STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **2**, December 2014, pp 352–367.
Published online in International Academic Press (www.IAPress.org)

Inexact Accelerated Proximal Gradient Algorithms For Matrix $l_{2,1}$-Norm Minimization Problem in Multi-Task Feature Learning

Yaping Hu [1], Zengxin Wei [2,*] and Gonglin Yuan [2]

[1] *School of Science, East China University of Science and Technology,China.*
[2] *College of Mathematics and Information Science, Guangxi University, China.*

Received: 21 September 2014; Accepted: 14 November 2014
Editor: David G. Yu

Abstract In this paper, we extend the implementable APG method to solve the matrix $l_{2,1}$-norm minimization problem arising in multi-task feature learning. We investigate that the resulting inner subproblem has closed-form solution which can be easily determined by taking the problem's favorable structures. Under suitable conditions, we can establish a comprehensive convergence result for the proposed method. Furthermore, we present three different inexact APG algorithms by using the Lipschitz constant, the eigenvalue of Hessian matrix and the Barzilai and Borwein parameter in the inexact model, respectively. Numerical experiments on simulated data and real data set are reported to show the efficiency of proposed method.

Keywords Multi-task feature learning, Matrix $l_{2,1}$-norm regularization, Accelerated proximal gradient method

AMS 2010 subject classifications 65K05, 90C30, 90C25

DOI: 10.19139/soic.v2i4.106

1. Introduction

Consider the following matrix $l_{2,1}$-norm minimization problem

$$\min_{X\in\Re^{n\times t}} \frac{1}{2}\|AX-b\|_2^2+\mu\|X\|_{2,1}, \tag{1}$$

*Correspondence to: College of Mathematics and Information Science, Guangxi University, Nanning, Guangxi, China. E-mail: zxwei@gxu.edu.cn.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

where the matrix $l_{2,1}$-norm $\|X\|_{2,1}$ is defined by the sum of l_2-norm of each row

$$\|X\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{t} X_{i,j}^2} = \sum_{i=1}^{n} \|X_{i,:}\|_2,$$

in which $X_{i,:}$ is the i-th row of X, $X_{:,j}$ is the j-th column of X. In the multi-task feature section, the given training set of t tasks is $\{(a_i^j, b_i^j)\}_{i=1}^{m_j}(j = 1, 2, \cdots, t)$, where $a_i^j \in \Re^n$ is the i-th sample for j-th task, b_i^j is the corresponding response and m_j is the number of training sample for the j-th task, the total number of training samples is $m = \sum_{j=1}^{t} m_j$, $A_j = [a_1^j, \cdots, a_{m_j}^j]^T \in \Re^{m_j \times n}$ denotes the data for the j-th task, $A = [A_1; \cdots; A_t] \in \Re^{m \times n}$, $b_j = [b_1^j, \cdots, b_{m_j}^j]^T \in \Re^{m_j}$, $b = [b_1; \cdots; b_t]^T \in \Re^m$, $X_{:,j} \in \Re^n$ be the sparse feature for j-th task, $X = [X_{:,1}, \cdots, X_{:,t}] \in \Re^{n \times t}$ 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. The first term in problem (1) is to measure the loss incurred by X on the training sample A and b, while the second is a regularization term. In addition, the regularization parameter $\mu > 0$ is used to balance both terms for minimization. The appealing property of the matrix $l_{2,1}$-norm regularization is that it encourages multiple predictors from different tasks to share similar parameter sparsity patterns [1, 12, 13].

Feature selection problem was first introduced in the filed of bio-informatics, and then studied widely involving with sparsity regularization [16]. Also, Obosinsky et al. [13] and Argyriou et al. [1] introduced the matrix $l_{2,1}$-norm regularization problem to the multi-task learning and lots of research has been laid in this topic in recent years. A practical challenge in using the $l_{2,1}$-norm regularization is to develop efficient algorithms to solve the resulting non-smooth optimization problems. In Liu et al. [9] the matrix $l_{2,1}$-norm minimization problem was formulated into two smooth convex optimization problems and then minimized by the Nesterov's gradient method [11]. Argyriou et al. [1] proved that the problem (1) is equivalent to a constrained optimization problem which can be solved by using an iterative alternating algorithm. Recently, Xiao et al. [19] presented a proximal alternating direction method to solve the problem (1) by generating approximate solutions to the matrix $l_{2,1}$-norm minimization problem. Experiments on simulated and real data sets demonstrated that those algorithms are efficient.

In this paper, based on the accelerated proximal gradient method (FISTA) [4], we now develop an inexact APG algorithm to minimize the matrix $l_{2,1}$-norm regularization problem. In the kth iteration with iterator $\bar{X}^k$, we solve the following subproblem

$$\min_{X \in \Re^{n \times t}} \langle \nabla F(\bar{X}^k), X - X^k \rangle + \frac{1}{2} \langle X - X^k, \mathcal{H}^k(X - X^k) \rangle + \mu \|X\|_{2,1}, \quad (2)$$

where $F(X^k) = \frac{1}{2}\|AX^k - b\|_2^2$, $\mathcal{H}^k : \Re^{n\times t} \to \Re^{n\times t}$ is a given self-adjoint positive definite linear operator. In FISTA [4], $\mathcal{H}^k$ is restricted to LI, where I denotes the identity map and L is a Lipschitz constant for ∇F. More significantly, in our algorithm, the subproblem (2) has closed-form solution which can be easily determined by taking the problem's favorable structures and $\mathcal{H}^k$ is not restricted to a Lipschitz constant. Specifically, we present three different implementable choices of the self-adjoint positive definite linear operator $\mathcal{H}^k$ which take advantage of some more useful information: (i) IAPG-EIG use eigenvalue of the Hessian matrix, $\mathcal{H}^k = \lambda_{max}(A^*A)I$; (ii) IAPG-BB, $\mathcal{H}^k = \alpha_k I$, α_k is the Barzilai and Borwein parameter; (iii) IAPG-L use the Lipschitz constant, $\mathcal{H}^k = LI$.

