of question. However, we did not test in this paper. This should be our further task. To the end, we hope that our method and its extensions could produce even applications for problems in relevant areas of the machine learning.

Acknowledgements

We thank two anonymous referees for their constructive suggestions which improved the paper greatly.
References


