(will be inserted by the editor)

Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning

Zhouchen Lin · Risheng Liu · Huan Li

Received: date / Accepted: date

Abstract Many problems in machine learning and other fields can be (re)formulated as linearly constrained separable convex programs. In most of the cases, there are multiple blocks of variables. However, the traditional alternating direction method (ADM) and its linearized version (LADM, obtained by linearizing the quadratic penalty term) are for the two-block case and cannot be naively generalized to solve the multi-block case. So there is great demand on extending the ADM based methods for the multi-block case. In this paper, we propose LADM with parallel splitting and adaptive penalty (LADMPSAP) to solve multi-block separable convex programs efficiently. When all the component objective functions have bounded subgradients, we obtain convergence results that are stronger than those of ADM and LADM, e.g., allowing the penalty parameter to be *unbounded* and proving the *sufficient and necessary conditions* for global convergence. We further propose a simple optimality measure and reveal the convergence rate of LADMPSAP in an ergodic sense. For programs with extra convex set constraints, with refined parameter estimation we devise a practical version of LADMPSAP for faster convergence. Finally, we generalize LADMPSAP to handle programs with more difficult objective functions by linearizing part of the objective function as well. LADMPSAP is particularly suitable for sparse representation and low-rank recovery problems

This paper is an extension of our prior work Lin et al (2011) and Liu et al (2013, oral presentation).

Z. Lin
Key Lab. of Machine Perception (MOE), School of EECS, Peking University. E-mail: zlin@pku.edu.cn

R. Liu (Corresponding author)
Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology,
School of Software Technology, Dalian University of Technology. E-mail: rsliu@dlut.edu.cn

H. Li
School of Software and Microelectronics, Peking University. E-mail: lihuan_ss@pku.edu.cn

because its subproblems have closed form solutions and the sparsity and low-rankness of the iterates can be preserved during the iteration. It is also highly parallelizable and hence fits for parallel or distributed computing. Numerical experiments testify to the advantages of LADMPSAP in speed and numerical accuracy.

Keywords Convex Programs · Alternating Direction Method · Linearized Alternating Direction Method · Proximal Alternating Direction Method · Parallel Splitting · Adaptive Penalty

1 Introduction

In recent years, convex programs have become increasingly popular for solving a wide range of problems in machine learning and other fields, ranging from theoretical modeling, e.g., latent variable graphical model selection (Chandrasekaran et al, 2012), low-rank feature extraction (e.g., matrix decomposition (Candès et al, 2011) and matrix completion (Candès and Recht, 2009)), subspace clustering (Liu et al, 2012), and kernel discriminant analysis (Ye et al, 2008), to real-world applications, e.g., face recognition (Wright et al, 2009), saliency detection (Shen and Wu, 2012), and video denoising (Ji et al, 2010). Most of the problems can be (re)formulated as the following linearly constrained separable convex program[1]:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} f_i(\mathbf{x}_i), \quad s.t. \quad \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b}, \tag{1}$$

where $\mathbf{x}_i$ and $\mathbf{b}$ could be either vectors or matrices[2], f_i is a closed proper convex function, and $\mathcal{A}_i : \mathbb{R}^{d_i} \to \mathbb{R}^m$ is a linear mapping. Without loss of generality, we may assume that none of the $\mathcal{A}_i$'s is a zero mapping, the solution to $\sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b}$ is non-unique, and the mapping $\mathcal{A}(\mathbf{x}_1, \cdots, \mathbf{x}_n) \equiv \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i)$ is onto[3].

1.1 Exemplar Problems in Machine Learning

In this subsection, we present some examples of machine learning problems that can be formulated as the model problem (1).

[1] If the objective function is not separable or there are extra convex set constraints, $\mathbf{x}_i \in X_i$, $i = 1, \cdots, n$, where X_i's are convex sets, the program can be transformed into (1) by introducing auxiliary variables, c.f. (26)-(28).

[2] In this paper we call each $\mathbf{x}_i$ a "block" of variables because it may consist of multiple scalar variables. We will use bold capital letters if a block is known to be a matrix.

[3] The last two assumptions are equivalent to that the matrix $\mathbf{A} \equiv (\mathbf{A}_1 \ \cdots \ \mathbf{A}_n)$ is not full column rank but full row rank, where $\mathbf{A}_i$ is the matrix representation of $\mathcal{A}_i$.

1.1.1 Latent Low-Rank Representation

Low-Rank Representation (LRR) (Liu et al, 2010, 2012) is a recently proposed technique for robust subspace clustering and has been applied to many machine learning and computer vision problems. However, LRR works well only when the number of samples is more than the dimension of the samples, which may not be satisfied when the data dimension is high. So Liu et al. (Liu and Yan, 2011) proposed latent LRR to overcome this difficulty. The mathematical model of latent LRR is as follows:

$$\min_{\mathbf{Z},\mathbf{L},\mathbf{E}} \|\mathbf{Z}\|_* + \|\mathbf{L}\|_* + \mu\|\mathbf{E}\|_1, \quad s.t. \quad \mathbf{X} = \mathbf{XZ} + \mathbf{LX} + \mathbf{E}, \tag{2}$$

where $\mathbf{X}$ is the data matrix, each column being a sample vector, $\|\cdot\|_*$ is the nuclear norm (Fazel, 2002), i.e., the sum of singular values, and $\|\cdot\|_1$ is the ℓ_1 norm (Candès et al, 2011), i.e., the sum of absolute values of all entries. Latent LRR is to decompose data into principal feature $\mathbf{XZ}$ and salient feature $\mathbf{LX}$, up to sparse noise $\mathbf{E}$.

1.1.2 Nonnegative Matrix Completion

Nonnegative matrix completion (NMC) (Xu et al, 2011) is a novel technique for dimensionality reduction, text mining, collaborative filtering, and clustering, etc. It can be formulated as:

$$\min_{\mathbf{X},\mathbf{e}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{e}\|^2, \quad s.t. \quad \mathbf{b} = \mathcal{P}_\Omega(\mathbf{X}) + \mathbf{e},\ \mathbf{X} \geq 0, \tag{3}$$

where $\mathbf{b}$ is the observed data in the matrix $\mathbf{X}$ contaminated by noise $\mathbf{e}$, Ω is an index set, $\mathcal{P}_\Omega$ is a linear mapping that selects those elements whose indices are in Ω, and $\|\cdot\|$ is the Frobenius norm. NMC is to recover the nonnegative low-rank matrix $\mathbf{X}$ from the observed noisy data $\mathbf{b}$.

To see that the NMC problem can be reformulated as (1), we introduce an auxiliary variable $\mathbf{Y}$ and rewrite (3) as

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{e}} \|\mathbf{X}\|_* + \chi_{\geq 0}(\mathbf{Y}) + \frac{1}{2\mu}\|\mathbf{e}\|^2, \quad s.t. \quad \begin{pmatrix} \mathcal{P}_\Omega(\mathbf{X}) \\ \mathbf{X} \end{pmatrix} - \begin{pmatrix} \mathbf{0} \\ \mathbf{Y} \end{pmatrix} + \begin{pmatrix} \mathbf{e} \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{0} \end{pmatrix}, \tag{4}$$

where $\chi_{\geq 0}(\mathbf{Y}) = \begin{cases} 0, & \text{if } \mathbf{Y} \geq 0, \\ +\infty, & \text{otherwise,} \end{cases}$ is the characteristic function of the set of nonegative matrices.

1.1.3 Group Sparse Logistic Regression with Overlap

Besides unsupervised learning models shown above, many supervised machine learning problems can also be written in the form of (1). For example, using

logistic function as the loss function in the group LASSO with overlap (Jacob et al, 2009; Deng et al, 2011), one obtains the following model:

$$\min_{\mathbf{w},b} \frac{1}{s}\sum_{i=1}^{s} \log\left(1+\exp\left(-y_i(\mathbf{w}^T\mathbf{x}_i+b)\right)\right)+\mu\sum_{j=1}^{t}\|\mathbf{S}_j\mathbf{w}\|, \tag{5}$$

where $\mathbf{x}_i$ and y_i, $i=1,\cdots,s$, are the training data and labels, respectively, and $\mathbf{w}$ and b parameterize the linear classifier. $\mathbf{S}_j$, $j=1,\cdots,t$, are the selection matrices, with only one 1 at each row and the rest entries are all zeros. The groups of entries, $\mathbf{S}_j\mathbf{w}$, $j=1,\cdots,t$, may overlap each other. This model can also be considered as an extension of the group sparse logistic regression problem (Meier et al, 2008) to the case of overlapped groups.

Introducing $\bar{\mathbf{w}}=(\mathbf{w}^T,b)^T$, $\bar{\mathbf{x}}_i=(\mathbf{x}_i^T,1)^T$, $\mathbf{z}=(\mathbf{z}_1^T,\mathbf{z}_2^T,\cdots,\mathbf{z}_t^T)^T$, and $\bar{\mathbf{S}}=(\mathbf{S},\mathbf{0})$, where $\mathbf{S}=(\mathbf{S}_1^T,\cdots,\mathbf{S}_t^T)^T$, (5) can be rewritten as

$$\min_{\bar{\mathbf{w}},\mathbf{z}} \frac{1}{s}\sum_{i=1}^{s} \log\left(1+\exp\left(-y_i(\bar{\mathbf{w}}^T\bar{\mathbf{x}}_i)\right)\right)+\mu\sum_{j=1}^{t}\|\mathbf{z}_j\|, \quad s.t. \quad \mathbf{z}=\bar{\mathbf{S}}\bar{\mathbf{w}}, \tag{6}$$

which is a special case of (1).

1.2 Related Work

Although general theories on convex programs are fairly complete nowadays, e.g., most of them can be solved by the interior point method (Boyd and Vandenberghe, 2004), when faced with large scale problems, which are typical in machine learning, the general theory may not lead to efficient algorithms. For example, when using CVX[4], an interior point based toolbox, to solve nuclear norm minimization problems (i.e., one of the f_i's is the nuclear norm of a matrix, e.g., (2) and (3)), such as matrix completion (Candès and Recht, 2009), robust principal component analysis (Candès et al, 2011), and low-rank representation (Liu et al, 2010, 2012), the complexity of each iteration is $O(q^6)$, where $q\times q$ is the matrix size. Such a complexity is unbearable for large scale computing.

To address the scalability issue, first order methods are often preferred. The accelerated proximal gradient (APG) algorithm (Beck and Teboulle, 2009; Toh and Yun, 2010) is popular due to its guaranteed $O(K^{-2})$ convergence rate, where K is the iteration number. However, APG is basically for unconstrained optimization. For constrained optimization, the constraints have to be added to the objective function as penalties, resulting in approximated solutions only. The alternating direction method (ADM)[5] (Fortin and Glowinski, 1983; Boyd et al, 2011; Lin et al, 2009a) has regained a lot of attention recently and is also widely used. It is especially suitable for separable convex programs like

[4] Available at `http://stanford.edu/~boyd/cvx`

[5] Also called the alternating direction method of multipliers (ADMM) in some literatures, e.g., (Boyd et al, 2011; Zhang et al, 2011; Deng and Yin, 2012).

(1) because it fully utilizes the separable structure of the objective function. Unlike APG, ADM can solve (1) exactly. Another first order method is the split Bregman method (Goldstein and Osher, 2008; Zhang et al, 2011), which is closely related to ADM (Esser, 2009) and is influential in image processing.

An important reason that first order methods are popular for solving large scale convex programs in machine learning is that the convex functions f_i's are often matrix or vector norms or characteristic functions of convex sets, which enables the following subproblems (called the proximal operation of f_i (Rockafellar, 1970))

$$\text{prox}_{f_i,\sigma}(\mathbf{w}) = \underset{\mathbf{x}_i}{\text{argmin}}\, f_i(\mathbf{x}_i) + \frac{\sigma}{2}\|\mathbf{x}_i - \mathbf{w}\|^2 \tag{7}$$

to have closed form solutions. For example, when f_i is the ℓ_1 norm, $\text{prox}_{f_i,\sigma}(\mathbf{w}) = \mathcal{T}_{\sigma^{-1}}(\mathbf{w})$, where $\mathcal{T}_\varepsilon(x) = \text{sgn}(x)\max(|x| - \varepsilon, 0)$ is the soft-thresholding operator (Goldstein and Osher, 2008); when f_i is the nuclear norm, the optimal solution is: $\text{prox}_{f_i,\sigma}(\mathbf{W}) = \mathbf{U}\mathcal{T}_{\sigma^{-1}}(\mathbf{\Sigma})\mathbf{V}^T$, where $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the singular value decomposition (SVD) of $\mathbf{W}$ (Cai et al, 2010); and when f_i is the characteristic function of the nonnegative cone, the optimal solution is $\text{prox}_{f_i,\sigma}(\mathbf{w}) = \max(\mathbf{w}, 0)$. Since subproblems like (7) have to be solved in each iteration when using first order methods to solve separable convex programs, that they have closed form solutions greatly facilitates the optimization.

However, when applying ADM to solve (1) with non-unitary linear mappings (i.e., $\mathcal{A}_i^\dagger\mathcal{A}_i$ is not the identity mapping, where $\mathcal{A}_i^\dagger$ is the adjoint operator of $\mathcal{A}_i$), the resulting subproblems may not have closed form solutions[6], hence need to be solved iteratively, making the optimization process awkward. Some work (Yang and Yuan, 2013; Lin et al, 2011) has considered this issue by linearizing the quadratic term $\|\mathcal{A}_i(\mathbf{x}_i) - \mathbf{w}\|^2$ in the subproblems, hence such a variant of ADM is called the linearized ADM (LADM). Deng and Yin (2012) further propose the generalized ADM that makes both ADM and LADM as its special cases and prove its globally linear convergence by imposing strong convexity on the objective function or full-rankness on some linear operators.

Nonetheless, most of the existing theories on ADM and LADM are for the *two-block* case, i.e., $n = 2$ in (1) (Fortin and Glowinski, 1983; Boyd et al, 2011; Lin et al, 2011; Deng and Yin, 2012). The number of blocks is restricted to two because the proofs of convergence for the two-block case are not applicable for the multi-block case, i.e., $n > 2$ in (1). Actually, a naive generalization of ADM or LADM to the multi-block case may diverge (see (15) and (Chen et al, 2013)). Unfortunately, in practice multi-block convex programs often occur, e.g., robust principal component analysis with dense noise (Candès et al, 2011), latent low-rank representation (Liu and Yan, 2011) (see (2)), and when there are extra convex set constraints (see (3) and (26)-(27)). So it is desirable to design practical algorithms for the multi-block case.

Recently He and Yuan (2013) and Tao (2014) considered the multi-block LADM and ADM, respectively. To safeguard convergence, He and Yuan (2013)

[6] Because $\|\mathbf{x}_i - \mathbf{w}\|^2$ in (7) becomes $\|\mathcal{A}_i(\mathbf{x}_i) - \mathbf{w}\|^2$, which cannot be reduced to $\|\mathbf{x}_i - \tilde{\mathbf{w}}\|^2$.

proposed LADM with Gaussian back substitution (LADMGB), which destroys the sparsity or low-rankness of the iterates during iterations when dealing with sparse representation and low-rank recovery problems, while Tao (2014) proposed ADM with parallel splitting, whose subproblems may not be easily solvable. Moreover, they all developed their theories with the penalty parameter being fixed, resulting in difficulty of tuning an optimal penalty parameter that fits for different data and data sizes. This has been identified as an important issue (Deng and Yin, 2012).

1.3 Contributions and Differences from Prior Work

To propose an algorithm that is more suitable for convex programs in machine learning, in this paper we aim at combining the advantages of (He and Yuan, 2013), (Tao, 2014), and (Lin et al, 2011), i.e., combining LADM, parallel splitting, and adaptive penalty. Hence we call our method LADM with parallel splitting and adaptive penalty (LADMPSAP). With LADM, the subproblems will have forms like (7) and hence can be easily solved. With parallel splitting, the sparsity and low-rankness of iterates can be preserved during iterations when dealing with sparse representation and low-rank recovery problems, saving both the storage and the computation load. With adaptive penalty, the convergence can be faster and it is unnecessary to tune an optimal penalty parameter. Parallel splitting also makes the algorithm highly parallelizable, making LADMPSAP suitable for parallel or distributed computing, which is important for large scale machine learning. When all the component objective functions have bounded subgradients, we prove convergence results that are stronger than the existing theories on ADM and LADM. For example, the penalty parameter can be *unbounded* and the *sufficient and necessary* conditions of the global convergence of LADMPSAP can be obtained as well. We also propose a simple optimality measure and prove the convergence rate of LADMPSAP in an ergodic sense under this measure. Our proof is simpler than those in (He and Yuan, 2012) and (Tao, 2014) which relied on a complex optimality measure. When a convex program has extra convex set constraints, we further devise a practical version of LADMPSAP that converges faster thanks to better parameter analysis. Finally, we generalize LADMPSAP to cope with more difficult f_i's, whose proximal operation (7) is not easily solvable, by further linearizing the smooth components of f_i's. Experiments testify to the advantage of LADMPSAP in speed and numerical accuracy.

Note that Goldfarb and Ma (2012) also proposed a multiple splitting algorithm for convex optimization. However, they only considered a special case of our model problem (1), i.e., all the linear mappings $\mathcal{A}_i$'s are identity mappings[7]. With their simpler model problem, linearization is unnecessary and a faster convergence rate, $O(K^{-2})$, can be achieved. In contrast, in this paper we aim at proposing a practical algorithm for efficiently solving more general problems like (1).

[7] The multi-block problems introduced in (Boyd et al, 2011) also fall within this category.