Accelerated proximal gradient (APG) algorithm was first studied by Nesterov [10] for minimizing smooth convex functions, then had been demonstrated to be efficient in solving various convex optimization problems, including composite convex objective functions [4], convex quadratic semidefinite programming problems [6], nuclear norm minimization problems [18] in matrix completion and l_1 minimization problems [17] in compressed sensing. In this paper, the extension of the APG algorithm to the matrix $l_{2,1}$-norm minimization problem in multi-task feature learning is interesting in terms of practical perspective, because it takes computational advantage over alternative algorithms for solving the problem (1). We solve the matrix $l_{2,1}$-norm minimization problem by using some inexact APG algorithms, establish the iteration complexities, and present numerical results to demonstrate the efficiency of our proposed algorithms. In particular, as we will show later in the paper, our inexact APG algorithm can be much more efficient than the IADM-MFL algorithm (the proximal alternating direction method in [19]) for solving the problem (1).

The paper is organized as follows. In Section 2 we present the APG algorithm to solve (1) and elaborate on how to derive the closed-form solution of the inner subproblem generated at each iteration. In Section 3 we prove that the proposed APG algorithm enjoys the same iteration complexity as the FISTA algorithm in [4]. We also present inexact APG version with three different choices of the self-adjoint positive definite linear operator $\mathcal{H}^k$. In Section 4 we conduct some preliminary numerical experiments to evaluate the practical performance of our proposed inexact APG algorithms for solving matrix $l_{2,1}$-norm minimization problems arising from simulated data and real data set, then compare it with the existing IADM-MFL method. Finally, we have a conclusion section.

2. An accelerated proximal gradient method

Consider the standard form of the matrix $l_{2,1}$-norm minimization problem (1)

$$\min_{X\in\Re^{n\times t}} \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2 + \mu\|X\|_{2,1}, \tag{3}$$

where $\mathcal{A}:\Re^{n\times t}\to\Re^m$ is a map defined by matric-vector multiplication based on each task, i.e. $\mathcal{A}(X)=[A_1X_{:,1};\cdots;A_tX_{:,t}]\in\Re^m$. Let $F(X)=\frac{1}{2}\|\mathcal{A}(X)-b\|_2^2$, we consider the following minimization problem

$$\min_{X\in\Re^{n\times t}}\Phi(X):=F(X)+\mu\|X\|_{2,1}, \tag{4}$$

where $dom(\mu\|X\|_{2,1})=\Re^{n\times t}$, and the gradient ∇F is Lipschitz continuous with modulus L on $\Re^{n\times t}$, i.e.,

$$\|\nabla F(X)-F(Y)\|\le L\|X-Y\|,\ \forall X,Y\in\Re^{n\times t}.$$

We note that the problem (3) has an optimal solution since the function $\|\cdot\|_{2,1}$ is coercive. In what follows, $\mathcal{X}^*$ denotes the set of optimal solutions. The inexact APG algorithm is described for minimizing the matrix $l_{2,1}$-norm regularization problem (3) as follows.

Algorithm 1: APG for matrix $l_{2,1}$-norm minimization problem (3)

Step0. Given a tolerance $\varepsilon>0$. Input $Y^1=X^0$, $t_1=1$. Set $k=1$. Iterate the following steps.

Step1. Find an minimizer

$$X^k=\arg\min_{Y\in\Re^{n\times t}}\{F(Y^k)+\langle\nabla F(Y^k),Y-Y^k\rangle+\frac{1}{2}\langle Y-Y^k,\mathcal{H}^k(Y-Y^k)\rangle+\mu\|Y\|_{2,1}\}, \tag{5}$$

where $\mathcal{H}_k$ is a self-adjoint positive definite linear operator that is chosen by the user.

Step2. Update $t_{k+1}=\frac{1+\sqrt{1+4t_k^2}}{2}$.

Step3. Compute $Y^{k+1}=X^k+(\frac{t_k-1}{t_{k+1}})(X^k-X^{k-1})$.

Remark. We give the details on how to chose the proper self-adjoint positive definite linear operator $\mathcal{H}^k$ in Section 3, but $\mathcal{H}^k$ is restricted to a Lipschitz constant in FISTA [4].

Given any positive definite linear operator $\mathcal{H}^k:\Re^{n\times t}\to\Re^{n\times t}$, we define $Q^k(\cdot):\Re^{n\times t}\to\Re$ by

$$Q^k(X)=F(Y^k)+\langle\nabla F(Y^k),X-Y^k\rangle+\frac{1}{2}\langle X-Y^k,\mathcal{H}^k(X-Y^k)\rangle. \tag{6}$$

Note that if we choose $\mathcal{H}^k = LI$, then we have $F(X) \leq Q^k(X)$ for all $X \in \Re^{n\times t}$. Suppose for each k, we have a minimizer:

$$\begin{aligned} X^k &= \arg\min_{X\in\Re^{n\times t}} Q^k(X) + \mu\|X\|_{2,1} \\ &= \arg\min_{X\in\Re^{n\times t}} \langle \nabla F(Y^k), X - Y^k\rangle + \frac{1}{2}\langle X - Y^k, \mathcal{H}^k(X - Y^k)\rangle + \mu\|X\|_{2,1} \\ &= \arg\min_{X\in\Re^{n\times t}} \langle \nabla F(Y^k), X - Y^k\rangle + \frac{1}{2}\langle X - Y^k, \mathcal{H}^k(X - Y^k)\rangle + \mu\|X\|_{2,1} \\ &= \arg\min_{X\in\Re^{n\times t}} \frac{1}{2}\langle X - (Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k)), \mathcal{H}^k(X - (Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k)))\rangle \\ &\quad +\mu\|X\|_{2,1} \end{aligned} \tag{7}$$

To get the exact solution of (7) more specifically, we set $M = Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k)$. The solution of (7) takes the following form:

$$X^{k+1} = \arg\min_{X_{1,:},\cdots,X_{n,:}} \sum_{i=1}^{n}(\mu\|X_{i,:}\|_{2,1} + \frac{1}{2}\langle X_{i,:} - M_{i,:}, \mathcal{H}^k(X_{i,:} - M_{i,:})\rangle), \tag{8}$$

which shows that the problem (8) can be decomposed into n separate subproblems of dimension t, i.e.

$$\min_{X_{i,:}\in\Re^t} \mu\|X_{i,:}\|_{2,1} + \frac{1}{2}\langle X_{i,:} - M_{i,:}, \mathcal{H}^k(X_{i,:} - M_{i,:})\rangle, \quad i = 1, 2, \cdots, n. \tag{9}$$

