HI, BCD! HYBRID INEXACT BLOCK COORDINATE DESCENT FOR HYPERSPECTRAL SUPER-RESOLUTION

†Ruiyuan Wu, †Chun-Hei Chan, ‡Hoi-To Wai, †Wing-Kin Ma, and ⋆Xiao Fu

†Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China
‡School of ECEE, Arizona State Univ., AZ, USA, ⋆ School of EECS, Oregon State Univ., OR, USA

ABSTRACT

Hyperspectral super-resolution (HSR) is a problem of recovering a high-spectral-spatial-resolution image from a multispectral measurement and a hyperspectral measurement, which have low spectral and spatial resolutions, respectively. We consider a low-rank structured matrix factorization formulation for HSR, which is a non-convex large-scale optimization problem. Our contributions contain both computational and theoretical aspects. On the computational side, we develop three inexact block coordinate descent (BCD) schemes that are empirically found to run many times faster than a state-of-the-art method, which uses exact BCD. We achieve this by applying concepts in the proximal gradient (PG) and Frank-Wolfe (FW) methods and by exploiting the HSR problem structures. On the theoretical side, we show that these inexact BCD schemes guarantee convergence to a stationary point. In particular, the convergence result for a hybrid PG-FW inexact BCD scheme is new.

***Index Terms*—** Hyperspectral super-resolution, inexact block coordinate descent, convergence analysis

1. INTRODUCTION

In this paper we address a super-resolution (SR) imaging problem arising from hyperspectral remote sensing. The problem is to reconstruct a high-spectral-spatial-resolution image from a low-spectral-resolution high-spatial-resolution measurement and a high-spectral-resolution low-spatial-resolution measurement, acquired from a multispectral (MS) sensor and a hyperspectral (HS) sensor, respectively (resp.). We call this problem hyperspectral super-resolution (HSR). Also called MS-HS data fusion in the literature, HSR is an exciting and relatively new direction. It enables us to accomplish HSR by computational techniques and using existing MS and HS sensors, rather than building high-spectral-spatial-resolution sensors which is known to be difficult [1–3].

There are several approaches to tackle the HSR problem [4, 5]. We will focus only on the low-rank structured matrix factorization (SMF) approach in this paper. There are a variety of SMF formulations and methods [6–10]; e.g., some use non-negative matrix factorization (NMF), some apply ℓ_1-sparse or total variation regularizations, to name a few. In this work we consider the SMF formulation used in [10]. It is a plain SMF model, with no regularization. It is based on the widely used linear mixture model in hyperspectral remote sensing. Despite its simplicity, this plain model has been found to work well in practice. The state of the art, called FUMI [10], adopts an exact block coordinate descent (BCD) strategy to tackle the corresponding SMF problem. It focuses on deriving fast solvers for the exact BCD updates; the technique used is ADMM. We follow a different strategy, namely, inexact BCD. The reason for us to consider this direction is as follows: HSR is a large-scale problem, and this means that it takes non-negligible computational time to execute one exact BCD update even though one has fast solvers at hand. Thus, can we attempt computationally cheaper, but inexact, BCD updates and see if this will lead to better computational efficiency?

In fact, the idea of inexact BCD is considered natural. Coupled NMF [7], one of the early SMF methods for HSR, is reminiscent of an inexact BCD in terms of ideas (although it is *not* BCD by principle). Inexact BCD and relevant ideas have recently received much interest in mathematical optimization [11–14]. Those studies tend to consider theoretical issues for a wide class of problems, and touch less on computational aspects which are often intimately related to structures of a specific problem. In this work we will develop three inexact BCD schemes—which use the proximal gradient (PG) method, the Frank-Wolfe (FW) method, and their hybrids to leverage on the underlying SMF problem structures for obtaining efficient inexact updates. Such inexact BCD schemes were not considered in the context of HSR, and our endeavor of handling the subsequent computational issues is new. We will also consider theoretical issues concerning convergence. As will be discussed further, our inexact BCD schemes guarantee convergence to a stationary point of the SMF problem. In particular, the convergence result for the hybrid inexact BCD scheme is novel.