We also note that Hong and Luo (2012) used the same linearization technique for the smooth components of f_i's as well, but they only considered a special class of f_i's. Namely, the non-smooth component of f_i is a sum of ℓ_1 and ℓ_2 norms or its epigraph is polyhedral. Moreover, for parallel splitting (Jacobi update) Hong and Luo (2012) has to incorporate a postprocessing to guarantee convergence, by interpolating between an intermediate iterate and the previous iterate. Third, Hong and Luo (2012) still focused on a fixed penalty parameter. Again, our method can handle more general f_i's, does not require postprocessing, and allows for an adaptive penalty parameter.

A more general splitting/linearization technique can be founded in (Zhang et al, 2011). However, the authors only proved that any accumulation point of the iteration is a Kuhn-Karush-Tucker (KKT) point and did not investigate the convergence rate. There was no evidence that the iteration could converge to a unique point. Moreover, the authors only studied the case of fixed penalty parameter.

Although dual ascent with dual decomposition (Boyd et al, 2011) can also solve (1) in a parallel way, it may break down when some f_i's are not strictly convex (Boyd et al, 2011), which typically happens in sparse or low-rank recovery problems where ℓ_1 norm or nuclear norm are used. Even if it works, since f_i is not strictly convex, dual ascent becomes dual *subgradient* ascent (Boyd et al, 2011), which is known to converge at a rate of $O(K^{-1/2})$ – slower than our $O(K^{-1})$ rate. Moreover, dual ascent requires choosing a good step size for each iteration, which is less convenient than ADM based methods.

1.4 Organization

The remainder of this paper is organized as follows. We first review LADM with adaptive penalty (LADMAP) for the two-block case in Section 2. Then we present LADMPSAP for the multi-block case in Section 3. Next, we propose a practical version of LADMPSAP for separable convex programs with convex set constraints in Section 4. We further extend LADMPSAP to proximal LADMPSAP for programs with more difficult objective functions in Section 5. We compare the advantage of LADMPSAP in speed and numerical accuracy with other first order methods in Section 6. Finally, we conclude the paper in Section 7.

2 Review of LADMAP for the Two-Block Case

We first review LADMAP (Lin et al, 2011) for the two-block case of (1). It consists of four steps:

1. Update $\mathbf{x}_1$:

$$\mathbf{x}_1^{k+1} = \underset{\mathbf{x}_1}{\operatorname{argmin}} f_1(\mathbf{x}_1) + \frac{\sigma_1^{(k)}}{2} \left\| \mathbf{x}_1 - \mathbf{x}_1^k + \mathcal{A}_1^\dagger(\tilde{\lambda}_1^k)/\sigma_1^{(k)} \right\|^2, \tag{8}$$

2. Update $\mathbf{x}_2$:

$$\mathbf{x}_2^{k+1} = \underset{\mathbf{x}_2}{\operatorname{argmin}} f_2(\mathbf{x}_2) + \frac{\sigma_2^{(k)}}{2} \left\| \mathbf{x}_2 - \mathbf{x}_2^k + \mathcal{A}_2^\dagger(\tilde{\lambda}_2^k)/\sigma_2^{(k)} \right\|^2, \tag{9}$$

3. Update λ:

$$\lambda^{k+1} = \lambda^k + \beta_k \left(\sum_{i=1}^{2} \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b} \right), \tag{10}$$

4. Update β:

$$\beta_{k+1} = \min(\beta_{\max}, \rho\beta_k), \tag{11}$$

where λ is the Lagrange multiplier, β_k is the penalty parameter, $\sigma_i^{(k)} = \eta_i\beta_k$ with $\eta_i > \|\mathcal{A}_i\|^2$ ($\|\mathcal{A}_i\|$ is the operator norm of $\mathcal{A}_i$),

$$\tilde{\lambda}_1^k = \lambda^k + \beta_k(\mathcal{A}_1(\mathbf{x}_1^k) + \mathcal{A}_2(\mathbf{x}_2^k) - \mathbf{b}), \tag{12}$$
$$\tilde{\lambda}_2^k = \lambda^k + \beta_k(\mathcal{A}_1(\mathbf{x}_1^{k+1}) + \mathcal{A}_2(\mathbf{x}_2^k) - \mathbf{b}), \tag{13}$$

and ρ is an adaptively updated parameter (see (20)). Please refer to (Lin et al, 2011) for details. Note that the latest $\mathbf{x}_1^{k+1}$ is immediately used to compute $\mathbf{x}_2^{k+1}$ (see (13)). So $\mathbf{x}_1$ and $\mathbf{x}_2$ have to be updated alternately, hence the name alternating direction method.

3 LADMPSAP for the Multi-Block Case

In this section, we extend LADMAP for multi-block separable convex programs (1). We also provide the *sufficient and necessary conditions* for global convergence when subgradients of the objective functions are all bounded. We further prove the convergence rate in an ergodic sense.

3.1 LADM with Parallel Splitting and Adaptive Penalty

Contrary to our intuition, the multi-block case is actually fundamentally different from the two-block one. For the multi-block case, it is very natural to generalize LADMAP for the two-block case in a straightforward way, with

$$\tilde{\lambda}_i^k = \lambda^k + \beta_k \left(\sum_{j=1}^{i-1} \mathcal{A}_j(\mathbf{x}_j^{k+1}) + \sum_{j=i}^{n} \mathcal{A}_j(\mathbf{x}_j^k) - \mathbf{b} \right), \ i = 1, \cdots, n. \tag{14}$$

Unfortunately, we were unable to prove the convergence of such a naive LADMAP using the same proof for the two-block case. This is because their Fejér monotone inequalities (see Remark 4) cannot be the same. That is why He et al. has to introduce an extra Gaussian back substitution (He et al, 2012; He and

Yuan, 2013) for correcting the iterates. Actually, the above naive generalization of LADMAP may be divergent (which is even worse than converging to a wrong solution), e.g., when applied to the following problem:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} \|\mathbf{x}_i\|_1, \quad s.t. \quad \sum_{i=1}^{n} \mathbf{A}_i\mathbf{x}_i = \mathbf{b}, \tag{15}$$

where $n \geq 5$ and $\mathbf{A}_i$ and $\mathbf{b}$ are Gaussian random matrix and vector, respectively, whose entries fulfil the standard Gaussian distribution independently. Chen et al (2013) also analyzed the naively generalized ADM for the multi-block case and showed that even for three blocks the iteration could still be divergent. They also provided sufficient conditions, which basically require that the linear mappings $\mathcal{A}_i$ should be orthogonal to each other ($\mathcal{A}_i^\dagger\mathcal{A}_j = 0$, $i \neq j$), to ensure the convergence of naive ADM.

Fortunately, by modifying $\tilde{\lambda}_i^k$ slightly we are able to prove the convergence of the corresponding algorithm. More specifically, our algorithm for solving (1) consists of the following steps:

1. Update $\mathbf{x}_i$'s in parallel:

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{x}_i}{\operatorname{argmin}} f_i(\mathbf{x}_i) + \frac{\sigma_i^{(k)}}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k + \mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)}\right\|^2, \quad i = 1,\cdots,n, \tag{16}$$

2. Update λ:

$$\lambda^{k+1} = \lambda^k + \beta_k\left(\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b}\right), \tag{17}$$

3. Update β:

$$\beta_{k+1} = \min(\beta_{\max}, \rho\beta_k), \tag{18}$$

where $\sigma_i^{(k)} = \eta_i\beta_k$,

$$\hat{\lambda}^k = \lambda^k + \beta_k\left(\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b}\right), \tag{19}$$

and

$$\rho = \begin{cases} \rho_0, \text{ if } \beta_k \max\left(\left\{\sqrt{\eta_i}\left\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\right\|, i = 1,\cdots,n\right\}\right)/\|\mathbf{b}\| < \varepsilon_2, \\ 1, \quad \text{otherwise}, \end{cases} \tag{20}$$

with $\rho_0 > 1$ being a constant and $0 < \varepsilon_2 \ll 1$ being a threshold. Indeed, we replace $\tilde{\lambda}_i^k$ with $\hat{\lambda}^k$ as (19), which is independent of i, and the rest procedures of the algorithm, including the scheme (18) and (20) to update the penalty parameter, are all inherited from (Lin et al, 2011), except that η_i's have to be made larger (see Theorem 1). As now $\mathbf{x}_i$'s are updated in parallel and β_k changes adaptively, we call the new algorithm LADM with *parallel splitting* and *adaptive penalty* (LADMPSAP).

Algorithm 1 LADMPSAP for Solving (1)

Initialize: Set $\rho_0 > 1$, $\varepsilon_1 > 0$, $\varepsilon_2 > 0$, $\beta_{\max} \gg 1 \gg \beta_0 > 0$, λ^0, $\eta_i > n\|\mathcal{A}_i\|^2$, $\mathbf{x}_i^0$, $i = 1, \cdots, n$.

while (21) or (22) is not satisfied **do**

Step 1: Compute $\hat{\lambda}^k$ as (19).

Step 2: Update $\mathbf{x}_i$'s in parallel by solving

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{x}_i}{\operatorname{argmin}} f_i(\mathbf{x}_i) + \frac{\eta_i\beta_k}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k + \mathcal{A}_i^\dagger(\hat{\lambda}^k)/(\eta_i\beta_k)\right\|^2,\ i = 1, \cdots, n. \tag{23}$$

Step 3: Update λ by (17) and β by (18) and (20).

end while

3.2 Stopping Criteria

Some existing work (e.g., (Liu et al, 2010; Favaro et al, 2011)) proposed stopping criteria out of intuition only, which may not guarantee that the correct solution is approached. Recently, Lin et al (2009a) and Boyd et al (2011) suggested that the stopping criteria can be derived from the KKT conditions of a problem. Here we also adopt such a strategy. Specifically, the iteration terminates when the following two conditions are met:

$$\left\|\sum_{i=1}^n \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b}\right\| / \|\mathbf{b}\| < \varepsilon_1, \tag{21}$$

$$\beta_k \max\left(\left\{\sqrt{\eta_i}\left\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\right\|, i = 1, \cdots, n\right\}\right) / \|\mathbf{b}\| < \varepsilon_2. \tag{22}$$

The first condition measures the feasibility error. The second condition is derived by comparing the KKT conditions of problem (1) and the optimality condition of subproblem (23). The rules (18) and (20) for updating β are actually hinted by the above stopping criteria such that the two errors are well balanced.

For better reference, we summarize the proposed LADMPSAP algorithm in Algorithm 1. For fast convergence, we suggest that $\beta_0 = \alpha m\varepsilon_2$ and $\alpha > 0$ and $\rho_0 > 1$ should be chosen such that β_k increases steadily along with iterations.

3.3 Global Convergence

In the following, we always use $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*)$ to denote the KKT point of problem (1). For the global convergence of LADMPSAP, we have the following theorem, where we denote $\{\mathbf{x}_i^k\} = \{\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k\}$ for simplicity.

Theorem 1 (Convergence of LADMPSAP)[8] *If $\{\beta_k\}$ is non-decreasing and upper bounded, $\eta_i > n\|\mathcal{A}_i\|^2$, $i = 1, \cdots, n$, then $\{(\{\mathbf{x}_i^k\}, \lambda^k)\}$ generated by LADMPSAP converge to a KKT point of problem (1).*

[8] Please see Appendix for all the proofs of our theoretical results hereafter.

3.4 Enhanced Convergence Results

Theorem 1 is a convergence result for general convex programs (1), where f_i's are general convex functions and hence $\{\beta_k\}$ needs to be bounded. Actually, almost all the existing theories on ADM and LADM even assumed a fixed β. For adaptive β_k, it will be more convenient if a user needs not to specify an upper bound on $\{\beta_k\}$ because imposing a large upper bound essentially equals to allowing $\{\beta_k\}$ to be unbounded. Since many machine learning problems choose f_i's as matrix/vector norms, which result in bounded subgradients, we find that the boundedness assumption can be removed. Moreover, we can further prove the *sufficient and necessary* condition for global convergence.

Theorem 2 (Sufficient Condition for Global Convergence) *If $\{\beta_k\}$ is non-decreasing and $\sum_{k=1}^{+\infty} \beta_k^{-1} = +\infty$, $\eta_i > n\|\mathcal{A}_i\|^2$, $\partial f_i(\mathbf{x})$ is bounded, $i = 1, \cdots, n$, then the sequence $\{\mathbf{x}_i^k\}$ generated by LADMPSAP converges to an optimal solution to (1).*

Remark 1 Theorem 2 does not claim that $\{\lambda^k\}$ converges to a point λ^∞. However, as we are more interested in $\{\mathbf{x}_i^k\}$, such a weakening is harmless.

We also have the following result on the necessity of $\sum_{k=1}^{+\infty} \beta_k^{-1} = +\infty$.

Theorem 3 (Necessary Condition for Global Convergence) *If $\{\beta_k\}$ is non-decreasing, $\eta_i > n\|\mathcal{A}_i\|^2$, $\partial f_i(\mathbf{x})$ is bounded, $i = 1, \cdots, n$, then $\sum_{k=1}^{+\infty} \beta_k^{-1} = +\infty$ is also a necessary condition for the global convergence of $\{\mathbf{x}_i^k\}$ generated by LADMPSAP to an optimal solution to (1).*

With the above analysis, when all the subgradients of the component objective functions are bounded we can remove $\beta_{\max}$ in Algorithm 1.

3.5 Convergence Rate

The convergence rate of ADM and LADM in the traditional sense is an open problem (Goldfarb and Ma, 2012). Although Hong and Luo (2012) claimed that they proved the linear convergence rate of ADM, their assumptions are actually quite strong. They assumed that the non-smooth part of f_i is a sum of ℓ_1 and ℓ_2 norms or its epigraph is polyhedral. Moreover, the convex constraint sets should all be polyhedral and bounded. So although their results are encouraging, for general convex programs the convergence rate is still a mystery. Recently, He and Yuan (2012) and Tao (2014) proved an $O(1/K)$ convergence rate of ADM and ADM with parallel splitting in an ergodic sense, respectively. Namely $\frac{1}{K}\sum_{k=1}^{K} \mathbf{x}_i$ violates an optimality measure in $O(1/K)$. Their proof is lengthy and is for fixed penalty parameter only.

In this subsection, based on a simple optimality measure we give a simple proof for the convergence rate of LADMPSAP. For simplicity, we denote $\mathbf{x} = (\mathbf{x}_1^T, \cdots, \mathbf{x}_n^T)^T$, $\mathbf{x}^* = ((\mathbf{x}_1^*)^T, \cdots, (\mathbf{x}_2^*)^T)^T$, and $f(\mathbf{x}) = \sum_{i=1}^n f_i(\mathbf{x}_i)$. We first have the following proposition.

Proposition 1 *$\tilde{\mathbf{x}}$ is an optimal solution to (1) if and only if there exists $\alpha > 0$, such that*

$$f(\tilde{\mathbf{x}}) - f(\mathbf{x}^*) + \sum_{i=1}^n \left\langle \mathcal{A}_i^\dagger(\lambda^*), \tilde{\mathbf{x}}_i - \mathbf{x}_i^* \right\rangle + \alpha \left\| \sum_{i=1}^n \mathcal{A}_i(\tilde{\mathbf{x}}_i) - \mathbf{b} \right\|^2 = 0. \tag{24}$$

Since the left hand side of (24) is always nonnegative and it becomes zero only when $\tilde{\mathbf{x}}$ is an optimal solution, we may use its magnitude to measure how far a point $\tilde{\mathbf{x}}$ is from an optimal solution. Note that in the unconstrained case, as in APG (Beck and Teboulle, 2009), one may simply use $f(\tilde{\mathbf{x}}) - f(\mathbf{x}^*)$ to measure the optimality. But here we have to deal with the constraints. Our criterion is simpler than that in (He and Yuan, 2012; Tao, 2014), which has to compare $(\{\mathbf{x}_i^k\}, \lambda^k)$ with all $(\mathbf{x}_1, \cdots, \mathbf{x}_n, \lambda) \in \mathbb{R}^{d_1} \times \cdots \times \mathbb{R}^{d_n} \times \mathbb{R}^m$.

Then we have the following convergence rate theorem for LADMPSAP in an ergodic sense.

Theorem 4 (Convergence Rate of LADMPSAP) *Define $\bar{\mathbf{x}}^K = \sum_{k=0}^K \gamma_k \mathbf{x}^{k+1}$, where $\gamma_k = \beta_k^{-1} / \sum_{j=0}^K \beta_j^{-1}$. Then the following inequality holds for $\bar{\mathbf{x}}^K$:*

$$\begin{aligned} &f(\bar{\mathbf{x}}^K) - f(\mathbf{x}^*) + \sum_{i=1}^n \left\langle \mathcal{A}_i^\dagger(\lambda^*), \bar{\mathbf{x}}_i^K - \mathbf{x}_i^* \right\rangle + \frac{\alpha\beta_0}{2} \left\| \sum_{i=1}^n \mathcal{A}_i(\bar{\mathbf{x}}_i^K) - \mathbf{b} \right\|^2 \\ &\le C_0 / \left(2 \sum_{k=0}^K \beta_k^{-1} \right), \end{aligned} \tag{25}$$

where $\alpha^{-1} = (n+1) \max\left(1, \left\{ \frac{\|\mathcal{A}_i\|^2}{\eta_i - n\|\mathcal{A}_i\|^2}, i = 1, \cdots, n \right\}\right)$ and $C_0 = \sum_{i=1}^n \eta_i \left\|\mathbf{x}_i^0 - \mathbf{x}_i^*\right\|^2 + \beta_0^{-2} \left\|\lambda^0 - \lambda^*\right\|^2$.