It is easy to see that the optimal solution $X^*_{i,:}$ must be in the direction of $M_{i,:}$ and takes the form $X^*_{i,:} = aM_{i,:}$ with scalar $a \geq 0$. By constructing the Lagrangian dual form, the closed-form solutions of (8) can be obtained (see e.g., [5, 7, 8]) explicitly by

$$X^*_{i,:} = max\{M_{i,:} - \frac{\mu(\mathcal{H}^k)^{-1}M_{i,:}}{\|M_{i,:}\|_2}, 0\}, \quad i = 1, \cdots, n. \tag{10}$$

Then the closed-form solution of (5) is given by

$$X^k = \begin{bmatrix} max\{(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{1,:} - \frac{\mu(\mathcal{H}^k)^{-1}(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{1,:}}{\|(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{1,:}\|_2}, 0\} \\ max\{(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{2,:} - \frac{\mu(\mathcal{H}^k)^{-1}(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{2,:}}{\|(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{2,:}\|_2}, 0\} \\ \cdots \\ max\{(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{n,:} - \frac{\mu(\mathcal{H}^k)^{-1}(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{n,:}}{\|(Y^k - \frac{1}{\mathcal{H}^k}\nabla F(Y^k))_{n,:}\|_2}, 0\} \end{bmatrix}, \tag{11}$$

where $\nabla F(Y^k) = \mathcal{A}^*(\mathcal{A}(Y^k) - b)$ is the gradient of $F(Y^k)$.

Therefore, when the APG is applied to solving (3), the generated inner subproblem has closed-form solution. This feature makes the implementation of the inexact APG for (3) very easy if one choose $\mathcal{H}^k = LI$.

3. Analysis of inexact APG method for (3)

3.1. *Convergence analysis*

First we need the following assumptions which have been given in paper [6].
Assumption A. (i)Let $\{\xi_k\}$, $\{\epsilon_k\}$ be given convergent sequences of nonnegative numbers such that

$$\sum_{k=1}^{\infty}\xi_k<\infty,\quad \sum_{k=1}^{\infty}\epsilon_k<\infty.$$

(ii)the minimizer X^k in (7) satisfies the following conditions:

$$\Phi(X^k)\le Q^k(X^k)+\mu\|X^k\|_{2,1}+\frac{\xi_k}{2t_k^2}, \tag{12}$$

$$\nabla F(X^k)+\mathcal{H}^k(X^k-Y^k)+\gamma_k=\delta_k \ \text{ with } \|(\mathcal{H}^k)^{-1/2}\delta_k\|\le\epsilon_k/(\sqrt{2}t_k), \tag{13}$$

where $\gamma_k\in\partial(\mu\|X^k\|_{2,1};\frac{\xi_k}{2t_k^2})$ (the set of $\frac{\xi_k}{2t_k^2}$-subgradients of $\mu\|X\|_{2,1}$ at X^k).

The following lemma shows that the optimal solution set of (3) is bounded.

Lemma 1
For each $\mu>0$, the optimal solution set $\mathcal{X}^*$ of (3) is bounded, and for any $X^*\in\mathcal{X}$, we have

$$\|X\|_F\le\chi. \tag{14}$$

where

$$\chi=\begin{cases} min\{\|b\|_2^2/(2\mu),\|\mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}b\|_{2,1}\}, & \text{if } \mathcal{A} \text{ is surjective,}\\ \|b\|_2^2/(2\mu), & \text{otherwise.}\end{cases} \tag{15}$$

Proof
From the definition of Frobenius norm $\|X\|_F=(\sum_{i=1}^n\sum_{j=1}^t X_{i,j}^2)^{1/2}$ and $l_{2,1}$-norm $\|X\|_{2,1}=\sum_{i=1}^n\sqrt{\sum_{j=1}^t X_{i,j}^2}$, we obtain that

$$\|X\|_F\le\|X\|_{2,1}. \tag{16}$$

By considering the objective value of (3) at $X=0$, for any $X^*\in\mathcal{X}^*$, we have

$$\mu\|X^*\|_{2,1}\le\frac{1}{2}\|\mathcal{A}(X^*)-b\|_2^2+\mu\|X^*\|_{2,1}\le\frac{1}{2}\|b\|_2^2. \tag{17}$$

This together with the inequality (16) show that $\|X^*\|_F\le\|b\|_2^2/(2\mu)$. On the other hand, if $\mathcal{A}$ is surjective, by considering the objective value of (3) at $X=\mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}b$, for any $X^*\in\mathcal{X}^*$, we have

$$\mu\|X^*\|_{2,1}\le\frac{1}{2}\|\mathcal{A}(X^*)-b\|_2^2+\mu\|X^*\|_{2,1}\le\mu\|\mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}b\|_{2,1}. \tag{18}$$

This together with the inequality (16) implies (14). □

By the convexity of $\mu\|\cdot\|_{2,1}$, i.e. $\mu\|X\|_{2,1} \geq \mu\|X^k\|_{2,1} + \langle X - X^k, \gamma_k\rangle$, and Assumption A, similar to Lemma 2.1 and Lemma 2.2 in [6], the following two lemmas are described. For the proof is essentially the same as Lemma 2.3 and Lemma 4.1 in [4], we omit its proof here.

Lemma 2
Given $Y^k \in \Re^{n\times t}$ and a positive definite linear operator $\mathcal{H}^k$ on $\Re^{n\times t}$ such that the conditions (12) and (13) hold. Then for any $X \in \Re^{n\times t}$, we have

$$\begin{aligned}\Phi(X) - \Phi(X^k) \quad \geq \quad & \frac{1}{2}\langle X^k - Y^k, \mathcal{H}^k(X^k - Y^k)\rangle + \langle Y^k - X, \mathcal{H}^k(X^k - Y^k)\rangle \\ & +\langle \delta_k, X - X^k\rangle - \xi_k/t_k^2. \end{aligned} \tag{19}$$

Lemma 3
Suppose that $\mathcal{H}^k \succeq \mathcal{H}^{k+1} \succ 0$ for all k. Let

$$C_k = \Phi(X^k) - \Phi(X^*) \geq 0, \quad D_k = t_k X^k - (t_k - 1)X^{k-1} - X^*. \tag{20}$$

Then

$$\begin{aligned} t_k^2 C_k + \frac{1}{2}\langle D_k, \mathcal{H}^k(D_k)\rangle \quad \geq \quad & t_{k+1}^2 C_{k+1} + \frac{1}{2}\langle D_{k+1}, \mathcal{H}^{k+1}(D_{k+1})\rangle \\ & -t_{k+1}\langle \delta_{k+1}, D_{k+1}\rangle - \xi_{k+1}. \end{aligned} \tag{21}$$

We should note that although we share the similar idea in the forthcoming lemma in this paper with Lemma 2.3 in [6], but as it can be seen later, the proof is much more clear and concise due to different technical details.