Our notations are largely standard. In addition, the ith column of a matrix $\boldsymbol{X}$ is denoted by $\boldsymbol{x}_i$; $\lambda_{\max}(\boldsymbol{A})$ denote the largest eigenvalues of a matrix $\boldsymbol{A}$; $\boldsymbol{e}_i$ denotes a unit vector with $[\boldsymbol{e}_i]_i = 1$ and $[\boldsymbol{e}_i]_j = 0$ for all $j \neq i$; $\iota_{\mathcal{X}}$ denote the indicator function of a set $\mathcal{X}$, i.e., $\iota_{\mathcal{X}}(\boldsymbol{x}) = 0$ if $\boldsymbol{x} \in \mathcal{X}$ and $\iota_{\mathcal{X}}(\boldsymbol{x}) = \infty$ if $\boldsymbol{x} \notin \mathcal{X}$; $\Pi_{\mathcal{X}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}\in\mathcal{X}} \|\boldsymbol{z} - \boldsymbol{x}\|_2^2$ denotes the projection onto $\mathcal{X}$.

2. PROBLEM STATEMENT

Let $\boldsymbol{X} \in \mathbb{R}^{M\times L}$ be a spectral-spatial matrix of an image where M and L denote the number of spectral bands and pixels, resp., and x_{ij} records the spectral reflectance at spectral band i and pixel j. This image has high spectral and spatial resolutions, and we will call it an SR image in the sequel. The SR image is observed by an MS sensor and an HS sensor, which have low spectral and spatial resolutions, resp. This is illustrated in Fig. 1. Assuming that the measured MS and HS images are co-registered, they are modeled as

$$\boldsymbol{Y}_{\rm M} = \boldsymbol{F}\boldsymbol{X} + \boldsymbol{V}_{\rm M}, \qquad \boldsymbol{Y}_{\rm H} = \boldsymbol{X}\boldsymbol{G} + \boldsymbol{V}_{\rm H}, \tag{1}$$

where $\boldsymbol{Y}_{\rm M} \in \mathbb{R}^{M_{\rm M}\times L}$ and $\boldsymbol{Y}_{\rm H} \in \mathbb{R}^{M\times L_{\rm H}}$ denote the spectral-spatial matrices of the MS and HS images, resp.; $M_{\rm M} < M$ is the number of spectral bands of the MS image; $L_{\rm H} < L$ is the number of HS image pixels; $\boldsymbol{F} \in \mathbb{R}^{M_{\rm M}\times M}$ and $\boldsymbol{G} \in \mathbb{R}^{L\times L_{\rm H}}$ are given ma-

This work was supported by a SHIAE Research Fund (project no. #MMT 59/17), in part by a General Research Fund of Hong Kong Reseach Grant Council (project no. CUHK 14205414), and in part by the U.S. National Science Foundation under project NSF ECCS-1608961.

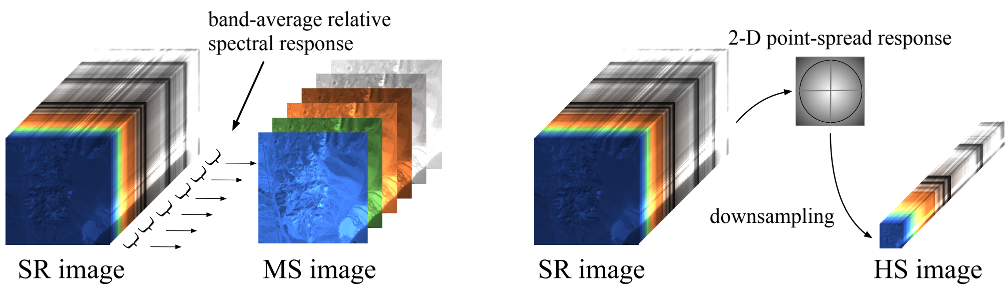

Fig. 1. Left: multispectral measurement of an SR image. Right: hyperspectral measurement of an SR image.

trices that model the spectral and spatial degradation effects of the MS and HS measurements, resp.; $\boldsymbol{V}_{\rm M}$ and $\boldsymbol{V}_{\rm H}$ are noise.