Theorem 4 means that $\bar{\mathbf{x}}^K$ is by $O\left(1 / \sum_{k=0}^K \beta_k^{-1}\right)$ from being an optimal solution. This theorem holds for both bounded and unbounded $\{\beta_k\}$. In the bounded case, $O\left(1 / \sum_{k=0}^K \beta_k^{-1}\right)$ is simply $O(1/K)$. Theorem 4 also hints that $\sum_{k=0}^K \beta_k^{-1}$ should approach infinity to guarantee the convergence of LADMPSAP, which is consistent with Theorem 3.

4 Practical LADMPSAP for Convex Programs with Convex Set Constraints

In real applications, we are often faced with convex programs with convex set constraints:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} f_i(\mathbf{x}_i),\ s.t.\ \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b},\ \mathbf{x}_i \in X_i, i = 1, \cdots, n, \tag{26}$$

where $X_i \subseteq \mathbb{R}^{d_i}$ is a closed convex set. In this section, we consider to extend LADMPSAP to solve the more complex convex set constraint model (26). We assume that the projections onto X_i's are all easily computable. For many convex sets used in machine learning, such an assumption is valid, e.g., when X_i's are nonnegative cones or positive semi-definite cones. In the following, we discuss how to solve (26) efficiently. For simplicity, we assume $X_i \neq \mathbb{R}^{d_i}$, $\forall i$. Finally, we assume that $\mathbf{b}$ is an interior point of $\sum_{i=1}^{n} \mathcal{A}_i(X_i)$.

We introduce auxiliary variables $\mathbf{x}_{n+i}$ to convert $\mathbf{x}_i \in X_i$ into $\mathbf{x}_i = \mathbf{x}_{n+i}$ and $\mathbf{x}_{n+i} \in X_i$, $i = 1, \cdots, n$. Then (26) can be reformulated as:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_{2n}} \sum_{i=1}^{2n} f_i(\mathbf{x}_i), \quad s.t. \quad \sum_{i=1}^{2n} \hat{\mathcal{A}}_i(\mathbf{x}_i) = \hat{\mathbf{b}}, \tag{27}$$

where

$$f_{n+i}(\mathbf{x}) \equiv \chi_{X_i}(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x} \in X_i, \\ +\infty, & \text{otherwise}, \end{cases}$$

is the characteristic function of X_i,

$$\hat{\mathcal{A}}_i(\mathbf{x}_i) = \begin{pmatrix} \mathcal{A}_i(\mathbf{x}_i) \\ 0 \\ \vdots \\ \mathbf{x}_i \\ \vdots \\ 0 \end{pmatrix}, \hat{\mathcal{A}}_{n+i}(\mathbf{x}_{n+i}) = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ -\mathbf{x}_{n+i} \\ \vdots \\ 0 \end{pmatrix}, \text{ and } \hat{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{28}$$

where $i = 1, \cdots, n$.

The adjoint operator $\hat{\mathcal{A}}_i^\dagger$ is

$$\hat{\mathcal{A}}_i^\dagger(\mathbf{y}) = \mathcal{A}_i^\dagger(\mathbf{y}_1) + \mathbf{y}_{i+1},\ \hat{\mathcal{A}}_{n+i}^\dagger(\mathbf{y}) = -\mathbf{y}_{i+1},\ i = 1, \cdots, n, \tag{29}$$

where $\mathbf{y}_i$ is the i-th sub-vector of $\mathbf{y}$, partitioned according to the sizes of $\mathbf{b}$ and $\mathbf{x}_i$, $i = 1, \cdots, n$.

Then LADMPSAP can be applied to solve problem (27). The Lagrange multiplier λ and the auxiliary multiplier $\hat{\lambda}$ are respectively updated as

$$\lambda_1^{k+1} = \lambda_1^k + \beta_k \left(\sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b} \right),\ \lambda_{i+1}^{k+1} = \lambda_{i+1}^k + \beta_k(\mathbf{x}_i^{k+1} - \mathbf{x}_{n+i}^{k+1}), \tag{30}$$

$$\hat{\lambda}_1^k = \lambda_1^k + \beta_k \left(\sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b} \right),\ \hat{\lambda}_{i+1}^k = \lambda_{i+1}^k + \beta_k(\mathbf{x}_i^k - \mathbf{x}_{n+i}^k), \tag{31}$$

Algorithm 2 LADMPSAP for (27), also a Practical Algorithm for (26).

Initialize: Set $\rho_0 > 1$, $\varepsilon_1 > 0$, $\varepsilon_2 > 0$, $\beta_{\max} \gg 1 \gg \beta_0 > 0$, $\lambda^0 = ((\lambda_1^0)^T, \cdots, (\lambda_{n+1}^0)^T)^T$, $\eta_i > n\|\mathcal{A}_i\|^2 + 2$, $\eta_{n+i} > 2$, $\mathbf{x}_i^0$, $\mathbf{x}_{n+i}^0 = \mathbf{x}_i^0$, $i = 1, \cdots, n$.
while (21) or (22) is not satisfied **do**
  Step 1: Compute $\hat{\lambda}^k$ as (31).
  Step 2: Update $\mathbf{x}_i$, $i = 1, \cdots, 2n$, in parallel as (32)-(33).
  Step 3: Update λ by (30) and β by (18) and (20).
end while
(Note that in (20), (21), and (22), n and $\mathcal{A}_i$ should be replaced by $2n$ and $\hat{\mathcal{A}}_i$, respectively.)

and $\mathbf{x}_i$ is updated as (see (16))

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}} f_i(\mathbf{x}) + \frac{\eta_i\beta_k}{2}\left\|\mathbf{x} - \mathbf{x}_i^k + [\mathcal{A}_i^\dagger(\hat{\lambda}_1^k) + \hat{\lambda}_{i+1}^k]/(\eta_i\beta_k)\right\|^2, \quad (32)$$

$$\begin{aligned}\mathbf{x}_{n+i}^{k+1} &= \arg\min_{\mathbf{x}\in X_i} \frac{\eta_{n+i}\beta_k}{2}\left\|\mathbf{x} - \mathbf{x}_{n+i}^k - \hat{\lambda}_{i+1}^k/(\eta_{n+i}\beta_k)\right\|^2 \\ &= \pi_{X_i}\left(\mathbf{x}_{n+i}^k + \hat{\lambda}_{i+1}^k/(\eta_{n+i}\beta_k)\right), \end{aligned} \quad (33)$$

where π_{X_i} is the projection onto X_i and $i = 1, \cdots, n$.

As for the choice of η_i's, although we can simply apply Theorem 1 to assign their values as $\eta_i > 2n(\|\mathcal{A}_i\|^2 + 1)$ and $\eta_{n+i} > 2n$, $i = 1, \cdots, n$, such choices are too pessimistic. As η_i's are related to the magnitudes of the differences in $\mathbf{x}_i^{k+1}$ from $\mathbf{x}_i^k$, we had better provide tighter estimate on η_i's in order to achieve faster convergence. Actually, we have the following better result.

Theorem 5 *For problem (27), if $\{\beta_k\}$ is non-decreasing and upper bounded and η_i's are chosen as $\eta_i > n\|\mathcal{A}_i\|^2 + 2$ and $\eta_{n+i} > 2$, $i = 1, \cdots, n$, then the sequence $\{(\{\mathbf{x}_i^k\}, \lambda^k)\}$ generated by LADMPSAP converge to a KKT point of problem (27).*

Finally, we summarize LADMPSAP for problem (27) in Algorithm 2, which is a practical algorithm for solving (26).

Remark 2 Analogs of Theorems 2 and 3 are also true for Algorithm 2 although ∂f_{n+i}'s are unbounded, thanks to our assumptions that all ∂f_i, $i = 1, \cdots, n$, are bounded and $\mathbf{b}$ is an interior point of $\sum_{i=1}^n \mathcal{A}_i(X_i)$, which result in an analog of Proposition 4. Consequently, $\beta_{\max}$ can also be removed if all ∂f_i, $i = 1, \cdots, n$, are bounded.

Remark 3 Since Algorithm 2 is an application of Algorithm 1 to problem (27), only with refined parameter estimation, its convergence rate in an ergodic sense is also $O\left(1/\sum_{k=0}^K \beta_k^{-1}\right)$, where K is the number of iterations.

5 Proximal LADMPSAP for Even More General Convex Programs

In LADMPSAP we have assumed that the subproblems (16) are easily solvable. In many machine learning problems, the functions f_i's are often matrix or vector norms or characteristic functions of convex sets. So this assumption often holds. Nonetheless, this assumption is not always true, e.g., when f_i is the logistic loss function (see (6)). So in this section we aim at generalizing LADMPSAP to solve even more general convex programs (1).

We are interested in the case that f_i can be decomposed into two components:

$$f_i(\mathbf{x}_i) = g_i(\mathbf{x}_i) + h_i(\mathbf{x}_i), \tag{34}$$

where both g_i and h_i are convex, g_i is $C^{1,1}$:

$$\|\nabla g_i(\mathbf{x}) - \nabla g_i(\mathbf{y})\| \leq L_i \|\mathbf{x} - \mathbf{y}\|, \quad \forall x, y \in \mathbb{R}^{d_i}, \tag{35}$$

and h_i may not be differentiable but its proximal operation is easily solvable. For brevity, we call L_i the Lipschitz constant of ∇g_i.

Recall that in each iteration of LADMPSAP, we have to solve subproblem (16). Since now we do not assume that the proximal operation of f_i (7) is easily solvable, we may have difficulty in solving subproblem (16). By (34), we write down (16) as

$$\mathbf{x}_i^{k+1} = \operatorname*{argmin}_{\mathbf{x}_i} h_i(\mathbf{x}_i) + g_i(\mathbf{x}_i) + \frac{\sigma_i^{(k)}}{2} \left\| \mathbf{x}_i - \mathbf{x}_i^k + \mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)} \right\|^2, \quad i = 1, \cdots, n, \tag{36}$$

Since $g_i(\mathbf{x}_i) + \frac{\sigma_i^{(k)}}{2} \left\| \mathbf{x}_i - \mathbf{x}_i^k + \mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)} \right\|^2$ is $C^{1,1}$, we may also linearize it at $\mathbf{x}_i^k$ and add a proximal term. Such an idea leads to the following updating scheme of $\mathbf{x}_i$:

$$\begin{aligned} \mathbf{x}_i^{k+1} &= \operatorname*{argmin}_{\mathbf{x}_i} h_i(\mathbf{x}_i) + g_i(\mathbf{x}_i^k) + \frac{\sigma_i^{(k)}}{2} \left\| \mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)} \right\|^2 \\ &\quad + \langle \nabla g_i(\mathbf{x}_i^k) + \mathcal{A}_i^\dagger(\hat{\lambda}^k), \mathbf{x}_i - \mathbf{x}_i^k \rangle + \frac{\tau_i^{(k)}}{2} \left\| \mathbf{x}_i - \mathbf{x}_i^k \right\|^2 \\ &= \operatorname*{argmin}_{\mathbf{x}_i} h_i(\mathbf{x}_i) + \frac{\tau_i^{(k)}}{2} \left\| \mathbf{x}_i - \mathbf{x}_i^k + \frac{1}{\tau_i^{(k)}} [\mathcal{A}_i^\dagger(\hat{\lambda}^k) + \nabla g_i(\mathbf{x}_i^k)] \right\|^2, \end{aligned} \tag{37}$$

where $i = 1, \cdots, n$. The choice of $\tau_i^{(k)}$ is presented in Theorem 6, i.e. $\tau_i^{(k)} = T_i + \beta_k \eta_i$, where $T_i \geq L_i$ and $\eta_i > n\|\mathcal{A}_i\|^2$ are both positive constants.

By our assumption on h_i, the above subproblems are easily solvable. The update of Lagrange multiplier λ and β are still respectively goes as (17) and (18) but with

$$\rho = \begin{cases} \rho_0, \text{ if } \max\left(\left\{ \|\mathcal{A}_i\|^{-1} \left\| \nabla g_i(\mathbf{x}_i^{k+1}) - \nabla g_i(\mathbf{x}_i^k) - \tau_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) \right\|, \right. \right. \\ \qquad \left. \left. i = 1, \cdots, n \right\} \right) / \|\mathbf{b}\| < \varepsilon_2, \\ 1, \quad \text{otherwise.} \end{cases} \tag{38}$$

Algorithm 3 Proximal LADMPSAP for Solving (1) with f_i Satisfying (34).

Initialize: Set $\rho_0 > 1$, $\beta_0 > 0$, λ^0, $T_i \geq L_i$, $\eta_i > n\|\mathcal{A}_i\|^2$, $\mathbf{x}_i^0$, $i = 1, \cdots, n$.
while (39) or (40) is not satisfied **do**
Step 1: Compute $\hat{\lambda}^k$ as (19).
Step 2: Update $\mathbf{x}_i$'s in parallel by solving

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{x}_i}{\operatorname{argmin}}\, h_i(\mathbf{x}_i) + \frac{\tau_i^{(k)}}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k + \frac{1}{\tau_i^{(k)}}[\mathcal{A}_i^\dagger(\hat{\lambda}^k) + \nabla g_i(\mathbf{x}_i^k)]\right\|^2,\ i = 1, \cdots, n, \tag{41}$$

where $\tau_i^{(k)} = T_i + \beta_k\eta_i$.
Step 3: Update λ by (17) and β by (18) with ρ defined in (38).
end while

The iteration terminates when the following two conditions are met:

$$\left\|\sum_{i=1}^n \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b}\right\| / \|\mathbf{b}\| < \varepsilon_1, \tag{39}$$

$$\max\left(\left\{\|\mathcal{A}_i\|^{-1}\left\|\nabla g_i(\mathbf{x}_i^{k+1}) - \nabla g_i(\mathbf{x}_i^k) - \tau_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k)\right\|, \atop i = 1, \cdots, n\right\}\right) / \|\mathbf{b}\| < \varepsilon_2. \tag{40}$$

These two conditions are also deduced from the KKT conditions.

We call the above algorithm as proximal LADMPSAP and summarize it in Algorithm 3.

As for the convergence of proximal LADMPSAP, we have the following theorem.

Theorem 6 (Convergence of Proximal LADMPSAP) *If β_k is non-decreasing and upper bounded, $\tau_i^{(k)} = T_i + \beta_k\eta_i$, where $T_i \geq L_i$ and $\eta_i > n\|\mathcal{A}_i\|^2$ are both positive constants, $i = 1, \cdots, n$, then $\{(\{\mathbf{x}_i^k\}, \lambda^k)\}$ generated by proximal LADMPSAP converge to a KKT point of problem (1).*

We further have the following convergence rate theorem for proximal LADMPSAP in an ergodic sense.

Theorem 7 (Convergence Rate of Proximal LADMPSAP) *Define $\bar{\mathbf{x}}_i^K = \sum_{k=0}^K \gamma_k \mathbf{x}_i^{k+1}$, where $\gamma_k = \beta_k^{-1} / \sum_{j=0}^K \beta_j^{-1}$. Then the following inequality holds for $\bar{\mathbf{x}}_i^K$:*

$$\sum_{i=1}^n \left(f_i(\bar{\mathbf{x}}_i^K) - f_i(\mathbf{x}_i^*) + \left\langle \mathcal{A}_i^\dagger(\lambda^*), \bar{\mathbf{x}}_i^K - \mathbf{x}_i^* \right\rangle\right) + \frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n \mathcal{A}_i(\bar{\mathbf{x}}_i^K) - \mathbf{b}\right\|^2 \leq C_0 / \sum_{k=0}^K 2\beta_k^{-1}, \tag{42}$$

where $\alpha^{-1} = (n+1)\max\left(1, \left\{\frac{\|\mathcal{A}_i\|^2}{\eta_i - n\|\mathcal{A}_i\|^2}, i = 1, \cdots, n\right\}\right)$ *and* $C_0 = \sum_{i=1}^{n} \beta_0^{-1}\tau_i^{(0)}\|\mathbf{x}_i^0 - \mathbf{x}_i^*\|^2 + \beta_0^{-2}\|\lambda^0 - \lambda^*\|^2$.

When there are extra convex set constraints, $\mathbf{x}_i \in X_i$, $i = 1, \cdots, n$, we can also introduce auxiliary variables as in Section 4 and have an analogy of Theorems 5 and 4.

Theorem 8 *For problem (27), where* f_i *is described at the beginning of Section 5, if* β_k *is non-decreasing and upper bounded and* $\tau_i^{(k)} = T_i + \eta_i\beta_k$, *where* $T_i \geq L_i$, $T_{n+i} = 0$, $\eta_i > n\|\mathcal{A}_i\|^2 + 2$, *and* $\eta_{n+i} > 2$, $i = 1, \cdots, n$, *then* $\{(\{\mathbf{x}_i^k\}, \lambda^k)\}$ *generated by proximal LADMPSAP converge to a KKT point of problem (27). The convergence rate in an ergodic sense is also* $O\left(1/\sum_{k=0}^{K}\beta_k^{-1}\right)$, *where* K *is the number of iterations.*

6 Numerical Results

In this section, we test the performance of LADMPSAP on three specific examples of problem (1), i.e., Latent Low-Rank Representation (see (2)), Nonnegative Matrix Completion (see (3)), and Group Sparse Logistic Regression with Overlap (see (6)).

6.1 Solving Latent Low-Rank Representation

We first solve the latent LRR problem Liu and Yan (2011) (2). In order to test LADMPSAP and related algorithms with data whose characteristics are controllable, we follow (Liu et al, 2010) to generate synthetic data, which are parameterized as $(s, p, d, \tilde{r})$, where s, p, d, and $\tilde{r}$ are the number of independent subspaces, points in each subspace, and ambient and intrinsic dimensions, respectively. The number of scale variables and constraints is $(sp) \times d$.