Lemma 4
Suppose that $\mathcal{H}^k \succeq \mathcal{H}^{k+1} \succ 0$ for all k. Then

$$t_k^2 C_k \leq (\bar{\xi}_k + \sqrt{\tau + \epsilon_1\sqrt{\tau + \xi_1}})^2. \tag{22}$$

Proof
For simplicity, we define $A_k = t_k^2 C_k$, $B_k = \frac{1}{2}\langle D_k, \mathcal{H}^k(D_k)\rangle \geq 0$, $E_k = t_k\langle \delta_k, D_k\rangle$, $\tau = \frac{1}{2}\langle X^0 - X^*, \mathcal{H}^1(X^0 - X^*)\rangle$ and $\bar{\xi}_k = \sum_{k=1}^k(\sqrt{\xi_k} + \epsilon_k)$. Note that we have $A_1 = \Phi(X^1) - \Phi(X^*)$, $B_1 = \frac{1}{2}\langle X_1 - X^*, \mathcal{H}^k(X_1 - X^*)\rangle$ and $|E_k| \leq \|(\mathcal{H}^k)^{-1/2}\delta_k\|\|(\mathcal{H}^k)^{-1/2}D_k\|t_k \leq \epsilon_k\|(\mathcal{H}^k)^{-1/2}D_k\|/\sqrt{2} = \epsilon_k\sqrt{B_k}$.

First, we show that $A_1 + B_1 \leq \tau + \epsilon_1\sqrt{B_1} + \xi_1$. By the step0 in Algorithm 1 $Y^1 := X^0$ and applying the inequality (19) to $X := X^*$ with $k := 1$, we have that

$$\begin{aligned} -A_1 \quad \geq \quad & \frac{1}{2}\langle X^1 - Y^1, \mathcal{H}^1(X^1 - Y^1)\rangle + \langle Y^1 - X^*, \mathcal{H}^1(X^1 - Y^1)\rangle + \delta_1, X^* - X^1\rangle - \xi_1 \\ = \quad & \frac{1}{2}\langle X^1 - X^*, \mathcal{H}^1(X^1 - X^*)\rangle - \frac{1}{2}\langle Y^1 - X^*, \mathcal{H}^1(Y^1 - X^*)\rangle + \delta_1, X^* - X^1\rangle - \xi_1 \\ = \quad & B_1 - \frac{1}{2}\langle X^0 - X^*, \mathcal{H}^1(X^0 - X^*)\rangle + \delta_1, X^* - X^1\rangle - \xi_1. \end{aligned}$$

Together with $\|(\mathcal{H}^1)^{-1/2}\delta_1\| \le \epsilon_1/\sqrt{2}$, it shows that

$$A_1 + B_1 \le \frac{1}{2}\langle X^0 - X^*, \mathcal{H}^1(X^0 - X^*)\rangle - \delta_1, X^* - X^1\rangle + \xi_1 \le \tau + \epsilon_1\sqrt{B_1} + \xi_1, \tag{23}$$

and $B_1 \le \tau + \epsilon_1\sqrt{B_1} + \xi_1$. Solve the above inequality, we further obtain that

$$\sqrt{B_1} \le \frac{1}{2}(\epsilon_1 + \sqrt{\epsilon_1^2 + 4(\tau + \xi_1)}) \le \epsilon_1 + \sqrt{\tau + \xi_1}$$

. Let

$$s_k = \epsilon_1\sqrt{B_1} + \cdots + \epsilon_k\sqrt{B_k} + \xi_1 + \cdots + \xi_k,$$

then

$$s_1 = \epsilon_1\sqrt{B_1} + \xi_1 \le \epsilon_1(\epsilon_1 + \sqrt{\tau + \xi_1}) + \xi_1 \le \epsilon_1^2 + \xi_1 + \epsilon_1\sqrt{\tau + \xi_1}.$$

Hence

$$\sqrt{\tau + s_1} \le \sqrt{\tau + \epsilon_1^2 + \xi_1 + \epsilon_1\sqrt{\tau + \xi_1}} \le \epsilon_1 + \sqrt{\xi_1} + \sqrt{\tau + \epsilon_1\sqrt{\tau + \xi_1}}. \tag{24}$$

By Lemma 3 we have for every $k \ge 1$

$$A_k + B_k - s_k \le A_{k-1} + B_{k-1} - s_{k-1} \le \cdots \le A_1 + B_1 - s_1 \le \tau. \tag{25}$$

and hence the inequality $B_k \le \tau + s_k$ holds true, which combined with the definition of s_k yields

$$\tau + s_k = \tau + s_{k-1} + \epsilon_k\sqrt{B_k} + \xi_k \le \tau + s_{k-1} + \epsilon_k\sqrt{\tau + s_k} + \xi_k. \tag{26}$$

Then the inequality (26) can equivalently be written as

$$(\tau + s_k) - \epsilon_k\sqrt{\tau + s_k} - (\tau + s_{k-1} + \xi_k) \le 0.$$

So we have

$$\begin{aligned}\sqrt{\tau + s_k} &\le \frac{1}{2}(\epsilon_k + \sqrt{\epsilon_k^2 + 4(\tau + s_{k-1} + \xi_k)}) \\ &\le \epsilon_k + \sqrt{\tau + s_{k-1} + \xi_k} \\ &\le \epsilon_k + \sqrt{\tau + s_{k-1}} + \sqrt{\xi_k}.\end{aligned}$$

Further we obtain that

$$\begin{aligned}\sqrt{\tau + s_k} &\le \sum_{j=2}^{k}(\epsilon_j + \sqrt{\xi_j}) + \sqrt{\tau + s_1} \\ &\le \sum_{j=2}^{k}(\epsilon_j + \sqrt{\xi_j}) + \epsilon_1 + \sqrt{\xi_1} + \sqrt{\tau + \epsilon_1\sqrt{\tau + \xi_1}} \\ &= \bar{\xi}_k + \sqrt{\tau + \epsilon_1\sqrt{\tau + \xi_1}},\end{aligned}$$

where the last inequality follows Equation (24). Together with $A_k \leq \tau + s_k$ for all k by Equation (25), it shows that the required Equation (22) holds. □

The following theorem gives an upper bound on the number of iterations for the inexact APG algorithm to achieve ϵ-optimality.