The problem is to estimate $\boldsymbol{X}$ from $\boldsymbol{Y}_{\rm M}$ and $\boldsymbol{Y}_{\rm H}$. To this end, we adopt the structured matrix factorization approach in which the SR image is assumed to follow a low-rank model

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S}, \tag{2}$$

where $\boldsymbol{A} \in \mathbb{R}^{M\times N}, \boldsymbol{S} \in \mathbb{R}^{N\times L}, N \ll \min\{M, L\}$. We also assume the widely used linear mixture model (LMM) in hyperspectral remote sensing, where $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N$ model the spectral signatures of endmembers (or materials) in the underlying scene, s_{ij} models the proportion of contribution of endmember i at pixel j, and N is the number of endmembers. In particular, $\boldsymbol{A}$ and $\boldsymbol{S}$ satisfy

$$\boldsymbol{A} \in \mathcal{A} := [0,1]^{M\times N}, \quad \boldsymbol{S} \in \mathcal{S} := \{\boldsymbol{S} \mid \boldsymbol{s}_i \in \mathcal{U}_N,\ i = 1, \ldots, L\},$$

where $\mathcal{U}_N = \{\boldsymbol{s} \mid \boldsymbol{s} \geq \boldsymbol{0}, \boldsymbol{1}^T\boldsymbol{s} = 1\}$ denotes the unit simplex on $\mathbb{R}^N$. Readers are referred to [10, 15] for the modeling details. Given a model order N, we intend to estimate $(\boldsymbol{A}, \boldsymbol{S})$, and thereby recover $\boldsymbol{X}$, by solving a data fitting problem

$$\min_{\boldsymbol{A}\in\mathcal{A}, \boldsymbol{S}\in\mathcal{S}} f(\boldsymbol{A}, \boldsymbol{S}) := \tfrac{1}{2}\|\boldsymbol{Y}_{\rm M} - \boldsymbol{F}\boldsymbol{A}\boldsymbol{S}\|_F^2 + \tfrac{1}{2}\|\boldsymbol{Y}_{\rm H} - \boldsymbol{A}\boldsymbol{S}\boldsymbol{G}\|_F^2. \tag{3}$$

Our interest will be centered on how Problem (3) is handled computationally.

A straightforward strategy for tackling Problem (3), at least by concepts, is to apply the following exact BCD

$$\boldsymbol{S}^{k+1} = \arg\min_{\boldsymbol{S}\in\mathcal{S}} f(\boldsymbol{A}^k, \boldsymbol{S}),\ \boldsymbol{A}^{k+1} = \arg\min_{\boldsymbol{A}\in\mathcal{A}} f(\boldsymbol{A}, \boldsymbol{S}^{k+1}) \tag{4}$$

for $k = 0, 1, \ldots$ and given a starting point $(\boldsymbol{A}^0, \boldsymbol{S}^0)$. It should be mentioned that each subproblem in (4) is convex. In [10] the authors developed an algorithm called FUMI, which uses custom-derived ADMM solvers to solve the subproblems in (4). However, we should note that the number of pixels L is often large in practice, and consequently solving the $\boldsymbol{S}$-update subproblem in (4) takes time even with custom-derived solvers.

3. INEXACT BCD FOR HSR

We endeavor to improve on the state of the art by taking on an inexact BCD strategy.

3.1. Inexact BCD by Proximal Gradient

Our first idea is to replace the exact BCD step (4) with proximal gradient (PG) updates. Let us first review some basic concepts. Let

$$\text{prox}_h(\boldsymbol{x}) := \arg\min_{\boldsymbol{z}\in\mathbb{R}^n} \tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + h(\boldsymbol{z})$$

define the proximal mapping of a function h. Consider an optimization problem in form of

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) + h(\boldsymbol{x}),$$

where f is smooth and has Lipschitz continuous gradient $\nabla f(\boldsymbol{x})$; h is convex, proper and closed. The PG iterations for handling the above problem is

$$\boldsymbol{x}^{k+1} = \text{prox}_{\gamma_k h}\left(\boldsymbol{x}^k - \gamma_k \nabla f(\boldsymbol{x}^k)\right), \quad k = 0, 1, 2, \ldots$$

where γ_k is the step size, and a standard step-size selection rule is to choose γ_k as the reciprocal of the Lipschitz constant of $\nabla f(\boldsymbol{x})$. For instances where one uses $h(\boldsymbol{x}) = \iota_{\mathcal{X}}(\boldsymbol{x})$ for some convex closed $\mathcal{X}$, we have

$$\text{prox}_{\gamma_k h}(\boldsymbol{x}) = \Pi_{\mathcal{X}}(\boldsymbol{x}),$$

i.e., the proximal mapping is the projection onto $\mathcal{X}$. Readers are referred to the literature for further details [16].