As first order methods are popular for solving convex programs in machine learning (Boyd et al, 2011), here we compare LADMPSAP with several conceivable first order algorithms, including APG (Beck and Teboulle, 2009), naive ADM, naive LADM, LADMGB, and LADMPS. Naive ADM and naive LADM are generalizations of ADM and LADM, respectively, which are straightforwardly generalized from two variables to multiple variables, as discussed in Section 3.1. Naive ADM is applied to solve (2) after rewriting the constraint of (2) as $\mathbf{X} = \mathbf{XP} + \mathbf{QX} + \mathbf{E}, \mathbf{P} = \mathbf{Z}, \mathbf{Q} = \mathbf{L}$. For LADMPS, β_k is fixed in order to show the effectiveness of adaptive penalty. The parameters of APG and ADM are the same as those in (Lin et al, 2009b) and (Liu and Yan, 2011), respectively. For LADM, we follow the suggestions in (Yang and Yuan, 2013) to fix its penalty parameter β at $2.5/\min(d, sp)$, where $d \times sp$ is the size of $\mathbf{X}$. For LADMGB, as there is no suggestion in He and Yuan (2013) on how to choose a fixed β, we simply set it the same as that in LADM. The rest of

the parameters are the same as those suggested in (He et al, 2012). We fix $\beta = \sigma_{\max}(\mathbf{X})\min(d, sp)\varepsilon_2$ in LADMPS and set $\beta_0 = \sigma_{\max}(\mathbf{X})\min(d, sp)\varepsilon_2$ and $\rho_0 = 10$ in LADMPSAP. For LADMPSAP, we also set $\eta_Z = \eta_L = 1.02\times 3\sigma^2_{\max}(\mathbf{X})$, where η_Z and η_L are the parameters η_i's in Algorithm 1 for $\mathbf{Z}$ and $\mathbf{L}$, respectively. For the stopping criteria, $\|\mathbf{X}\mathbf{Z}^k+\mathbf{L}^k\mathbf{X}+\mathbf{E}^k-\mathbf{X}\|/\|\mathbf{X}\| \le \varepsilon_1$ and $\max(\|\mathbf{Z}^k-\mathbf{Z}^{k-1}\|, \|\mathbf{L}^k-\mathbf{L}^{k-1}\|, \|\mathbf{E}^k-\mathbf{E}^{k-1}\|)/\|\mathbf{X}\| \le \varepsilon_2$, with $\varepsilon_1 = 10^{-3}$ and $\varepsilon_2 = 10^{-4}$ are used for all the algorithms. For the parameter μ in (2), we empirically set it as $\mu = 0.01$. To measure the relative errors in the solutions we run LADMPSAP 2000 iterations with $\rho_0 = 1.01$ to obtain the estimated ground truth solution $(\mathbf{Z}^*, \mathbf{L}^*, \mathbf{E}^*)$. The experiments are run and timed on a notebook computer with an Intel Core i7 2.00 GHz CPU and 6GB memory, running Windows 7 and Matlab 7.13.

Table 1 shows the results of related algorithms. We can see that LADMPS and LADMPSAP are faster and more numerically accurate than LADMGB, and LADMPSAP is even faster than LADMPS thanks to the adaptive penalty. Moreover, naive ADM and naive LADM have relatively poorer numerical accuracy, possibly due to converging to wrong solutions. The numerical accuracy of APG is also worse than those of LADMPS and LADMPSAP because it only solves an approximate problem by adding the constraint to the objective function as penalty. Note that although we do not require $\{\beta_k\}$ to be bounded, this does not imply that β_k will grow infinitely. As a matter of fact, when LADMPSAP terminates the final values of β_k are 21.1567, 42.2655, and 81.4227 for the three data settings, respectively.

We then test the performance of the above six algorithms on the Hopkins155 database (Tron and Vidal, 2007), which consists of 156 sequences, each having 39 to 550 data vectors drawn from two or three motions. For computational efficiency, we preprocess the data by projecting them to be 5-dimensional using PCA. We test all algorithms with $\mu = 2.4$, which is the best parameter for LRR on this database (Liu et al, 2010). Table 2 shows the results on the Hopkins155 database. We can also see that LADMPSAP is faster than other methods in comparison. In particular, LADMPSAP is faster than LADMPS, which uses a fixed β. This testify to the advantage of using an adaptive penalty.

6.2 Solving Nonnegative Matrix Completion

This subsection evaluates the performance of the practical LADMPSAP proposed in Section 4 for solving nonnegative matrix completion (Xu et al, 2011) (3).

We first evaluate the numerical performance on synthetic data to demonstrate the superiority of practical LADMPSAP over the conventional LADM[9] (Yang and Yuan, 2013). The nonnegative low-rank matrix $\mathbf{X}_0$ is generated by truncating the singular values of a randomly generated matrix. As LADM

[9] Code available at `http://math.nju.edu.cn/~jfyang/IADM_NNLS/index.html`

Table 1 Comparisons of APG, naive ADM (nADM), naive LADM (nLADM), LADMGB, LADMPS, and LADMPSAP on the latent LRR problem (2). The quantities include computing time (in seconds), number of iterations, relative errors, and clustering accuracy (in percentage). They are averaged over 10 runs.

$(s,\ p,\ d,\ \tilde{r})$	Method	Time	#Iter.	$\frac{\|\hat{\mathbf{Z}}-\mathbf{Z}^*\|}{\|\mathbf{Z}^*\|}$	$\frac{\|\hat{\mathbf{L}}-\mathbf{L}^*\|}{\|\mathbf{L}^*\|}$	$\frac{\|\hat{\mathbf{E}}-\mathbf{E}^*\|}{\|\mathbf{E}^*\|}$	Acc.
(5, 50, 250, 5)	APG	18.20	236	0.3389	0.3167	0.4500	95.6
	nADM	16.32	172	0.3993	0.3928	0.5592	95.6
	nLADM	21.34	288	0.4553	0.4408	0.5607	95.6
	LADMGB	24.10	290	0.4520	0.4355	0.5610	95.6
	LADMPS	17.15	232	0.0163	0.0139	**0.0446**	95.6
	LADMPSAP	**8.04**	**109**	**0.0089**	**0.0083**	0.0464	95.6
(10, 50, 500, 5)	APG	85.03	234	0.1020	0.0844	0.7161	95.8
	nADM	78.27	170	0.0928	0.1026	0.6636	95.8
	nLADM	181.42	550	0.2077	0.2056	0.6623	95.8
	LADMGB	214.94	550	0.1877	0.1848	0.6621	95.8
	LADMPS	64.65	200	0.0167	0.0089	0.1059	95.8
	LADMPSAP	**37.85**	**117**	**0.0122**	**0.0055**	**0.0780**	95.8
(20, 50, 1000, 5)	APG	544.13	233	0.0319	0.0152	0.2126	95.2
	nADM	466.78	166	0.0501	0.0433	0.2676	95.2
	nLADM	1888.44	897	0.1783	0.1746	0.2433	95.2
	LADMGB	2201.37	897	0.1774	0.1736	0.2434	95.2
	LADMPS	367.68	177	0.0151	0.0105	0.0872	95.2
	LADMPSAP	**260.22**	**125**	**0.0106**	**0.0041**	**0.0671**	95.2

Table 2 Comparisons of APG, naive ADM (nADM), naive LADM (nLADM), LADMGB, LADMPS, and LADMPSAP on the Hopkins155 database. The quantities include average computing time, average number of iterations, and average classification errors on all 156 sequences.

Method	Time (seconds)	#Iteration	Error (%)
APG	10.37	67	8.33
nADM	24.76	144	8.33
nLADM	15.50	112	8.33
LADMGB	16.05	113	8.36
LADMPS	15.58	113	8.33
LADMPSAP	**3.80**	**26**	8.33

cannot handle the nonnegativity constraint, it actually solve the standard matrix completion problem, i.e., (3) without the nonnegativity constraint. For LADMPSAP, we follow the conditions in Theorem 5 to set η_i's and set the rest of the parameters the same as those in Section 6.1. The stopping tolerances are set as $\varepsilon_1 = \varepsilon_2 = 10^{-5}$. The numerical comparison is shown in Table 3, where the relative nonnegative feasibility (FA) is defined as (Xu et al, 2011):

$$\text{FA} := \|\min(\hat{\mathbf{X}}, 0)\|/\|\mathbf{X}_0\|,$$

in which $\mathbf{X}_0$ is the ground truth and $\hat{\mathbf{X}}$ is the computed solution. It can be seen that the numerical performance of LADMPSAP is much better than that

Table 3 Comparisons on the NMC problem (3) with synthetic data, averaged on 10 runs. q, t, and d_r denote, respectively, the sample ratio, the number of measurements $t = q(mn)$, and the "degree of freedom" defined by $d_r = r(m + n - r)$ for an $m \times n$ matrix with rank r and q. Here we set $m = n$ and fix $r = 10$ in all the tests.

X			LADM				LADMPSAP			
n	q	t/d_r	#Iter.	Time (s)	RelErr	FA	#Iter.	Time (s)	RelErr	FA
1000	20%	10.05	375	177.92	1.35E-5	6.21E-4	58	**24.94**	**9.67E-6**	**0**
	10%	5.03	1000	459.70	4.60E-5	6.50E-4	109	**42.68**	**1.72E-5**	**0**
5000	20%	50.05	229	1613.68	1.08E-5	1.93E-4	49	**369.96**	**9.05E-6**	**0**
	10%	25.03	539	2028.14	1.20E-5	7.70E-5	89	**365.26**	**9.76E-6**	**0**
10000	10%	50.03	463	6679.59	1.11E-5	4.18E-5	89	**1584.39**	**1.03E-5**	**0**

Table 4 Comparisons on the image inpainting problem. "PSNR" stands for "Peak Signal to Noise Ratio" measured in decibel (dB).

Method	#Iter.	Time (s)	PSNR (dB)	FA
FPCA	179	228.99	27.77	9.41E-4
LADM	228	207.95	26.98	2.92E-3
LADMPSAP	143	**134.89**	**31.39**	**0**

of LADM, thus again verifies the efficiency of our proposed parallel splitting and adaptive penalty scheme for enhancing ADM/LADM type algorithms.

We then consider the image inpainting problem, which is to fill in the missing pixel values of a corrupted image. As the pixel values are nonnegative, the image inpainting problem can be formulated as the NMC problem. To prepare a low-rank image, we also truncate the singular values of a 1024×1024 grayscale image "man"[10] to obtain an image of rank 40, shown in Fig. 1 (a)-(b). The corrupted image is generated from the original image (all pixels have been normalized in the range of [0, 1]) by sampling 20% of the pixels uniformly at random and adding Gaussian noise with mean zero and standard deviation 0.1.

Besides LADM, here we also consider another recently proposed fixed point continuation with approximate SVD (FPCA (Ma et al, 2011)) on this problem. Similar to LADM, the code of FPCA[11] can only solve the standard matrix completion problem without the nonnegativity constraint. This time we set $\varepsilon_1 = 10^{-3}$ and $\varepsilon_2 = 10^{-1}$ as the thresholds for stopping criteria. The recovered images are shown in Fig. 1 (c)-(e) and the quantitative results are in Table 4. One can see that on our test image both the qualitative and the quantitative results of LADMPSAP are better than those of FPCA and LADM. Note that LADMPSAP is faster than FPCA and LADM even though they do not handle the nonnegativity constraint.

[10] Available at `http://sipi.usc.edu/database/`

[11] Code available at `http://www1.se.cuhk.edu.hk/~sqma/softwares.html`

(a) Original (b) Corrupted (c) FPCA (d) LADM (e) LADMPSAP

Fig. 1 Image inpainting by FPCA, LADM and LADMPSAP.

Table 5 Comparisons among ADM, LADM, LADMPS, LADMPSAP, and proximal LADMPSAP (pLADMPSAP) on the group sparse logistic regression with overlap problem. The quantities include the computing time (in seconds), number of outer iterations, and relative errors.

(s, p, t, q)	Method	Time	#Iter.	$\frac{\|\hat{\bar{\mathbf{w}}}-\bar{\mathbf{w}}^*\|}{\|\bar{\mathbf{w}}^*\|}$	$\frac{\|\hat{\mathbf{z}}-\mathbf{z}^*\|}{\|\mathbf{z}^*\|}$
(300, 901, 100, 10)	ADM	294.15	43	0.4800	0.4790
	LADM	229.03	43	0.5331	0.5320
	LADMPS	105.50	47	0.2088	0.2094
	LADMPSAP	57.46	39	0.0371	0.0368
	pLADMPSAP	**1.97**	141	**0.0112**	**0.0112**
(450, 1351, 150, 15)	ADM	450.96	33	0.4337	0.4343
	LADM	437.12	36	0.5126	0.5133
	LADMPS	201.30	39	0.1938	0.1937
	LADMPSAP	136.64	37	0.0321	0.0306
	pLADMPSAP	**4.16**	150	**0.0131**	**0.0131**
(600, 1801, 200, 20)	ADM	1617.09	62	1.4299	1.4365
	LADM	1486.23	63	1.5200	1.5279
	LADMPS	494.52	46	0.4915	0.4936
	LADMPSAP	216.45	32	0.0787	0.0783
	pLADMPSAP	**5.77**	127	**0.0276**	**0.0277**

6.3 Solving Group Sparse Logistic Regression with Overlap

In this subsection, we apply proximal LADMPSAP to solve the problem of group sparse logistic regression with overlap (5).

The Lipschitz constant of the gradient of logistic function with respect to $\bar{\mathbf{w}}$ can be proven to be $L_{\bar{w}} \leq \frac{1}{4s}\|\bar{\mathbf{X}}\|_2^2$, where $\bar{\mathbf{X}} = (\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \cdots, \bar{\mathbf{x}}_s)$. Thus (5) can be directly solved by Algorithm 3.

6.3.1 Synthetic Data

To assess the performance of proximal LADMPSAP, we simulate data with $p = 9t+1$ variables, covered by t groups of ten variables with overlap of one variable between two successive groups: $\{1, \cdots, 10\}$, $\{10, \cdots, 19\}$, $\cdots$, $\{p-9, \cdots, p\}$. We randomly choose q groups to be the support of $\mathbf{w}$. If the chosen groups have overlapping variables with the unchosen groups, the overlapping variables are removed from the support of $\mathbf{w}$. So the support of $\mathbf{w}$ may be less than $10q$. $\mathbf{y} = (y_1, \cdots, y_s)^T$ is chosen as $(1, -1, 1, -1, \cdots)^T$. $\mathbf{X} \in \mathbb{R}^{p\times s}$ is generated as

Table 6 Comparisons among the Active Set method (Jacob et al, 2009), LADM, LADMPSAP, and proximal LADMPSAP (pLADMPSAP) on the pathway analysis. We present the CPU time (in seconds), classification error rate, and number of pathways. Results are estimated by three-fold cross validation. #Pathway gives the number of pathways that the selected genes belong to in each of the cross validation.

Method	Time	Error	#Pathway
Active Set	2179	0.36 ± 0.03	6, 5, 78
LADM	2433	0.315 ± 0.049	7, 9, 10
LADMPSAP	1593	0.329 ± 0.011	7, 9, 9
pLADMPSAP	**179**	0.312 ± 0.026	4, 6, 6

follows. For $\mathbf{X}_{i,j}$, if i is in the support of $\mathbf{w}$ and $\mathbf{y}_j = 1$, then $\mathbf{X}_{i,j}$ is generated uniformly on $[0.5, 1.5]$; if i is in the support of $\mathbf{w}$ and $\mathbf{y}_j = -1$, then $\mathbf{X}_{i,j}$ is generated uniformly on $[-1.5, -0.5]$; if i is not in the support of $\mathbf{w}$, then $\mathbf{X}_{i,j}$ is generated uniformly on $[-0.5, 0.5]$. Then the rows whose indices are in the support of $\mathbf{w}$ are statistically different from the remaining rows in $\mathbf{X}$, hence can be considered as informative rows. We use model (6) to select the informative rows for classification, where $\mu = 0.1$. If the ground truth support of $\mathbf{w}$ is recovered, then the two groups of data are linearly separable by considering only the coordinates in the support of $\mathbf{w}$.

We compare proximal LADMPSAP with a series of ADM based methods, including ADM, LADM, LADMPS, and LADMPSAP, where the subproblems for $\mathbf{w}$ and $\mathbf{b}$ have to be solved iteratively, e.g., by APG (Beck and Teboulle, 2009). We terminate the inner loop by APG when the norm of gradient of the objective function of the subproblem is less than 10^{-6}. As for the outer loop, we choose $\varepsilon_1 = 2 \times 10^{-4}$ and $\varepsilon_2 = 2 \times 10^{-3}$ as the thresholds to terminate the iterations.

For ADM, LADM, and LADMPS, which use a fixed penalty β, as we do not find any suggestion on its choice in the literature (the choice suggested in (Yang and Yuan, 2013) is for nuclear norm regularized least square problem only) we try multiple choices of β and choose the one that results in the fastest convergence. For LADMPSAP, we set $\beta_0 = 0.2$ and $\rho_0 = 5$. For proximal LADMPSAP we set $T_1 = \frac{1}{4s}\|\bar{\mathbf{X}}\|_2^2$, $\eta_1 = 2.01\|\bar{\mathbf{S}}\|_2^2$, $T_2 = 0$, $\eta_2 = 2.01$, $\beta_0 = 1$, and $\rho_0 = 5$. To measure the relative errors in the solutions we iterate proximal LADMPSAP for 2,000 times and regard its output as the ground truth solution $(\bar{\mathbf{w}}^*, \mathbf{z}^*)$.

Table 5 shows the comparison among related algorithms. The ground truth support of $\mathbf{w}$ is recovered by all the compared algorithms. We can see that ADM, LADM, LADMPS, and LADMPSAP are much slower than proximal LADMPSAP because of the time-consuming subproblem computation, although they have much smaller number of outer iterations. Their numerical accuracies are also inferior to that of proximal LADMPSAP. We can also see that LADMPSAP is faster and more numerically accurate than ADM, LADM, and LADMPS. This again testifies to the effectiveness of using adaptive penalty.