Theorem 1
Suppose that Assumption A holds, and $\mathcal{H}^k \succeq \mathcal{H}^{k+1} \succ 0$ for all k. Then

$$0 \leq \Phi(X^k) - \Phi(X^*) \leq \frac{4}{(k+1)^2}(\bar{\xi}_k + \sqrt{\tau + \epsilon_1\sqrt{\tau + \xi_1}})^2. \tag{27}$$

Hence

$$\Phi(X^k) - \Phi(X^*) \leq \epsilon, \text{ whenever } k \geq 2(\bar{\xi}_k + \sqrt{\tau + \epsilon_1\sqrt{\tau + \xi_1}})/\sqrt{\epsilon} - 1. \tag{28}$$

Proof
By Theorem 2.1 in [6], we obtain (27). Based on the basic properties of inequality, we get the required result in (28). □

3.2. *Choices of* $\mathcal{H}^k$

In Section 2, an APG method (Algorithm 1) is presented for solving matrix $l_{2,1}$-norm minimization problem (3) with the desired convergent rate of $\mathcal{O}(1/k^2)$. However, an important issue on how to solve the inner subproblem (5) efficiently has not been addressed.

In this subsection, we propose three different implementable choices of the self-adjoint positive definite linear operator $\mathcal{H}^k$ which take advantage of more useful information.

(i)First we change our attention to consider the function $F(X)$ in problem (3),

$$F(X) = \frac{1}{2}\|\mathcal{A}(X) - b\|_2^2.$$

Clearly, $F(X)$ is a quadratic function, its gradient is $\nabla F(Y^k) = \mathcal{A}^*(\mathcal{A}(Y) - b)$, and its Hessian matrix is $G = \mathcal{A}^*\mathcal{A}$. Consider the following quadratic approximation of $\Phi(X) := F(X) + \mu\|X\|_{2,1}$ at a given point Y:

$$\Psi(X) := F(X) + \langle \nabla F(Y), X - Y\rangle + \frac{1}{2}\langle X - Y, G(X - Y)\rangle + \mu\|X\|_{2,1}. \tag{29}$$

Suppose we have the eigenvalue decomposition $G = P\Lambda P^T$, where $\Lambda = diag(\lambda)$ and $\lambda = (\lambda_1, \cdots, \lambda_n)^T$ is the vector of eigenvalues of G. Then the quadratic term

$$\begin{aligned}\langle X - Y, G(X - Y)\rangle &= \langle X - Y, P\Lambda P^T(X - Y)\rangle = \langle \hat{X} - \hat{Y}, \Lambda\hat{X} - \hat{Y}\rangle \\ &= \sum_{i=1}^{n}\sum_{j=1}^{t}(\hat{X} - \hat{Y})_{i,j}\lambda_j,\end{aligned}$$

where $\hat{X} - \hat{Y} = P^T(X - Y)P$. For the choice of $\mathcal{H}^k$, one may simply choose $\mathcal{H}^k = \lambda_{max}(\mathcal{A}^*\mathcal{A})I$ to approximate the Hessian matrix. So the minimizer of the inner subproblem (5) is

$$X^k_{i,:} = (1 - \frac{\mu}{\lambda_{max}\|(Y^k - \frac{1}{\lambda_{max}}\nabla F(Y^k))_{i,:}\|_2})_+(Y^k - \frac{1}{\lambda_{max}}\nabla F(Y^k))_{i,:}, \quad (30)$$

where $(\cdot)_+ = \max(\cdot, 0)$ and $i = 1, \cdots, n$. We are now ready to state the steps of the inexact version APG method with eigenvalue of the Hessian matrix (IAPG-EIG) as follows.

Algorithm 2: IAPG-EIG for matrix $l_{2,1}$-norm minimization problem (3)

Step0. Given a tolerance $\varepsilon > 0$. Input $Y^1 = X^0$, $t_1 = 1$. Set $k = 1$. Iterate the following steps.

Step1. Solve X^k via (30).

Step2. Update $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.

Step3. Compute $Y^{k+1} = X^k + (\frac{t_k - 1}{t_{k+1}})(X^k - X^{k-1})$.

However, if the Hessian matrix $\mathcal{A}^*\mathcal{A}$ is ill-conditioned, this choice of $\mathcal{H}^k = \lambda_{max}I$ may not work very well in practice since $\mathcal{H}^k$ may not be a good approximation of the Hessian matrix of $F(X)$. To find a better approximation of the Hessian matrix, we propose the following technique.

(ii)The spectral gradient method (also named the two-point stepsize method) was initially given by Barzilai and Borwein [3] for solving strict quadratic minimization problems. This method consists essentially of a steepest descent method, where the choice of the stepsize along the negative gradient direction is potentially derived from a two-point approximation to a secant equation underlying the quasi-Newton method. Raydan [14] showed that the BB method is globally convergent in the strictly convex quadratic case. Raydan [15] extended the BB method for solving general unconstrained optimization problems and Yuan & Wei [20] extended the BB method for nonsmooth convex optimization problems.

Let $a_{k-1} = Y^k - Y^{k-1}$, $b_{k-1} = \nabla F(Y^k) - \nabla F(Y^{k-1})$, then the Barzilai and Borwein parameter is defined as $\alpha_k = \frac{\langle a_{k-1}, b_{k-1}\rangle}{\langle a_{k-1}, a_{k-1}\rangle}$. One may simply choose $\mathcal{H}^k = \alpha_k I$, so the minimizer of the inner subproblem (5) is

$$X^k_{i,:} = (1 - \frac{\mu}{\alpha_k\|(Y^k - \frac{1}{\alpha_k}\nabla F(Y^k))_{i,:}\|_2})_+(Y^k - \frac{1}{\alpha_k}\nabla F(Y^k))_{i,:}, \quad (31)$$

where $i = 1, \cdots, n$. We are now ready to state the steps of the inexact version APG method with Barzilai and Borwein parameter (IAPG-BB) as follows.

Algorithm 3: IAPG-BB for matrix $l_{2,1}$-norm minimization problem (3)

Step0. Given a tolerance $\varepsilon > 0$. Input $Y^1 = X^0$, $t_1 = 1$. Set $k = 1$. Iterate the following steps.

Step1. Solve X^k via (31).

Step2. Update $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.

Step3. Compute $Y^{k+1} = X^k + (\frac{t_k-1}{t_{k+1}})(X^k - X^{k-1})$.

(iii) Note that we can always choose $\mathcal{H}^k = LI$ if there are no other better choices, where L is the Lipschitz constant. We denote this algorithm as IAPG-L, the steps follow similar arguments as in Algorithm 3 and we omit it here.

4. Numerical results

In this section we report our numerical experiments conducted for the matrix $l_{2,1}$-norm minimization problem (3) in MATLAB R2010a running on a PC Intel Pentium CPU at 2.8 GHz and 1 GB of memory. We perform two types of experiments concentrating on the simulated data and real data to show the performance of the proposed algorithms IAPG-L, IAPG-EIG and IAPG-BB. In each test, we compare our algorithm to the IADM-MFL algorithm [19], which has been verified that it outperforms the SLEP algorithm. For convenience, we use the IADM-MFL Matlab package provided by Y. Xiao to do comparison. For each test, we start at zeros points and stop these algorithms when the relative change between successive iterations falls below a small number tol, i.e.

$$\text{RelChg} = \frac{\|X_k - X_{k-1}\|_F}{\|X_{k-1}\|_F} \leq tol. \tag{32}$$

Those algorithms is also forced to stop when the number of iterations exceeds 1000.

4.1. Simulated data

As [1], we create synthetic data sets by generating task parameters $\bar{X}_{:,j}$ from a 5-dimensional Gaussian distribution with zero mean and covariance equal to diag$\{1, 0.64, 0.49, 0.36, 0.25\}$. To these 5-dimensional $\bar{X}_{:,j}$, 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 $randn(m_j, n)$. The outputs b_j are computed from the A_j and $\bar{X}_{:,j}$ as

$$b_j = A_j \bar{X}_{:,j} + \omega,$$

where ω is zero-mean Gaussian noise with standard deviation equal to 1.e-2. Let X^* be the 'optimal' solution produced by algorithm, we use the relative error to

measure the quality of X^* to original $\bar{X}$, i.e.

$$\text{RelErr} = \frac{\|\mathrm{X}^* - \bar{\mathrm{X}}\|_{\mathrm{F}}}{\|\bar{\mathrm{X}}\|_{\mathrm{F}}}. \tag{33}$$

First, we take $\mu = 1e-2$, $t = 200$, $n = 15$, $tol = 1e-3$, and $m_j = 100$ for all $j = 1, 2, \cdots, t$ in those algorithms, and examine the objective function values and the testing error rate behavior when each algorithm is proceeding. The convergence behavior of both algorithms are reported in Figure 4.1. To illustrate the convergence behavior of those solvers, we draw four figures to show the decreasing function values as the iteration and CPU time increase. It is clear that each algorithm generates decreasing sequences and eventually attains nearly equal function values in the end. The preliminary numerical comparisons indicate that those algorithms are efficient, but the CPU time plots show that IAPG-BB is the fastest and the proposed algorithms IAPG-L, IAPG-EIG, IAPG-BB are better than IADM-MFL.

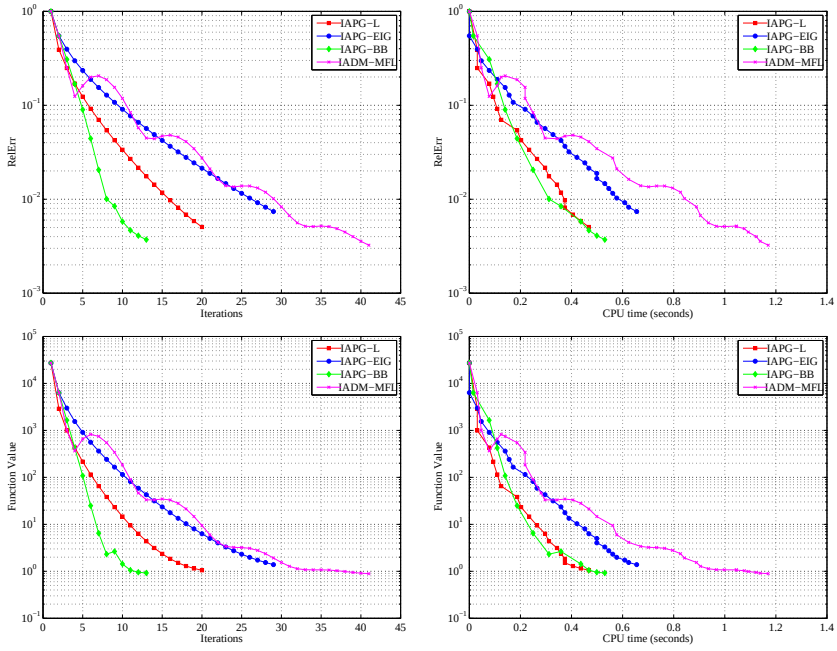

Figure 4.1. Convergence performance of IAPG-L, IAPG-EIG, IAPG-BB and IADM-MFL on simulated data. First row: Relative error; Second row: objective function values; The x-axes represents the number of iterations in first column and CPU time in second column.

Second, the number of tasks and dimensions may also affect the performance of the method, so we report the numerical results of IAPG-L, IAPG-EIG, IAPG-BB

and IADM-MFL with different number of tasks and dimensions. The columns of Tables 4.1 and 4.2 have the following meanings, m: the dimension of the outputs; n: the dimension of the test data; t: the total number of tasks; ITER: the total number of iterations; TIME: the CPU time in second; RelErr: the relative error of solution; RelChg: the relative change of solution when the program stops.