6.3.2 Pathway Analysis on Breast Cancer Data

Then we consider the pathway analysis problem using the breast cancer gene expression data set (van de Vijver et al, 2002), which consists of 8141 genes in 295 breast cancer tumors (78 metastatic and 217 non-metastatic). We follow Jacob et al (2009) and use the canonical pathways from MSigDB (Subramanian et al, 2005) to generate the overlapping gene sets, which contains 639 groups of genes, 637 of which involve genes from our study. The statistics of the 637 gene groups are summarized as follows: the average number of genes in each group is 23.7, the largest gene group has 213 genes, and 3510 genes appear in these 637 groups with an average appearance frequency of about four. We follow Jacob et al (2009) to restrict the analysis to the 3510 genes and balance the data set by using three replicates of each metastasis patient in the training set. We use model (6) to select genes, where $\mu = 0.08$. We want to predict whether a tumor is metastatic ($y_i = 1$) or non-metastatic ($y_i = -1$).

We compare proximal LADMPSAP with the active set method, which was adopted in (Jacob et al, 2009)[12], LADM, and LADMPSAP. In LADMPSAP and proximal LADMPSAP, we both set $\beta_0 = 0.8$ and $\rho_0 = 1.1$. For LADM, we try multiple choices of β and choose the one that results in the fastest convergence. In LADM and LADMPSAP, we terminate the inner loop by APG when the norm of gradient of the objective function of the subproblem is less than 10^{-6}. The thresholds for terminating the outer loop are all chosen as $\varepsilon_1 = 10^{-3}$ and $\varepsilon_2 = 6 \times 10^{-3}$. For the three LADM based methods, we first solve (6) to select genes. Then we use the selected genes to re-train a traditional logistic regression model and use the model to predict the test samples. As in (Jacob et al, 2009) we partition the whole data set into three subsets to do the experiment three times. Each time we select one subset as the test set and the other two as the training set (i.e., there are $(78+217) \times 2/3 = 197$ samples for training). It is worth mentioning that Jacob et al (2009) only kept the 300 genes that are the most correlated with the output in the pre-processing step. In contrast, we use all the 3510 genes in the training phase.

Table 6 shows that proximal LADMPSAP is more than ten times faster than the active set method used in (Jacob et al, 2009), although it computes with a more than ten times larger training set. Proximal LADMPSAP is also much faster than LADM and LADMPSAP due to the lack of inner loop to solve subproblems. The prediction error and the sparseness at the pathway level by proximal LADMPSAP is also competitive with those of other methods in comparison.

7 Conclusions

In this paper, we propose linearized alternating direction method with parallel splitting and adaptive penalty (LADMPSAP) for efficiently solving linearly constrained multi-block separable convex programs, which are abundant

[12] Code available at `http://cbio.ensmp.fr/~ljacob/documents/overlasso-package.tgz`.

in machine learning. LADMPSAP fully utilizes the properties that the proximal operations of the component objective functions and the projections onto convex sets are easily solvable, which are usually satisfied by machine learning problems, making each of its iterations cheap. It is also highly parallel, making it appealing for parallel or distributed computing. Numerical experiments testify to the advantages of LADMPSAP over other possible first order methods.

Although LADMPSAP is inherently parallel, when solving the proximal operations of component objective functions we will still face basic numerical algebraic computations. So for particular large scale machine learning problems, it will be interesting to integrate the existing distributed computing techniques (e.g., parallel incomplete Cholesky factorization (Chang et al, 2007; Chang, 2011) and caching factorization techniques (Boyd et al, 2011)) with our LADMPSAP in order to effectively address the scalability issues.

Acknowledgements Z. Lin is supported by NSFC (Nos. 61272341, 61231002, 61121002). R. Liu is supported by NSFC (No. 61300086), the China Postdoctoral Science Foundation (No. 2013M530917), the Fundamental Research Funds for the Central Universities (No. DUT12RC(3)67) and the Open Project Program of the State Key Lab of CAD&CG (No.A1404), Zhejiang University. Z. Lin also thanks Xiaoming Yuan, Wotao Yin, and Edward Chang for valuable discussions and HTC for financial support.

A Proof of Theorem 1

To prove this theorem, we first have the following lemmas and propositions.

Lemma 1 (KKT Condition) *The Kuhn-Karush-Tucker (KKT) condition of problem (1) is that there exists* $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*)$*, such that*

$$\sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^*) = \mathbf{b}, \tag{43}$$

$$-\mathcal{A}_i^\dagger(\lambda^*) \in \partial f_i(\mathbf{x}_i^*),\ i = 1, \cdots, n, \tag{44}$$

where ∂f_i *is the subgradient of* f_i*.*

The first is the feasibility condition and the second is the duality condition. Such $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*)$ is called a KKT point of problem (1).

Lemma 2 *For* $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$ *generated by Algorithm 1, we have that*

$$-\sigma_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{u}_i^k) \in \partial f_i(\mathbf{x}_i^{k+1}), \quad i = 1, \cdots, n, \tag{45}$$

where $\mathbf{u}_i^k = \mathbf{x}_i^k - \mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)}$.

This can be easily proved by checking the optimality conditions of (16).

Lemma 3 *For* $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$ *generated by Algorithm 1 and a KKT point* $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*)$ *of problem (1), the following inequality holds:*

$$\left\langle -\sigma_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{u}_i^k) + \mathcal{A}_i^\dagger(\lambda^*), \mathbf{x}_i^{k+1} - \mathbf{x}_i^* \right\rangle \geq 0, \quad i = 1, \cdots, n. \tag{46}$$

This can be deduced by the monotonicity of subgradient mapping (Rockafellar, 1970).

Lemma 4 *For $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$ generated by Algorithm 1 and a KKT point $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*)$ of problem (1), we have that*

$$\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\right\|^2 + \left\|\lambda^{k+1} - \lambda^*\right\|^2 \tag{47}$$

$$= \beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\|\mathbf{x}_i^{k} - \mathbf{x}_i^*\right\|^2 + \left\|\lambda^{k} - \lambda^*\right\|^2 \tag{48}$$

$$-2\beta_k \sum_{i=1}^{n} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{u}_i^k) + \mathcal{A}_i^\dagger(\lambda^*) \right\rangle \tag{49}$$

$$-\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^{k}\right\|^2 - \left\|\lambda^{k+1} - \lambda^{k}\right\|^2 \tag{50}$$

$$-2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^k - \mathbf{u}_i^k \right\rangle \tag{51}$$

$$+2\left\langle \lambda^{k+1} - \lambda^k, \lambda^{k+1} \right\rangle. \tag{52}$$

Proof This can be easily checked. First, we add (49) and (51) to have

$$-2\beta_k \sum_{i=1}^{n} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{u}_i^k) + \mathcal{A}_i^\dagger(\lambda^*) \right\rangle \tag{53}$$

$$-2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^k - \mathbf{u}_i^k \right\rangle \tag{54}$$

$$= -2\beta_k \sum_{i=1}^{n} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathcal{A}_i^\dagger(\lambda^*) \right\rangle + 2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\rangle \tag{55}$$

$$= -2\beta_k \sum_{i=1}^{n} \left\langle \mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^*), \lambda^* \right\rangle + 2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\rangle \tag{56}$$

$$= -2\left\langle \beta_k \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^*), \lambda^* \right\rangle + 2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\rangle \tag{57}$$

$$= -2\left\langle \beta_k \left(\sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b} \right), \lambda^* \right\rangle \tag{58}$$

$$+2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\rangle \tag{59}$$

$$= -2\left\langle \lambda^{k+1} - \lambda^k, \lambda^* \right\rangle + 2\beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\rangle, \tag{60}$$

where we have used (43) in (57). Then we apply the identity

$$2\left\langle \mathbf{a}_{k+1} - \mathbf{a}^*, \mathbf{a}_{k+1} - \mathbf{a}_k \right\rangle = \|\mathbf{a}_{k+1} - \mathbf{a}^*\|^2 - \|\mathbf{a}_k - \mathbf{a}^*\|^2 + \|\mathbf{a}_{k+1} - \mathbf{a}_k\|^2 \tag{61}$$

to see that (47)-(52) holds. □

Proposition 2 *For $\{(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k,\lambda^k)\}$ generated by Algorithm 1 and a KKT point $(\mathbf{x}_1^*,\cdots,\mathbf{x}_n^*,\lambda^*)$ of problem (1), the following inequality holds:*

$$\beta_k\sum_{i=1}^n\sigma_i^{(k)}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\|^2+\|\lambda^{k+1}-\lambda^*\|^2 \tag{62}$$

$$\le \beta_k\sum_{i=1}^n\sigma_i^{(k)}\|\mathbf{x}_i^{k}-\mathbf{x}_i^*\|^2+\|\lambda^{k}-\lambda^*\|^2 \tag{63}$$

$$-2\beta_k\sum_{i=1}^n\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,-\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)+\mathcal{A}_i^\dagger(\lambda^*)\right\rangle \tag{64}$$

$$-\beta_k\sum_{i=1}^n\left(\sigma_i^{(k)}-n\beta_k\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2-\|\lambda^k-\hat{\lambda}^k\|^2. \tag{65}$$

Proof We continue from (51)-(52). As $\sigma_i^{(k)}(\mathbf{x}_i^k-\mathbf{u}_i^k)=\mathcal{A}_i^\dagger(\hat{\lambda}^k)$, we have

$$-2\beta_k\sum_{i=1}^n\sigma_i^{(k)}\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,\mathbf{x}_i^k-\mathbf{u}_i^k\right\rangle+2\left\langle\lambda^{k+1}-\lambda^k,\lambda^{k+1}\right\rangle \tag{66}$$

$$=-2\beta_k\sum_{i=1}^n\left\langle\mathcal{A}_i(\mathbf{x}_i^{k+1}-\mathbf{x}_i^*),\hat{\lambda}^k\right\rangle+2\left\langle\lambda^{k+1}-\lambda^k,\lambda^{k+1}\right\rangle \tag{67}$$

$$=-2\beta_k\left\langle\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^*),\hat{\lambda}^k\right\rangle+2\left\langle\lambda^{k+1}-\lambda^k,\lambda^{k+1}\right\rangle \tag{68}$$

$$=-2\left\langle\lambda^{k+1}-\lambda^k,\hat{\lambda}^k\right\rangle+2\left\langle\lambda^{k+1}-\lambda^k,\lambda^{k+1}\right\rangle \tag{69}$$

$$=2\left\langle\lambda^{k+1}-\lambda^k,\lambda^{k+1}-\hat{\lambda}^k\right\rangle \tag{70}$$

$$=\|\lambda^{k+1}-\lambda^k\|^2+\|\lambda^{k+1}-\hat{\lambda}^k\|^2-\|\lambda^k-\hat{\lambda}^k\|^2 \tag{71}$$

$$=\|\lambda^{k+1}-\lambda^k\|^2+\beta_k^2\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1}-\mathbf{x}_i^k)\right\|^2-\|\lambda^k-\hat{\lambda}^k\|^2 \tag{72}$$

$$\le\|\lambda^{k+1}-\lambda^k\|^2+\beta_k^2\left(\sum_{i=1}^n\|\mathcal{A}_i\|\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|\right)^2-\|\lambda^k-\hat{\lambda}^k\|^2 \tag{73}$$

$$\le\|\lambda^{k+1}-\lambda^k\|^2+n\beta_k^2\sum_{i=1}^n\|\mathcal{A}_i\|^2\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2-\|\lambda^k-\hat{\lambda}^k\|^2 \tag{74}$$

Plugging the above into (51)-(52), we have (62)-(65). □

Remark 4 Proposition 2 shows that the sequence $\{(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k,\lambda^k)\}$ is Fejér monotone. Proposition 2 is different from Lemma 1 in Supplementary Material of (Lin et al, 2011) because for $n>2$ we cannot obtain an (in)equality that is similar to Lemma 1 in Supplementary Material of (Lin et al, 2011) such that each term with minus sign could be made non-positive. Such Fejér monotone (in)equalities are the corner stones for proving the convergence of Lagrange multiplier based optimization algorithms. As a result, we cannot prove the convergence of the naively generalized LADM for the multi-block case.

Then we have the following proposition.

Proposition 3 *Let $\sigma_i^{(k)}=\eta_i\beta_k$, $i=1,\cdots,n$. If $\{\beta_k\}$ is non-decreasing, $\eta_i>n\|\mathcal{A}_i\|^2$, $i=1,\cdots,n$, $\{(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k,\lambda^k)\}$ is generated by Algorithm 1, and $(\mathbf{x}_1^*,\cdots,\mathbf{x}_n^*,\lambda^*)$ is any KKT point of problem (1), then*

1) $\left\{\sum_{i=1}^n\eta_i\|\mathbf{x}_i^k-\mathbf{x}_i^*\|^2+\beta_k^{-2}\|\lambda^k-\lambda^*\|^2\right\}$ *is nonnegative and non-increasing.*

2) $\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|\to 0$, $i=1,\cdots,n$, *and* $\beta_k^{-1}\|\lambda^k-\hat{\lambda}^k\|\to 0$.

3) $\sum_{k=1}^{+\infty}\beta_k^{-1}\left\langle \mathbf{x}_i^{k+1}-\mathbf{x}_i^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)+\mathcal{A}_i^\dagger(\lambda^*)\right\rangle<+\infty$, $i=1,\cdots,n$.

Proof We divide both sides of (62)-(65) by β_k^2 to have

$$\sum_{i=1}^n \eta_i\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\|^2+\beta_k^{-2}\|\lambda^{k+1}-\lambda^*\|^2 \tag{75}$$

$$\le \sum_{i=1}^n \eta_i\|\mathbf{x}_i^{k}-\mathbf{x}_i^*\|^2+\beta_k^{-2}\|\lambda^{k}-\lambda^*\|^2 \tag{76}$$

$$-2\beta_k^{-1}\sum_{i=1}^n\left\langle \mathbf{x}_i^{k+1}-\mathbf{x}_i^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)+\mathcal{A}_i^\dagger(\lambda^*)\right\rangle \tag{77}$$

$$-\sum_{i=1}^n\left(\eta_i-n\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2 \tag{78}$$

$$-\beta_k^{-2}\|\lambda^k-\hat{\lambda}^k\|^2. \tag{79}$$

Then by (46), $\eta_i>n\|\mathcal{A}_i\|^2$ and the non-decrement of $\{\beta_k\}$, we can easily obtain 1). Second, we sum both sides of (75)-(79) over k to have

$$2\sum_{k=0}^{+\infty}\beta_k^{-1}\sum_{i=1}^n\left\langle \mathbf{x}_i^{k+1}-\mathbf{x}^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)+\mathcal{A}_i^\dagger(\lambda^*)\right\rangle \tag{80}$$

$$+\sum_{i=1}^n\left(\eta_i-n\|\mathcal{A}_i\|^2\right)\sum_{k=0}^{+\infty}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2 \tag{81}$$

$$+\sum_{k=0}^{+\infty}\beta_k^{-2}\|\lambda^k-\hat{\lambda}^k\|^2 \tag{82}$$

$$\le \sum_{i=1}^n \eta_i\|\mathbf{x}_i^{0}-\mathbf{x}^*\|^2+\beta_0^{-2}\|\lambda^{0}-\lambda^*\|^2. \tag{83}$$

Then 2) and 3) can be easily deduced. □

Now we are ready to prove Theorem 1. The proof resembles that in (Lin et al, 2011).

Proof **(of Theorem 1)** By Proposition 3-1) and the boundedness of $\{\beta_k\}$, $\{(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k,\lambda^k)\}$ is bounded, hence has an accumulation point, say $(\mathbf{x}_1^{k_j},\cdots,\mathbf{x}_n^{k_j},\lambda^{k_j})\to(\mathbf{x}_1^\infty,\cdots,\mathbf{x}_n^\infty,\lambda^\infty)$. We accomplish the proof in two steps.

1. We first prove that $(\mathbf{x}_1^\infty,\cdots,\mathbf{x}_n^\infty,\lambda^\infty)$ is a KKT point of problem (1).

By Proposition 3-2),

$$\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^k)-\mathbf{b}=\beta_k^{-1}(\hat{\lambda}^k-\lambda^k)\to 0.$$

So any accumulation point of $\{(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k)\}$ is a feasible solution.

Since $-\sigma_i^{(k_j-1)}(\mathbf{x}_i^{k_j}-\mathbf{u}_i^{k_j-1})\in\partial f_i(\mathbf{x}_i^{k_j})$, we have

$$\begin{aligned}\sum_{i=1}^n f_i(\mathbf{x}_i^{k_j}) &\le \sum_{i=1}^n f_i(\mathbf{x}_i^*)+\sum_{i=1}^n\left\langle \mathbf{x}_i^{k_j}-\mathbf{x}_i^*, -\sigma_i^{(k_j-1)}(\mathbf{x}_i^{k_j}-\mathbf{u}_i^{k_j-1})\right\rangle\\ &= \sum_{i=1}^n f_i(\mathbf{x}_i^*)+\sum_{i=1}^n\left\langle \mathbf{x}_i^{k_j}-\mathbf{x}_i^*, -\eta_i\beta_{k_j-1}(\mathbf{x}_i^{k_j}-\mathbf{x}_i^{k_j-1})-\mathcal{A}_i^\dagger(\hat{\lambda}^{k_j-1})\right\rangle.\end{aligned}$$

Let $j \to +\infty$. By observing Proposition 3-2) and the boundedness of $\{\beta_k\}$, we have

$$\begin{aligned}\sum_{i=1}^n f_i(\mathbf{x}_i^\infty) &\le \sum_{i=1}^n f_i(\mathbf{x}_i^*) + \sum_{i=1}^n \left\langle \mathbf{x}_i^\infty - \mathbf{x}_i^*, -\mathcal{A}_i^\dagger(\lambda^\infty)\right\rangle \\ &= \sum_{i=1}^n f_i(\mathbf{x}_i^*) - \sum_{i=1}^n \left\langle \mathcal{A}(\mathbf{x}_i^\infty - \mathbf{x}_i^*), \lambda^\infty\right\rangle \\ &= \sum_{i=1}^n f_i(\mathbf{x}_i^*) - \left\langle \sum_{i=1}^n \mathcal{A}(\mathbf{x}_i^\infty) - \mathbf{b}, \lambda^\infty\right\rangle \\ &= \sum_{i=1}^n f_i(\mathbf{x}_i^*).\end{aligned}$$

So we conclude that $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty)$ is an optimal solution to (1).

Again by $-\sigma_i^{(k_j-1)}(\mathbf{x}_i^{k_j} - \mathbf{u}_i^{k_j-1}) \in \partial f_i(\mathbf{x}_i^{k_j})$ we have

$$\begin{aligned}f_i(\mathbf{x}) &\ge f_i(\mathbf{x}_i^{k_j}) + \left\langle \mathbf{x} - \mathbf{x}_i^{k_j}, -\sigma_i^{(k_j-1)}(\mathbf{x}_i^{k_j} - \mathbf{u}_i^{k_j-1})\right\rangle \\ &= f_i(\mathbf{x}_i^{k_j}) + \left\langle \mathbf{x} - \mathbf{x}_i^{k_j}, -\eta_i\beta_{k_j-1}(\mathbf{x}_i^{k_j} - \mathbf{x}_i^{k_j-1}) - \mathcal{A}_i^\dagger(\hat\lambda^{k_j-1})\right\rangle.\end{aligned}$$

Fixing $\mathbf{x}$ and letting $j \to +\infty$, we see that

$$f_i(\mathbf{x}) \ge f_i(\mathbf{x}_i^\infty) + \left\langle \mathbf{x} - \mathbf{x}_i^\infty, -\mathcal{A}_i^\dagger(\lambda^\infty)\right\rangle, \quad \forall \mathbf{x}.$$

So $-\mathcal{A}_i^\dagger(\lambda^\infty) \in \partial f_i(\mathbf{x}_i^\infty)$, $i = 1, \cdots, n$. Thus $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$ is a KKT point of problem (1).

2. We next prove that the whole sequence $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$ converges to $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$.

By choosing $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*) = (\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$ in Proposition 3, we have

$$\sum_{i=1}^n \eta_i\|\mathbf{x}_i^{k_j} - \mathbf{x}_i^\infty\|^2 + \beta_{k_j}^{-2}\|\lambda^{k_j} - \lambda^\infty\|^2 \to 0.$$

By Proposition 3-1), we readily have

$$\sum_{i=1}^n \eta_i\|\mathbf{x}_i^k - \mathbf{x}_i^\infty\|^2 + \beta_k^{-2}\|\lambda^k - \lambda^\infty\|^2 \to 0.$$

So $(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k) \to (\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$.

As $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$ can be an arbitrary accumulation point of $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$, we conclude that $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$ converge to a KKT point of problem (1). □

B Proof of Theorem 2

We first have the following proposition.

Proposition 4 *If $\{\beta_k\}$ is non-decreasing and unbounded, $\eta_i > n\|\mathcal{A}_i\|^2$ and $\partial f_i(\mathbf{x})$ is bounded for $i = 1, \cdots, n$, then Proposition 3 holds and*

$$\beta_k^{-1}\lambda^k \to 0. \tag{84}$$

Proof As the conditions here are stricter than those in Proposition 3, Proposition 3 holds. Then we have that $\{\beta_k^{-1}\|\lambda^k - \lambda^*\|\}$ is bounded due to Proposition 3-1). So $\{\beta_k^{-1}\lambda^k\}$ is bounded due to $\beta_k^{-1}\|\lambda^k\| \le \beta_k^{-1}\|\lambda^k - \lambda^*\| + \beta_k^{-1}\|\lambda^*\|$. $\{\beta_k^{-1}\hat\lambda^k\}$ is also bounded thanks to Proposition 3-2).

We rewrite Lemma 2 as

$$-\eta_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) - \mathcal{A}_i^\dagger(\beta_k^{-1}\hat\lambda^k) \in \beta_k^{-1}\partial f_i(\mathbf{x}_i^{k+1}), \quad i = 1, \cdots, n. \tag{85}$$

Then by the boundedness of $\partial f_i(x)$, the unboundedness of $\{\beta_k\}$ and Proposition 3-2), letting $k \to +\infty$, we have that

$$\mathcal{A}_i^\dagger(\check{\lambda}^\infty) = 0, \quad i = 1, \cdots, n. \tag{86}$$

where $\check{\lambda}^\infty$ is any accumulation point of $\{\beta_k^{-1}\hat{\lambda}^k\}$, which is the same as that of $\{\beta_k^{-1}\lambda^k\}$ due to Proposition 3-2).

Recall that we have assumed that the mapping $\mathcal{A}(\mathbf{x}_1, \cdots, \mathbf{x}_n) \equiv \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i)$ is onto. So $\cap_{i=1}^n null(\mathcal{A}_i^\dagger) = 0$. Therefore by (86), $\check{\lambda}^\infty = 0$. □

Based on Proposition 4, we can prove Theorem 2 as follows.

Proof **(of Theorem 2)** When $\{\beta_k\}$ is bounded, the convergence has been proven in Theorem 1. In the following, we only focus on the case that $\{\beta_k\}$ is unbounded.

By Proposition 3-1), $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k)\}$ is bounded, hence has at least one accumulation point $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty)$. By Proposition 3-2), $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty)$ is a feasible solution.

Since $\sum_{k=1}^{+\infty} \beta_k^{-1} = +\infty$ and Proposition 3-3), there exists a subsequence $\{(\mathbf{x}_1^{k_j}, \cdots, \mathbf{x}_n^{k_j})\}$ such that

$$\left\langle \mathbf{x}_i^{k_j} - \mathbf{x}_i^*, -\sigma_i^{(k_j-1)}(\mathbf{x}_i^{k_j} - \mathbf{u}_i^{k_j-1}) + \mathcal{A}_i^\dagger(\lambda^*) \right\rangle \to 0, \quad i = 1, \cdots, n. \tag{87}$$

As $\mathbf{p}_i^{k_j} \equiv -\sigma_i^{(k_j-1)}(\mathbf{x}_i^{k_j} - \mathbf{u}_i^{k_j-1}) \in \partial f_i(\mathbf{x}_i^{k_j})$ and ∂f_i is bounded, we may assume that

$$\mathbf{x}_i^{k_j} \to \mathbf{x}_i^\infty \quad \text{and} \quad \mathbf{p}_i^{k_j} \to \mathbf{p}_i^\infty.$$

It can be easily proven that

$$\mathbf{p}_i^\infty \in \partial f_i(\mathbf{x}_i^\infty).$$

Then letting $j \to \infty$ in (87), we have

$$\left\langle \mathbf{x}_i^\infty - \mathbf{x}_i^*, \mathbf{p}_i^\infty + \mathcal{A}_i^\dagger(\lambda^*) \right\rangle = 0, \quad i = 1, \cdots, n. \tag{88}$$

Then by $\mathbf{p}_i^{k_j} \in \partial f_i(\mathbf{x}_i^{k_j})$,

$$\sum_{i=1}^{n} f_i(\mathbf{x}_i^{k_j}) \le \sum_{i=1}^{n} f_i(\mathbf{x}_i^*) + \sum_{i=1}^{n} \left\langle \mathbf{x}_i^{k_j} - \mathbf{x}_i^*, \mathbf{p}_i^{k_j} \right\rangle. \tag{89}$$

Letting $j \to \infty$ and making use of (88), we have

$$\begin{aligned} \sum_{i=1}^{n} f_i(\mathbf{x}_i^\infty) &\le \sum_{i=1}^{n} f_i(\mathbf{x}_i^*) + \sum_{i=1}^{n} \left\langle \mathbf{x}_i^\infty - \mathbf{x}_i^*, \mathbf{p}_i^\infty \right\rangle \\ &= \sum_{i=1}^{n} f_i(\mathbf{x}_i^*) - \sum_{i=1}^{n} \left\langle \mathbf{x}_i^\infty - \mathbf{x}_i^*, \mathcal{A}_i^\dagger(\lambda^*) \right\rangle \\ &= \sum_{i=1}^{n} f_i(\mathbf{x}_i^*) - \sum_{i=1}^{n} \left\langle \mathcal{A}_i(\mathbf{x}_i^\infty - \mathbf{x}_i^*), \lambda^* \right\rangle \\ &= \sum_{i=1}^{n} f_i(\mathbf{x}_i^*). \end{aligned} \tag{90}$$

So together with the feasibility of $\{(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty)\}$ we have that $\{(\mathbf{x}_1^{k_j}, \cdots, \mathbf{x}_n^{k_j})\}$ converges to an optimal solution $\{(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty)\}$ to (1).

Finally, we set $\mathbf{x}_i^* = \mathbf{x}_i^\infty$ and λ^* be the corresponding Lagrange multiplier λ^∞ in Proposition 3. By Proposition 4, we have that

$$\sum_{i=1}^{n} \eta_i \|\mathbf{x}_i^{k_j} - \mathbf{x}_i^\infty\|^2 + \beta_{k_j}^{-2} \|\lambda^{k_j} - \lambda^\infty\|^2 \to 0.$$

By Proposition 3-1), we readily have

$$\sum_{i=1}^{n}\eta_i\|\mathbf{x}_i^k-\mathbf{x}_i^\infty\|^2+\beta_k^{-2}\|\lambda^k-\lambda^\infty\|^2\to 0.$$

So $(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k)\to(\mathbf{x}_1^\infty,\cdots,\mathbf{x}_n^\infty)$. □

C Proof of Theorem 3

Proof **(of Theorem 3)** We first prove that there exist linear mappings $\mathcal{B}_i$, $i=1,\cdots,n$, such that $\mathcal{B}_i$'s are not all zeros and $\sum_{i=1}^{n}\mathcal{B}_i\mathcal{A}_i^\dagger=0$. Indeed, $\sum_{i=1}^{n}\mathcal{B}_i\mathcal{A}_i^\dagger=0$ is equivalent to

$$\sum_{i=1}^{n}\mathbf{B}_i\mathbf{A}_i^T=0, \tag{91}$$

where $\mathbf{A}_i$ and $\mathbf{B}_i$ are the matrix representations of $\mathcal{A}_i$ and $\mathcal{B}_i$, respectively. (91) can be further written as

$$(\mathbf{A}_1\ \cdots\ \mathbf{A}_n)\begin{pmatrix}\mathbf{B}_1^T\\ \vdots\\ \mathbf{B}_n^T\end{pmatrix}=0. \tag{92}$$

Recall that we have assumed that the solution to $\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i)=\mathbf{b}$ is non-unique. So $(\mathbf{A}_1\ \cdots\ \mathbf{A}_n)$ is not full column rank hence (92) has nonzero solutions. Thus there exist $\mathcal{B}_i$'s such that they are not all zeros and $\sum_{i=1}^{n}\mathcal{B}_i\mathcal{A}_i^\dagger=0$.

By Lemma 2,

$$-\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)\in\partial f_i(\mathbf{x}_i^{k+1}),\quad i=1,\cdots,n. \tag{93}$$

As ∂f_i is bounded, $i=1,\cdots,n$, so is

$$\sum_{i=1}^{n}\mathcal{B}_i(\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k))=\beta_k(\mathbf{v}^{k+1}-\mathbf{v}^k), \tag{94}$$

where $\mathbf{v}^k=\phi(\mathbf{x}_1^k,\cdots,\mathbf{x}_n^k)$ and

$$\phi(\mathbf{x}_1,\cdots,\mathbf{x}_n)=\sum_{i=1}^{n}\eta_i\mathcal{B}_i(\mathbf{x}_i). \tag{95}$$

In (94) we have utilized $\sum_{i=1}^{n}\mathcal{B}_i\mathcal{A}_i^\dagger=0$ to cancel $\hat{\lambda}^k$, whose boundedness is uncertain.

Then we have that there exists a constant $C>0$ such that

$$\|\mathbf{v}^{k+1}-\mathbf{v}^k\|\le C\beta_k^{-1}. \tag{96}$$

If $\sum_{k=1}^{+\infty}\beta_k^{-1}<+\infty$, then $\{\mathbf{v}^k\}$ is a Cauchy sequence, hence has a limit $\mathbf{v}^\infty$. Define $\mathbf{v}^*=\phi(\mathbf{x}_1^*,\cdots,\mathbf{x}_n^*)$, where $(\mathbf{x}_1^*,\cdots,\mathbf{x}_n^*)$ is any optimal solution. Then

$$\|\mathbf{v}^\infty-\mathbf{v}^*\|=\left\|\mathbf{v}^0+\sum_{k=0}^{\infty}(\mathbf{v}^{k+1}-\mathbf{v}^k)-\mathbf{v}^*\right\| \tag{97}$$

$$\ge\|\mathbf{v}^0-\mathbf{v}^*\|-\sum_{k=0}^{\infty}\|\mathbf{v}^{k+1}-\mathbf{v}^k\| \tag{98}$$

$$\ge\|\mathbf{v}^0-\mathbf{v}^*\|-C\sum_{k=0}^{\infty}\beta_k^{-1}. \tag{99}$$

So if $(\mathbf{x}_1^0, \cdots, \mathbf{x}_n^0)$ is initialized badly such that

$$\|\mathbf{v}^0 - \mathbf{v}^*\| > C \sum_{k=0}^{\infty} \beta_k^{-1}, \tag{100}$$

then $\|\mathbf{v}^\infty - \mathbf{v}^*\| > 0$, which implies that $(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k)$ cannot converge to $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*)$. Note that (100) is possible because ϕ is not a zero mapping given the conditions on $\mathcal{B}_i$. □

D Proofs of Proposition 1 and Theorem 4

Proof **(of Proposition 1)** If $\tilde{\mathbf{x}}$ is optimal, it is easy to check that (24) holds.

Since $-\mathcal{A}_i^\dagger(\lambda^*) \in \partial f_i(\mathbf{x}_i^*)$, we have

$$f(\tilde{\mathbf{x}}) - f(\mathbf{x}^*) + \sum_{i=1}^{n} \left\langle \mathcal{A}_i^\dagger(\lambda^*), \tilde{\mathbf{x}}_i - \mathbf{x}_i^* \right\rangle \geq 0.$$

So if (24) holds, we have

$$f(\tilde{\mathbf{x}}) - f(\mathbf{x}^*) + \sum_{i=1}^{n} \left\langle \mathcal{A}_i^\dagger(\lambda^*), \tilde{\mathbf{x}}_i - \mathbf{x}_i^* \right\rangle = 0, \tag{101}$$

$$\sum_{i=1}^{n} \mathcal{A}_i(\tilde{\mathbf{x}}_i) - \mathbf{b} = 0. \tag{102}$$

With (102), we have

$$\sum_{i=1}^{n} \left\langle \mathcal{A}_i^\dagger(\lambda^*), \tilde{\mathbf{x}}_i - \mathbf{x}_i^* \right\rangle = \sum_{i=1}^{n} \langle \lambda^*, \mathcal{A}_i(\tilde{\mathbf{x}}_i - \mathbf{x}_i^*) \rangle = \left\langle \lambda^*, \sum_{i=1}^{n} \mathcal{A}_i(\tilde{\mathbf{x}}_i - \mathbf{x}_i^*) \right\rangle = 0. \tag{103}$$

So (101) reduces to $f(\tilde{\mathbf{x}}) = f(\mathbf{x}^*)$. As $\tilde{\mathbf{x}}$ satisfies the feasibility condition, it is an optimal solution to (1). □

Proof **(of Theorem 4)** We first deduce

$$\begin{aligned}
&\left\| \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b} \right\|^2 \\
=& \left\| \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b} + \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) \right\|^2 \\
\leq& \left(\left\| \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b} \right\| + \sum_{i=1}^{n} \left\| \mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) \right\| \right)^2 \\
\leq& (n+1) \left(\left\| \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b} \right\|^2 + \sum_{i=1}^{n} \|\mathcal{A}_i\|^2 \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\|^2 \right) \\
\leq& (n+1) \left(\beta_k^{-2} \left\| \lambda_k - \hat{\lambda}_k \right\|^2 + \max\left\{ \frac{\|\mathcal{A}_i\|^2}{\eta_i - n\|\mathcal{A}_i\|^2} \right\} \sum_{i=1}^{n} \left(\eta_i - n\|\mathcal{A}_i\|^2 \right) \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\|^2 \right) \\
\leq& (n+1) \max\left\{ 1, \left\{ \frac{\|\mathcal{A}_i\|^2}{\eta_i - n\|\mathcal{A}_i\|^2} \right\} \right\} \left(\beta_k^{-2} \left\| \lambda_k - \hat{\lambda}_k \right\|^2 + \sum_{i=1}^{n} \left(\eta_i - n\|\mathcal{A}_i\|^2 \right) \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\|^2 \right) \\
=& \alpha^{-1} \left(\beta_k^{-2} \left\| \lambda_k - \hat{\lambda}_k \right\|^2 + \sum_{i=1}^{n} \left(\eta_i - n\|\mathcal{A}_i\|^2 \right) \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\|^2 \right).
\end{aligned} \tag{104}$$