As we can see from Table 4.1 that all of those algorithms terminate successfully at a solution of the problem and all algorithms have good performance for all tests. The proposed algorithms IAPG-L, IAPG-EIG, IAPG-BB are better than IADM-MFL, and IAPG-BB is more competitive than other algorithms as IAPG-BB algorithm can get the solution of all the test data at a smaller number of iterations and smaller CPU time.

Table 4.1. Numerical Results for Random Problems

b,X	IAPG-L	IAPG-EIG	IAPG-BB	IADM-MFL
(m, n, t)	ITER/TIME/RelErr	ITER/TIME/RelErr	ITER/TIME/RelErr	ITER/TIME/RelErr
(5000,5, 50)	13/0.0936/2.07e-003	18/0.0468/4.14e-003	11/0.0780/1.45e-003	23/0.1092/3.56e-003
(10000,5,100)	11/0.0780/2.17e-003	17/0.1248/3.59e-003	10/0.0936/1.45e-003	24/0.1404/1.63e-003
(15000,5,150)	12/0.1404/2.11e-003	18/0.2184/3.50e-003	10/0.1404/1.64e-003	24/0.3588/1.82e-003
(20000,5,200)	11/0.1716/2.25e-003	17/0.2184/3.65e-003	10/0.4836/1.52e-003	24/0.4056/1.68e-003
(25000,5,250)	11/0.2184/2.06e-003	17/0.3900/3.34e-003	10/0.4992/1.51e-003	24/0.6084/1.58e-003
(30000,5,300)	13/0.2964/2.25e-003	19/0.7488/4.06e-003	11/0.6084/1.52e-003	26/0.8268/1.78e-003
(5000,10,50)	17/0.0936/3.73e-003	25/0.1092/5.63e-003	12/0.0624/2.50e-003	33/0.1560/3.34e-003
(10000,10,100)	17/0.0780/3.60e-003	24/0.2496/6.09e-003	12/0.1716/2.60e-003	32/0.3120/3.36e-003
(15000,10,150)	16/0.2808/3.80e-003	24/0.3744/5.53e-003	12/0.2652/2.37e-003	32/0.5304/3.11e-003
(20000,10,200)	16/0.3432/3.13e-003	23/0.4212/5.26e-003	11/0.4524/2.37e-003	31/0.5928/2.97e-003
(25000,10,250)	17/0.4680/3.94e-003	25/0.6864/5.99e-003	12/0.5148/2.54e-003	33/0.8892/3.59e-003
(30000,10,300)	16/0.6396/3.64e-003	24/0.7800/5.32e-003	11/0.7488/2.60e-003	31/1.1232/3.25e-003
(5000,15,50)	21/0.0936/4.82e-003	30/0.1716/7.52e-003	13/0.0936/3.47e-003	43/0.2496/2.83e-003
(10000,15,100)	21/0.1872/4.95e-003	30/0.3276/7.63e-003	13/0.2496/3.75e-003	42/0.4836/3.49e-003
(15000,15,150)	21/0.3276/5.71e-003	30/0.5148/8.58e-003	13/0.4212/4.48e-003	42/0.7020/4.08e-003
(20000,15,200)	20/0.4524/5.06e-003	29/0.7020/7.40e-003	13/0.4992/3.71e-003	41/1.0296/3.24e-003
(25000,15,250)	22/0.7488/5.15e-003	31/0.9204/8.16e-003	13/0.7488/4.15e-003	45/1.4040/3.27e-003
(30000,15,300)	21/0.8580/5.71e-003	31/1.2324/7.89e-003	13/1.1076/4.15e-003	44/1.7472/3.47e-003
(5000,20,50)	26/0.1560/7.24e-003	36/0.1716/1.12e-002	21/0.2184/2.99e-003	54/0.3432/4.27e-003
(10000,20,100)	26/0.3588/6.66e-003	36/0.3432/1.06e-002	14/0.3588/6.13e-003	51/0.6552/4.55e-003
(15000,20,150)	27/0.4680/7.14e-003	38/0.6396/1.07e-002	17/0.6240/3.05e-003	53/0.9984/4.91e-003
(20000,20,200)	25/0.5772/7.42e-003	36/0.9048/1.03e-002	17/0.8580/3.11e-003	52/1.5132/4.27e-003
(25000,20,250)	26/1.1076/6.56e-003	36/1.3728/1.03e-002	14/0.9828/6.20e-003	52/1.7784/4.33e-003
(30000,20,300)	26/1.1232/7.66e-003	37/1.8720/1.11e-002	20/1.9500/3.00e-003	54/2.4804/4.35e-003
(5000,25,50)	31/0.1716/9.03e-003	42/0.2808/1.42e-002	21/0.2184/3.36e-003	61/0.4992/5.91e-003
(10000,25,100)	32/0.3432/9.64e-003	44/0.6240/1.45e-002	21/0.5460/3.50e-003	66/0.8892/4.94e-003
(15000,25,150)	33/0.5460/9.51e-003	45/0.8736/1.46e-002	21/0.8736/3.47e-003	69/1.3260/4.57e-003
(20000,25,200)	30/0.7020/9.84e-003	42/1.1700/1.40e-002	21/1.2480/3.58e-003	64/1.7472/4.94e-003
(25000,25,250)	32/1.0764/1.01e-002	44/1.6536/1.51e-002	21/1.8096/3.50e-003	67/2.2620/5.18e-003
(30000,25,300)	31/1.2792/9.90e-003	43/2.1684/1.46e-002	21/2.1996/3.55e-003	65/2.8392/5.11e-003

4.2. Real data

In this subsection, we evaluate the proposed method IAPG with the Lipschitz constant and compare it with IADM-MFL by using the real data set. $dmoz$ is a text categorization data set available at http://www.dmoz.org/, in which each of the 10 tasks corresponds to one of the subcategories of the Arts category. We randomly sample 10%, 15%, 25%, 50% and 75% of this data set from each task for training and run both algorithms simultaneously to learn the joint feature among the task. The numerical results are listed in Table 4.2. When running both codes, we set all the parameter values as the previous subsection except for $\mu = 1e-4$ and $\beta = 0.01/mean(|y|)$. To illustrate the convergence behavior of both solvers,

we draw two figures to show the decreasing function values in the first 200 steps as the iteration increases. As can be seen from Figure 4.2, each algorithm generates decreasing sequences and eventually attains nearly equal function values in the end. From Table 4.2, we can get the conclusion that IAPG-L works better on these problems.

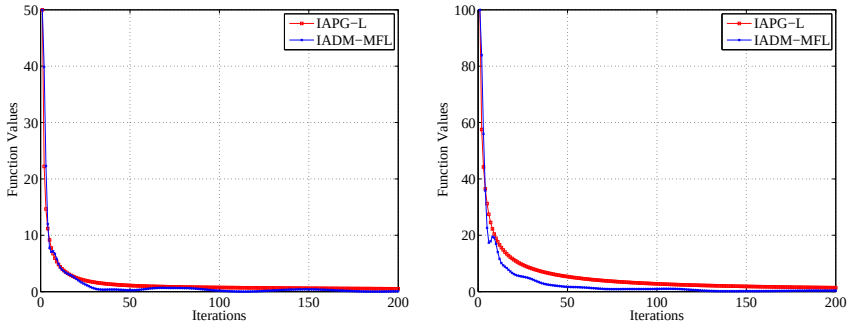

Figure 4.2. The convergence of IAPG-L and IADM-MFL on the *demo* data set when 25% (left figure) and 50% (right figure) of the data are used for training.

Table 4.2. Numerical Results for Demo Data

Demo	b,X	IAPG-L			IADM-MFL		
	(m, n, t)	ITER	TIME	RelChg	ITER	TIME	RelChg
10%	(40,500,10)	191	0.84	9.94e-004	427	2.23	9.99e-004
15%	(60,500,10)	179	0.78	9.93e-004	381	2.26	1.00e-003
25%	(100,500,10)	341	3.53	9.98e-004	524	5.27	9.96e-004
50%	(200,500,10)	292	1.76	9.99e-004	485	3.10	9.58e-004
75%	(300,500,10)	365	3.78	9.99e-004	440	3.98	9.96e-004

5. Conclusion

In this paper, we extend the APG method to solve matrix $l_{2,1}$-norm minimization problem in multi-task feature learning. We investigate the performance of our proposed algorithm in which the resulting inner subproblem has closed-form solution. And it can be easily determined by taking the problem's favorable structures. We also present inexact APG framework with three different choices of the self-adjoint positive definite linear operator $\mathcal{H}^k$: (1) IAPG-EIG use the eigenvalue of Hessian matrix, $\mathcal{H}^k = \lambda_{max}(A^*A)I$; (2) IAPG-BB use the Barzilai and Borwein parameter, $\mathcal{H}^k = \alpha_k I$; (3) IAPG-L use the Lipschitz continuous, $\mathcal{H}^k = LI$. We design efficient implementations of the algorithm and give comprehensive convergence results. The numerical experiments illustrate that the proposed algorithm is very promising and competitive, and it also provides a new approach to solve the joint feather selection problem in multi-task learning.

Acknowledgements

This work is supported by Guangxi NSF (Grant No. 2012GXNSFAA053002) and China NSF (Grant No. 11161003 and 11261006). The authors would appreciate the great work of the referees for their valuable comments and suggestions which lead to significant improvements on the presentation.

REFERENCES

1. A. Argyriou, T. Evgeniou and M. Pontil, *Convex multi-convex feature learning*, Machine Learning, 73(2008), 243-272.
2. D.P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, New York: Academic Press, 1982.
3. J. Barzilai and J.M. Borwein, *Two point step size gradient method*, IMA Journal of Numerical Analysis, 8(1988), 141-148.
4. A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sciences, 2(2009), 183-202.
5. J. Duchi and Y. Singer, *Efficient online and batch learning using forward backward splitting*, Journal of Machine Learning Research, 10(2009), 2899-2934.
6. K. Jiang, D. Sun and K.C. Toh, *An Inexact Accelerated Proximal Gradient Method for Large Scale Linearly Constrained Convex SDP*, SIAM Journal on Optimization, 22(2012), 1042-1064.
7. M. Kowalski, *Sparse regression using mixednorms*, Applied and Computational Harmonic Analysis, 27(2009), 303-324.
8. M. Kowalski, M. Szafranski and L. Ralaivola, *Multiple Indefinite Kernel Learning with Mixed Normregularization*, Proceedings of the 26th Annual International Conference on Machine Learning, 2009.
9. J. Liu, S. Ji and J. Ye, *Multi-Task Feather Learning Via Efficient $l_{2,1}$-norm Minimization*, in Conference on Uncertainty in Artificial Intelligence, 2009.
10. Y. Nesterov, *A method of solving a convex programming problem with convergence rate $O(1/k^2)$*, Soviet Mathematics Doklady, 27(1983), 372-376.
11. Y. Nesterov, *Gradient Methods for Minimizing Composite Objective Function*, CORE report, 2007.
12. F. Nie, H. Huang, X. Cai and C. Ding, *Efficient and Robust Feature Selection via Joint $l_{2,1}$-Norms minimization*, Neural Information Processing Systems Foundation, 2010.
13. G. Obozinski, B. Taskar and M. I. Jordan, *Multi-Task Feature Selection*, Technical Report, UC Berkeley, 2006.
14. M. Raydan, *On the Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*, IMA Journal of Numerical Analysis, 13(1993), 321-326.
15. M. Raydan, *The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*, SIAM Journal on Optimization, 7(1997), 26-33.
16. Y. Saeys, I. Inza and P. Larranaga, *A review of feature selection techniques in bioinformatics*, Bioinformatics, 23(2007), 2507-2517.
17. Z. Shen, K.C. Toh, and S. Yun, *An Accelerated Proximal Gradient Algorithm for Frame-Based Image Restoration via the Balanced Approach*, SIAM Journal on Imaging Sciences, 4(2011), 573-596.
18. K.C. Toh and S.W. Yun, *An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems*, Pacific Journal of Optimization, 6(2010), 615-640.
19. Y. Xiao, S. Wu and B. He, *A proximal alternating direction method for $l_{2,1}$-norm least squares problem in multi-task feature learning*, Journal of Industrial and Management Optimization, 8(2012), 1057-1069.

20. G. Yuan and Z. Wei, *The Barzilai and Borwein Gradient Method with Nonmonotone Line Search for Nonsmooth Convex Optimization Problems*, Mathematical Modelling and Analysis, 17(2012), 203-216.