By Proposition 2, we have

$$\begin{aligned}
&\beta_k^{-1} \sum_{i=1}^{n} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{u}_i^k) + \mathcal{A}_i^\dagger(\lambda^*) \right\rangle \\
&+ \frac{1}{2} \sum_{i=1}^{n} \left(\eta_i - n\|\mathcal{A}_i\|^2 \right) \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\|^2 + \frac{1}{2} \beta_k^{-2} \|\lambda^k - \hat{\lambda}^k\|^2 \\
\leq& \frac{1}{2} \left(\sum_{i=1}^{n} \eta_i \left\| \mathbf{x}_i^k - \mathbf{x}_i^* \right\|^2 + \beta_k^{-2} \left\| \lambda^k - \lambda^* \right\|^2 \right) \\
&- \frac{1}{2} \left(\sum_{i=1}^{n} \eta_i \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^* \right\|^2 + \beta_{k+1}^{-2} \left\| \lambda^{k+1} - \lambda^* \right\|^2 \right).
\end{aligned} \tag{105}$$

So by Lemma 2 and combining the above inequalities, we have

$$\begin{aligned}
&\beta_k^{-1}\left(f(\mathbf{x}^{k+1})-f(\mathbf{x}^*)+\sum_{i=1}^n\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,\mathcal{A}_i^\dagger(\lambda^*)\right\rangle+\frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2\right)\\
\le&\ \beta_k^{-1}\left(\sum_{i=1}^n\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,-\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)\right\rangle+\sum_{i=1}^n\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,\mathcal{A}_i^\dagger(\lambda^*)\right\rangle\right)\\
&+\frac{\alpha}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2\\
\le&\ \beta_k^{-1}\sum_{i=1}^n\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,-\sigma_i^{(k)}(\mathbf{x}_i^{k+1}-\mathbf{u}_i^k)+\mathcal{A}_i^\dagger(\lambda^*)\right\rangle\\
&+\frac{1}{2}\beta_k^{-2}\left\|\lambda_k-\hat{\lambda}_k\right\|^2+\frac{1}{2}\sum_{i=1}^n\left(\eta_i-n\left\|\mathcal{A}_i\right\|^2\right)\left\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\right\|^2\\
\le&\ \frac{1}{2}\left(\sum_{i=1}^n\eta_i\left\|\mathbf{x}_i^k-\mathbf{x}_i^*\right\|^2+\beta_k^{-2}\left\|\lambda^k-\lambda^*\right\|^2\right)\\
&-\frac{1}{2}\left(\sum_{i=1}^n\eta_i\left\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\right\|^2+\beta_{k+1}^{-2}\left\|\lambda^{k+1}-\lambda^*\right\|^2\right).
\end{aligned}\tag{106}$$

Here we use the fact that $\beta_k\ge\beta_0$, which is guaranteed by (18) and (20). Summing the above inequalities from $k=0$ to K, and dividing both sides with $\sum_{k=0}^K\beta_k^{-1}$, we have

$$\sum_{k=0}^K\gamma_k f(\mathbf{x}^{k+1})-f(\mathbf{x}^*)+\sum_{i=1}^n\left\langle\sum_{k=0}^K\gamma_k\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,\mathcal{A}_i^\dagger(\lambda^*)\right\rangle \tag{107}$$

$$+\frac{\alpha\beta_0}{2}\sum_{k=0}^K\gamma_k\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2 \tag{108}$$

$$\le\frac{1}{2\sum_{k=0}^K\beta_k^{-1}}\left(\sum_{i=1}^n\eta_i\left\|\mathbf{x}_i^0-\mathbf{x}_i^*\right\|^2+\beta_0^{-2}\left\|\lambda^0-\lambda^*\right\|^2\right). \tag{109}$$

Next, by the convexity of f and the squared Frobenius norm $\|\cdot\|^2$, we have

$$f(\bar{\mathbf{x}}^K)-f(\mathbf{x}^*)+\sum_{i=1}^n\left\langle\bar{\mathbf{x}}_i^K-\mathbf{x}_i^*,\mathcal{A}_i^\dagger(\lambda^*)\right\rangle+\frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\bar{\mathbf{x}}_i^K)-\mathbf{b}\right\|^2 \tag{110}$$

$$\le\sum_{k=0}^K\gamma_k f(\mathbf{x}^{k+1})-f(\mathbf{x}^*)+\sum_{i=1}^n\left\langle\sum_{k=0}^K\gamma_k\mathbf{x}_i^{k+1}-\mathbf{x}_i^*,\mathcal{A}_i^\dagger(\lambda^*)\right\rangle \tag{111}$$

$$+\frac{\alpha\beta_0}{2}\sum_{k=0}^K\gamma_k\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2. \tag{112}$$

Combining (107)-(109) and (110)-(112), we have

$$f(\bar{\mathbf{x}}^K)-f(\mathbf{x}^*)+\sum_{i=1}^n\left\langle\bar{\mathbf{x}}_i^K-\mathbf{x}_i^*,\mathcal{A}_i^\dagger(\lambda^*)\right\rangle+\frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\bar{\mathbf{x}}_i^K)-\mathbf{b}\right\|^2 \tag{113}$$

$$\le\frac{1}{2\sum_{k=0}^K\beta_k^{-1}}\left(\sum_{i=1}^n\eta_i\left\|\mathbf{x}_i^0-\mathbf{x}_i^*\right\|^2+\beta_0^{-2}\left\|\lambda^0-\lambda^*\right\|^2\right). \tag{114}$$

□

E Proof of Theorem 5

We only need to prove the following proposition. Then by the same technique for proving Theorem 1, we can prove Theorem 5.

Proposition 5 *For $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_{2n}^k, \lambda^k)\}$ generated by Algorithm 2 and a KKT point $(\mathbf{x}_1^*, \cdots, \mathbf{x}_{2n}^*, \lambda^*$ of problem (27), we have that*

$$\beta_k \sum_{i=1}^{2n} \sigma_i^{(k)} \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\|^2 + \|\lambda^{k+1} - \lambda^*\|^2 \tag{115}$$

$$\leq \beta_k \sum_{i=1}^{n} \sigma_i^{(k)} \|\mathbf{x}_i^k - \mathbf{x}_i^*\|^2 + \|\lambda^k - \lambda^*\|^2 \tag{116}$$

$$-2\beta_k \sum_{i=1}^{2n} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, -\sigma_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{u}_i^k) + \hat{\mathcal{A}}_i^\dagger(\lambda^*) \right\rangle \tag{117}$$

$$-\beta_k \sum_{i=1}^{n} \left(\sigma_i^{(k)} - \beta_k(n\|\mathcal{A}_i\|^2 + 2)\right) \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 \tag{118}$$

$$-\beta_k \sum_{i=n+1}^{2n} \left(\sigma_i^{(k)} - 2\beta_k\right) \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 \tag{119}$$

$$-\|\lambda^k - \hat{\lambda}^k\|^2. \tag{120}$$

Proof We continue from (72):

$$-2\beta_k \sum_{i=1}^{2n} \sigma_i^{(k)} \left\langle \mathbf{x}_i^{k+1} - \mathbf{x}_i^*, \mathbf{x}_i^k - \mathbf{u}_i^k \right\rangle + 2\left\langle \lambda^{k+1} - \lambda^k, \lambda^{k+1} \right\rangle \tag{121}$$

$$= \|\lambda^{k+1} - \lambda^k\|^2 + \beta_k^2 \left\| \sum_{i=1}^{2n} \hat{\mathcal{A}}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) \right\|^2 - \|\lambda^k - \hat{\lambda}^k\|^2 \tag{122}$$

$$= \|\lambda^{k+1} - \lambda^k\|^2 + \beta_k^2 \left\| \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) \right\|^2 \tag{123}$$

$$+\beta_k^2 \sum_{i=1}^{n} \left\| (\mathbf{x}_i^{k+1} - \mathbf{x}_i^k) - (\mathbf{x}_{n+i}^{k+1} - \mathbf{x}_{n+i}^k) \right\|^2 - \|\lambda^k - \hat{\lambda}^k\|^2 \tag{124}$$

$$\leq \|\lambda^{k+1} - \lambda^k\|^2 + n\beta_k^2 \sum_{i=1}^{n} \|\mathcal{A}_i\|^2 \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 \tag{125}$$

$$+2\beta_k^2 \sum_{i=1}^{n} \left(\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \|\mathbf{x}_{n+i}^{k+1} - \mathbf{x}_{n+i}^k\|^2 \right) - \|\lambda^k - \hat{\lambda}^k\|^2. \tag{126}$$

Then we can have (115)-(120). □

F Proof of Theorem 6

To prove Theorem 6, we need the following proposition:

Proposition 6 *For $\{(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)\}$ generated by Algorithm 3 and a KKT point $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*)$ of problem (1) with f_i described in Section 5, we have that*

$$\sum_{i=1}^n \left(f_i(\mathbf{x}_i^{k+1}) - f_i(\mathbf{x}_i^*) + \left\langle \mathcal{A}_i^\dagger(\lambda^*), \mathbf{x}_i^{k+1} - \mathbf{x}_i^* \right\rangle \right) \tag{127}$$

$$\leq \frac{1}{2} \sum_{i=1}^n \tau_i^{(k)} \left(\|\mathbf{x}_i^k - \mathbf{x}_i^*\|^2 - \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\|^2 \right) + \frac{1}{2\beta_k} \left(\|\lambda^k - \lambda^*\|^2 - \|\lambda^{k+1} - \lambda^*\|^2 \right) \tag{128}$$

$$-\frac{1}{2} \sum_{i=1}^n \left(\tau_i^{(k)} - L_i - n\beta_k \|\mathcal{A}_i\|^2 \right) \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 - \frac{1}{2\beta_k} \|\hat{\lambda}^k - \lambda^k\|^2 \tag{129}$$

Proof It can be observed that

$$0 \in \partial h_i(\mathbf{x}_i^{k+1}) + \nabla g_i(\mathbf{x}_i^k) + \mathcal{A}_i^\dagger(\hat{\lambda}^k) + \tau_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k).$$

So we have

$$h_i(\mathbf{x}_i) - h_i(\mathbf{x}_i^{k+1}) \geq \left\langle -\nabla g_i(\mathbf{x}_i^k) - \mathcal{A}_i^\dagger(\hat{\lambda}^k) - \tau_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k), \mathbf{x}_i - \mathbf{x}_i^{k+1} \right\rangle, \quad \forall \mathbf{x}_i,$$

and

$$\begin{aligned}
&\sum_{i=1}^n f_i(\mathbf{x}_i^{k+1}) = \sum_{i=1}^n \left(h_i(\mathbf{x}_i^{k+1}) + g_i(\mathbf{x}_i^{k+1}) \right) \\
&\leq \sum_{i=1}^n \left(h_i(\mathbf{x}_i^{k+1}) + g_i(\mathbf{x}_i^k) + \left\langle \nabla g_i(\mathbf{x}_i^k), \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\rangle + \frac{L_i}{2} \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 \right) \\
&= \sum_{i=1}^n \left(h_i(\mathbf{x}_i^{k+1}) + g_i(\mathbf{x}_i^k) + \left\langle \nabla g_i(\mathbf{x}_i^k), \mathbf{x}_i - \mathbf{x}_i^k \right\rangle + \left\langle \nabla g_i(\mathbf{x}_i^k), \mathbf{x}_i^{k+1} - \mathbf{x}_i \right\rangle \right. \\
&\quad \left. + \frac{L_i}{2} \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 \right) \\
&\leq \sum_{i=1}^n \left(g_i(\mathbf{x}_i) + h_i(\mathbf{x}_i) + \left\langle \mathcal{A}_i^\dagger(\hat{\lambda}^k) + \tau_i^{(k)}(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k), \mathbf{x}_i - \mathbf{x}_i^{k+1} \right\rangle + \frac{L_i}{2} \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 \right).
\end{aligned}$$

On the one hand,

$$\sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i)+\left\langle\mathcal{A}_i^{\dagger}(\hat{\lambda}^k),\mathbf{x}_i^{k+1}-\mathbf{x}_i\right\rangle\right)-\left\langle\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b},\hat{\lambda}^k-\lambda\right\rangle \tag{130}$$

$$\leq\sum_{i=1}^{n}\left(-\tau_i^{(k)}\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^k,\mathbf{x}_i^{k+1}-\mathbf{x}_i\right\rangle+\frac{L_i}{2}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2\right)-\left\langle\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b},\hat{\lambda}^k-\lambda\right\rangle$$

$$=\sum_{i=1}^{n}\left(-\tau_i^{(k)}\left\langle\mathbf{x}_i^{k+1}-\mathbf{x}_i^k,\mathbf{x}_i^{k+1}-\mathbf{x}_i\right\rangle+\frac{L_i}{2}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2\right)-\frac{1}{\beta_k}\left\langle\lambda^{k+1}-\lambda^k,\hat{\lambda}^k-\lambda\right\rangle$$

$$=\sum_{i=1}^{n}\left[\frac{\tau_i^{(k)}}{2}\left(\|\mathbf{x}_i^k-\mathbf{x}_i\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2\right)+\frac{L_i}{2}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2\right]$$
$$-\frac{1}{2\beta_k}\left(\|\lambda^{k+1}-\lambda\|^2-\|\lambda^k-\lambda\|^2+\|\hat{\lambda}^k-\lambda^k\|^2-\|\lambda^{k+1}-\hat{\lambda}^k\|^2\right)$$

$$=\sum_{i=1}^{n}\left[\frac{\tau_i^{(k)}}{2}\left(\|\mathbf{x}_i^k-\mathbf{x}_i\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2\right)+\frac{L_i}{2}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2\right]$$
$$-\frac{1}{2\beta_k}\left(\|\lambda^{k+1}-\lambda\|^2-\|\lambda^k-\lambda\|^2+\|\hat{\lambda}^k-\lambda^k\|^2-\beta_k^2\left\|\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1}-\mathbf{x}_i^k)\right\|^2\right)$$

$$\leq\frac{1}{2}\sum_{i=1}^{n}\tau_i^{(k)}\left(\|\mathbf{x}_i^k-\mathbf{x}_i\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i\|^2\right)-\frac{1}{2}\sum_{i=1}^{n}\left(\tau_i^{(k)}-L_i-n\beta_k\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2$$
$$+\frac{1}{2\beta_k}\left(\|\lambda^k-\lambda\|^2-\|\lambda^{k+1}-\lambda\|^2-\|\hat{\lambda}^k-\lambda^k\|^2\right). \tag{131}$$

On the other hand,

$$\sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i)+\left\langle\mathcal{A}_i^{\dagger}(\lambda),\mathbf{x}_i^{k+1}-\mathbf{x}_i\right\rangle\right)-\left\langle\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i)-\mathbf{b},\hat{\lambda}^k-\lambda\right\rangle$$
$$=\sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i)+\left\langle\mathcal{A}_i^{\dagger}(\hat{\lambda}^k),\mathbf{x}_i^{k+1}-\mathbf{x}_i\right\rangle\right)-\left\langle\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b},\hat{\lambda}^k-\lambda\right\rangle.$$

So we have

$$\sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i)+\left\langle\mathcal{A}_i^{\dagger}(\lambda),\mathbf{x}_i^{k+1}-\mathbf{x}_i\right\rangle\right)-\left\langle\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i)-\mathbf{b},\hat{\lambda}^k-\lambda\right\rangle$$
$$\leq\frac{1}{2}\sum_{i=1}^{n}\tau_i^{(k)}\left(\|\mathbf{x}_i^k-\mathbf{x}_i\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i\|^2\right)$$
$$-\frac{1}{2}\sum_{i=1}^{n}\left(\tau_i^{(k)}-L_i-n\beta_k\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2$$
$$+\frac{1}{2\beta_k}\left(\|\lambda^k-\lambda\|^2-\|\lambda^{k+1}-\lambda\|^2-\|\hat{\lambda}^k-\lambda^k\|^2\right).$$

Let $\mathbf{x}_i=\mathbf{x}_i^*$ and $\lambda=\lambda^*$, we have

$$\sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i^*)+\left\langle\mathcal{A}_i^{\dagger}(\lambda^*),\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\right\rangle\right)$$
$$\leq\frac{1}{2}\sum_{i=1}^{n}\tau_i^{(k)}\left[\|\mathbf{x}_i^k-\mathbf{x}_i^*\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\|^2\right]+\frac{1}{2\beta_k}\left(\|\lambda^k-\lambda^*\|^2-\|\lambda^{k+1}-\lambda^*\|^2\right)$$
$$-\frac{1}{2}\sum_{i=1}^{n}\left(\tau_i^{(k)}-L_i-n\beta_k\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^k\|^2-\frac{1}{2\beta_k}\|\hat{\lambda}^k-\lambda^k\|^2.$$

Proof **(of Theorem 6)** As $\mathbf{x}^*$ minimizes $\sum_{i=1}^n f(\mathbf{x}_i) + \left\langle \lambda^*, \sum_{i=1}^n \mathcal{A}_i(\mathbf{x}_i) - \mathbf{b} \right\rangle$, we have

$$0 \le \sum_{i=1}^n \left(f_i(\mathbf{x}_i^{k+1}) - f_i(\mathbf{x}_i^*) + \left\langle \mathcal{A}_i^\dagger(\lambda^*), \mathbf{x}_i^{k+1} - \mathbf{x}_i^* \right\rangle \right).$$

By Proposition 6, we have

$$\begin{aligned}
&\sum_{i=1}^n \frac{1}{2}\left(\tau_i^{(k)} - L_i - n\beta_k\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \frac{1}{2\beta_k}\|\hat{\lambda}^k - \lambda^k\|^2 \\
\le\ & \frac{1}{2}\sum_{i=1}^n \tau_i^{(k)}\left(\|\mathbf{x}_i^k - \mathbf{x}_i^*\|^2 - \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\|^2\right) + \frac{1}{2\beta_k}\left(\|\lambda^k - \lambda^*\|^2 - \|\lambda^{k+1} - \lambda^*\|^2\right).
\end{aligned}$$

Dividing both sides by β_k and using $\tau_i^{(k)} - L_i - n\beta_k\|\mathcal{A}_i\|^2 \ge \beta_k(\eta_i - n\|\mathcal{A}_i\|^2)$, the non-decrement of β_k and the non-increment of $\beta_k^{-1}\tau_i^{(k)}$, we have

$$\begin{aligned}
&\frac{1}{2}\sum_{i=1}^n \left(\eta_i - n\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \frac{1}{2\beta_k^2}\|\hat{\lambda}^k - \lambda^k\|^2 \qquad (132)\\
\le\ & \frac{1}{2}\sum_{i=1}^n \left(\beta_k^{-1}\tau_i^{(k)}\|\mathbf{x}_i^k - \mathbf{x}_i^*\|^2 - \beta_{k+1}^{-1}\tau_i^{(k+1)}\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\|^2\right) \\
&+ \left(\frac{1}{2\beta_k^2}\|\lambda^k - \lambda^*\|^2 - \frac{1}{2\beta_{k+1}^2}\|\lambda^{k+1} - \lambda^*\|^2\right). \qquad (133)
\end{aligned}$$

It can be easily seen that $(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k)$ is bounded, hence has an accumulation point, say $(\mathbf{x}_1^{k_j}, \cdots, \mathbf{x}_n^{k_j}, \lambda^{k_j}) \to (\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$.

Summing (132)-(133) over $k = 0, \cdots, \infty$, we have

$$\begin{aligned}
&\frac{1}{2}\sum_{i=1}^n \left(\eta_i - n\|\mathcal{A}_i\|^2\right)\sum_{k=0}^\infty \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \sum_{k=0}^\infty \frac{1}{2\beta_k^2}\|\hat{\lambda}^k - \lambda^k\|^2 \\
\le\ & \frac{1}{2}\sum_{i=1}^n \beta_0^{-1}\tau_i^{(0)}\|\mathbf{x}_i^0 - \mathbf{x}_i^*\|^2 + \frac{1}{2\beta_0^2}\|\lambda^0 - \lambda^*\|^2.
\end{aligned}$$

So $\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\| \to 0$ and $\beta_k^{-2}\|\hat{\lambda}^k - \lambda^k\| \to 0$ as $k \to \infty$. Hence $\left\|\sum_{i=1}^n \mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b}\right\| \to 0$, which means that $\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty$ is a feasible solution.

From (130)-(131), we have

$$\begin{aligned}
&\sum_{i=1}^n \left(f_i(\mathbf{x}_i^{k_j+1}) - f_i(\mathbf{x}_i) + \left\langle \mathcal{A}_i^\dagger(\hat{\lambda}^{k_j}), \mathbf{x}_i^{k_j+1} - \mathbf{x}_i \right\rangle\right) - \left\langle \sum_{i=1}^n \mathcal{A}_i(\mathbf{x}_i^{k_j+1}) - \mathbf{b}, \hat{\lambda}^{k_j} - \lambda \right\rangle \\
\le\ & \frac{1}{2}\sum_{i=1}^n \tau_i^{(k_j)}\left(\|\mathbf{x}_i^{k_j} - \mathbf{x}_i\|^2 - \|\mathbf{x}_i^{k_j+1} - \mathbf{x}_i\|^2\right) \\
&-\frac{1}{2}\sum_{i=1}^n \left(\tau_i^{(k_j)} - L_i - n\beta_{k_j}\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k_j+1} - \mathbf{x}_i^{k_j}\|^2 \\
&+\frac{1}{2\beta_{k_j}}\left(\|\lambda^{k_j} - \lambda\|^2 - \|\lambda^{k_j+1} - \lambda\|^2 - \|\hat{\lambda}^{k_j} - \lambda^{k_j}\|^2\right).
\end{aligned}$$

Let $j \to \infty$. By the boundedness of $\tau_i^{(k_j)}$ we have

$$\sum_{i=1}^n \left(f_i(\mathbf{x}_i^\infty) - f_i(\mathbf{x}_i) + \left\langle \mathcal{A}_i^\dagger(\lambda^\infty), \mathbf{x}_i^\infty - \mathbf{x}_i \right\rangle\right) \le 0, \quad \forall \mathbf{x}_i.$$

Together with the feasibility of $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty)$, we can see that $(\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$ is a KKT point.

By choosing $(\mathbf{x}_1^*, \cdots, \mathbf{x}_n^*, \lambda^*) = (\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$ we have

$$\sum_{i=1}^{n} \eta_i \|\mathbf{x}_i^{k_j} - \mathbf{x}_i^\infty\|^2 + \frac{1}{\beta_{k_j}^2} \|\lambda^{k_j} - \lambda^\infty\|^2 \to 0.$$

Using (132)-(133), we have

$$\sum_{i=1}^{n} \eta_i \|\mathbf{x}_i^{k} - \mathbf{x}_i^\infty\|^2 + \frac{1}{\beta_{k}^2} \|\lambda^{k} - \lambda^\infty\|^2 \to 0.$$

So $(\mathbf{x}_1^k, \cdots, \mathbf{x}_n^k, \lambda^k) \to (\mathbf{x}_1^\infty, \cdots, \mathbf{x}_n^\infty, \lambda^\infty)$.

G Proof of Theorem 7

Proof **(of Theorem 7)** By the definition of α and $\tau_i^{(k)}$,

$$\begin{aligned}
&\frac{1}{2}\left[\sum_{i=1}^{n}\left(\tau_i^{(k)} - L_i - n\beta_k\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \frac{1}{\beta_k}\|\hat{\lambda}^k - \lambda^k\|^2\right] \qquad (134)\\
\geq\ & \frac{\beta_k}{2}\left[\sum_{i=1}^{n}\left(\eta_i - n\|\mathcal{A}_i\|^2\right)\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \frac{1}{\beta_k^2}\|\hat{\lambda}^k - \lambda^k\|^2\right]\\
\geq\ & \frac{\alpha\beta_k}{2}(n+1)\left(\sum_{i=1}^{n}\|\mathcal{A}_i\|^2\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\|^2 + \frac{1}{\beta_k^2}\|\hat{\lambda}^k - \lambda^k\|^2\right)\\
\geq\ & \frac{\alpha\beta_k}{2}(n+1)\left(\sum_{i=1}^{n}\|\mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k)\|^2 + \frac{1}{\beta_k^2}\|\hat{\lambda}^k - \lambda^k\|^2\right)\\
=\ & \frac{\alpha\beta_k}{2}(n+1)\left(\sum_{i=1}^{n}\|\mathcal{A}_i(\mathbf{x}_i^{k+1} - \mathbf{x}_i^k)\|^2 + \left\|\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^k) - \mathbf{b}\right\|^2\right)\\
\geq\ & \frac{\alpha\beta_k}{2}\left\|\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b}\right\|^2. \qquad (135)
\end{aligned}$$

So by (127)-(129) and the non-decrement of β_k, we have

$$\begin{aligned}
&\sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1}) - f_i(\mathbf{x}_i^*) + \left\langle \mathcal{A}_i^\dagger(\lambda^*), \mathbf{x}_i^{k+1} - \mathbf{x}_i^*\right\rangle\right) + \frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b}\right\|^2 \qquad (136)\\
\leq\ & \sum_{i=1}^{n}\left(f_i(\mathbf{x}_i^{k+1}) - f_i(\mathbf{x}_i^*) + \left\langle \mathcal{A}_i^\dagger(\lambda^*), \mathbf{x}_i^{k+1} - \mathbf{x}_i^*\right\rangle\right) + \frac{\alpha\beta_k}{2}\left\|\sum_{i=1}^{n}\mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b}\right\|^2 \qquad (137)\\
\leq\ & \frac{1}{2}\sum_{i=1}^{n}\tau_i^{(k)}\left(\|\mathbf{x}_i^k - \mathbf{x}_i^*\|^2 - \|\mathbf{x}_i^{k+1} - \mathbf{x}_i^*\|^2\right) + \frac{1}{2\beta_k}\left(\|\lambda^k - \lambda^*\|^2 - \|\lambda^{k+1} - \lambda^*\|^2\right). \qquad (138)
\end{aligned}$$

Dividing both sides by β_k and using the non-decrement of β_k and the non-increment of $\beta_k^{-1}\tau_i^{(k)}$, we have

$$\frac{1}{\beta_k}\left[\sum_{i=1}^n\left(f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i^*)+\left\langle\mathcal{A}_i^\dagger(\lambda^*),\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\right\rangle\right)+\frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2\right] \tag{139}$$

$$\leq\frac{1}{2}\sum_{i=1}^n\beta_k^{-1}\tau_i^{(k)}\left(\|\mathbf{x}_i^k-\mathbf{x}_i^*\|^2-\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\|^2\right)+\frac{1}{2\beta_k^2}\left(\|\lambda^k-\lambda^*\|^2-\|\lambda^{k+1}-\lambda^*\|^2\right)$$

$$\leq\frac{1}{2}\sum_{i=1}^n\left(\beta_k^{-1}\tau_i^{(k)}\|\mathbf{x}_i^k-\mathbf{x}_i^*\|^2-\beta_{k+1}^{-1}\tau_i^{(k+1)}\|\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\|^2\right)$$

$$+\left(\frac{1}{2\beta_k^2}\|\lambda^k-\lambda^*\|^2-\frac{1}{2\beta_{k+1}^2}\|\lambda^{k+1}-\lambda^*\|^2\right). \tag{140}$$

Summing over $k=0,\cdots,K$ and dividing both sides by $\sum_{k=0}^K\beta_k^{-1}$, we have

$$\sum_{i=1}^n\left(\sum_{k=0}^K\gamma^k f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i^*)+\left\langle\mathcal{A}_i^\dagger(\lambda^*),\sum_{k=0}^K\gamma^k\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\right\rangle\right) \tag{141}$$

$$+\frac{\alpha\beta_0}{2}\sum_{k=0}^K\gamma^k\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2 \tag{142}$$

$$\leq\left(\sum_{i=1}^n\beta_0^{-1}\tau_i^{(0)}\|\mathbf{x}_i^0-\mathbf{x}_i^*\|^2+\beta_0^{-2}\|\lambda^0-\lambda^*\|^2\right)/\sum_{k=0}^K2\beta_k^{-1}. \tag{143}$$

Using the convexity of f_i and $\|\cdot\|^2$, we have

$$\sum_{i=1}^n\left(f_i(\bar{\mathbf{x}}_i^K)-f_i(\mathbf{x}_i^*)+\left\langle\mathcal{A}_i^\dagger(\lambda^*),\bar{\mathbf{x}}_i^K-\mathbf{x}_i^*\right\rangle\right)+\frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\bar{\mathbf{x}}_i^K)-\mathbf{b}\right\|^2 \tag{144}$$

$$\leq\sum_{i=1}^n\left(\sum_{k=0}^K\gamma^k f_i(\mathbf{x}_i^{k+1})-f_i(\mathbf{x}_i^*)+\left\langle\mathcal{A}_i^\dagger(\lambda^*),\sum_{k=0}^K\gamma^k\mathbf{x}_i^{k+1}-\mathbf{x}_i^*\right\rangle\right) \tag{145}$$

$$+\frac{\alpha\beta_0}{2}\sum_{k=0}^K\gamma^k\left\|\sum_{i=1}^n\mathcal{A}_i(\mathbf{x}_i^{k+1})-\mathbf{b}\right\|^2. \tag{146}$$

So we have

$$\sum_{i=1}^n\left(f_i(\bar{\mathbf{x}}_i^K)-f_i(\mathbf{x}_i^*)+\left\langle\mathcal{A}_i^\dagger(\lambda^*),\bar{\mathbf{x}}_i^K-\mathbf{x}_i^*\right\rangle\right)+\frac{\alpha\beta_0}{2}\left\|\sum_{i=1}^n\mathcal{A}_i(\bar{\mathbf{x}}_i^K)-\mathbf{b}\right\|^2 \tag{147}$$

$$\leq\left(\sum_{i=1}^n\beta_0^{-1}\tau_i^{(0)}\|\mathbf{x}_i^0-\mathbf{x}_i^*\|^2+\beta_0^{-2}\|\lambda^0-\lambda^*\|^2\right)/\sum_{k=0}^K2\beta_k^{-1}. \tag{148}$$

References

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

Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press

Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2011) Distributed optimization and statistical learning via the alternating direction method of multipliers. In: Jordan M (ed) Foundations and Trends in Machine Learning

Cai J, Candès E, Shen Z (2010) A singular value thresholding algorithm for matrix completion. SIAM J Optimization 20(4):1956–1982

Candès E, Recht B (2009) Exact matrix completion via convex optimization. Foundations of Computational Mathematics 9(6):717–772

Candès E, Li X, Ma Y, Wright J (2011) Robust principal component analysis? J ACM 58(3):No.11

Chandrasekaran V, Parrilo P, Willsky A (2012) Latent variable graphical model selection via convex optimization. The Annals of Statistics 40(4):1935–1967

Chang E (2011) Foundations of Large-Scale Multimedia Information Management and Retrieval: Mathematics of Perception. Springer-Verlag New York Inc

Chang E, Zhu K, Wang H, Bai H, Li J, Qiu Z, Cui H (2007) Psvm: Parallelizing support vector machines on distributed computers. In: NIPS

Chen C, He B, Ye Y, Yuan X (2013) The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. Preprint

Deng W, Yin W (2012) On the global and linear convergence of the generalized alternating direction method of multipliers. Tech. rep., DTIC Document

Deng W, Yin W, Zhang Y (2011) Group sparse optimization by alternating direction method. TR11-06, Department of Computational and Applied Mathematics, Rice University

Esser E (2009) Applications of Lagrangian-based alternating direction methods and connections to split Bregman. CAM Report 09-31, UCLA

Favaro P, Vidal R, Ravichandran A (2011) A closed form solution to robust subspace estimation and clustering. In: CVPR

Fazel M (2002) Matrix rank minimization with applications. PhD thesis

Fortin M, Glowinski R (1983) Augmented Lagrangian methods. North-Holland

Goldfarb D, Ma S (2012) Fast multiple splitting algorithms for convex optimization. SIAM J Optimization 22(2):533–556

Goldstein T, Osher S (2008) The split Bregman method for ℓ_1 regularized problems. SIAM J Imaging Sciences 2(2):323–343

He B, Yuan X (2012) On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. SIAM J Numerical Analysus 50(2):700–709

He B, Yuan X (2013) Linearized alternating direction method with Gaussian back substitution for separable convex programming. Numerical Algebra, Control and Optimization 3(2):247–260

He B, Tao M, Yuan X (2012) Alternating direction method with Gaussian back substitution for separable convex programming. SIAM J Optimization 22(2):313–340

Hong M, Luo ZQ (2012) On the linear convergence of the alternating direction method of multipliers. Preprint, arXiv:12083922

Jacob L, Obozinski G, Vert J (2009) Group Lasso with overlap and graph Lasso. In: ICML

Ji H, Liu C, Shen Z, Xu Y (2010) Robust video denoising using low rank matrix completion. In: CVPR

Lin Z, Chen M, Ma Y (2009a) The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. UIUC Technical Report UILU-ENG-09-2215

Lin Z, Ganesh A, Wright J, Wu L, Chen M, Ma Y (2009b) Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. UIUC Technical Report UILU-ENG-09-2214

Lin Z, Liu R, Su Z (2011) Linearized alternating direction method with adaptive penalty for low-rank representation. In: NIPS

Liu G, Yan S (2011) Latent low-rank representation for subspace segmentation and feature extraction. In: ICCV

Liu G, Lin Z, Yu Y (2010) Robust subspace segmentation by low-rank representation. In: ICML

Liu G, Lin Z, Yan S, Sun J, Yu Y, Ma Y (2012) Robust recovery of subspace structures by low-rank representation. IEEE Trans on PAMI 35(1):171–184

Liu R, Lin Z, Su Z (2013, oral presentation) Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning. In: ACML

Ma S, Goldfarb D, Chen L (2011) Fixed point and bregman iterative methods for matrix rank minimization. Mathematical Programming 128(1-2):321–359

Meier L, Geer SVD, Bühlmann P (2008) The group Lasso for logistic regression. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70(1):53–71

Rockafellar R (1970) Convex Analysis. Princeton University Press

Shen X, Wu Y (2012) A unified approach to salient object detection via low rank matrix recovery. In: CVPR

Subramanian A, Tamayo P, Mootha V, Mukherjee S, et al (2005) Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. Proceedings of the National Academy of Sciences 102(43):267–288

Tao M (2014) Some parallel splitting methods for separable convex programming with $O(1/t)$ convergence rate. Pacific J. Optimization 10(2):359–384

Toh K, Yun S (2010) An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. Pacific J Optimization 6(15):615–640

Tron R, Vidal R (2007) A benchmark for the comparison of 3D montion segmentation algorithms. In: CVPR

van de Vijver M, He Y, van't Veer L, Dai H, et al (2002) A gene-expression signature as a predictor of survival in breast cancer. The New England Journal of Medicine 347(25):1999–2009

Wright J, Yang A, Ganesh A, Sastry S, Ma Y (2009) Robust face recognition via sparse representation. IEEE Trans on PAMI 31(2):210–227

Xu Y, Yin W, Wen Z (2011) An alternating direction algorithm for matrix completion with nonnegative factors. CAAM Technical Report TR11-03

Yang J, Yuan X (2013) Linearized augmented Lagrangian and alternating direction methods for nuclear norm minimization. Mathematics of Computation 82(281):301–329

Ye J, Ji S, Chen J (2008) Multi-class discriminant kernel learning via convex programming. JMLR 9:719–758

Zhang X, Burger M, Osher S (2011) A unified primal-dual algorithm framework based on Bregman iteration. Journal of Scientific Computing 46(1):20–46