To see how PG works here, rewrite Problem (3) as

$$\min_{\boldsymbol{A}\in\mathbb{R}^{M\times N}, \boldsymbol{S}\in\mathbb{R}^{N\times L}} f(\boldsymbol{A}, \boldsymbol{S}) + \iota_{\mathcal{A}}(\boldsymbol{A}) + \iota_{\mathcal{S}}(\boldsymbol{S}).$$

Consider a combination of BCD and PG methods as follows:

$$\boldsymbol{S}^{k+1} = \text{prox}_{\gamma_{S,k}\iota_{\mathcal{S}}}\left(\boldsymbol{S}^k - \gamma_{S,k}\nabla_{\boldsymbol{S}} f(\boldsymbol{A}^k, \boldsymbol{S}^k)\right), \tag{5a}$$

$$\boldsymbol{A}^{k+1} = \text{prox}_{\gamma_{A,k}\iota_{\mathcal{A}}}\left(\boldsymbol{A}^k - \gamma_{A,k}\nabla_{\boldsymbol{A}} f(\boldsymbol{A}^k, \boldsymbol{S}^{k+1})\right), \tag{5b}$$

where $\gamma_{S,k}$ and $\gamma_{A,k}$ are step sizes; we will choose $\gamma_{S,k}$ as the reciprocal of the Lipschitz constant of $\nabla_{\boldsymbol{S}} f(\boldsymbol{A}^k, \boldsymbol{S})$ with respect to (w.r.t.) $\boldsymbol{S}$, and choose $\gamma_{A,k}$ as the reciprocal of the Lipschitz constant of $\nabla_{\boldsymbol{A}} f(\boldsymbol{A}, \boldsymbol{S}^{k+1})$ w.r.t. $\boldsymbol{A}$. The rationale is to replace the exact BCD updates by inexact ones, thereby attempting to improve computational efficiency.

The implementation details of (5) are as follows. We have

$$\nabla_{\boldsymbol{S}} f(\boldsymbol{A}, \boldsymbol{S}) = (\boldsymbol{F}\boldsymbol{A})^T(\boldsymbol{F}\boldsymbol{A}\boldsymbol{S} - \boldsymbol{Y}_{\rm M}) + \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{S}\boldsymbol{G} - \boldsymbol{Y}_{\rm H})\boldsymbol{G}^T,$$
$$\nabla_{\boldsymbol{A}} f(\boldsymbol{A}, \boldsymbol{S}) = \boldsymbol{F}^T(\boldsymbol{F}\boldsymbol{A}\boldsymbol{S} - \boldsymbol{Y}_{\rm M})\boldsymbol{S}^T + (\boldsymbol{A}\boldsymbol{S}\boldsymbol{G} - \boldsymbol{Y}_{\rm H})(\boldsymbol{S}\boldsymbol{G})^T.$$

The proximal mapping $\text{prox}_{\gamma_{A,k}\iota_{\mathcal{A}}}$ is simple: if we write $\boldsymbol{Z} = \text{prox}_{\gamma_{A,k}\iota_{\mathcal{A}}}(\boldsymbol{W})$, then $z_{ij} = \max\{0, \min\{w_{ij}, 1\}\}$ for all i, j. The proximal mapping $\text{prox}_{\gamma_{S,k}\iota_{\mathcal{S}}}$ is more complex computationally. If we write $\boldsymbol{Z} = \text{prox}_{\gamma_{S,k}\iota_{\mathcal{S}}}(\boldsymbol{W})$, then

$$\boldsymbol{z}_i = \Pi_{\mathcal{U}_N}(\boldsymbol{w}_i), \quad i = 1, \ldots, L.$$

The above unit simplex projections do not have a closed form, but they can be computed by an available algorithm [17] with a worst-case complexity of $\mathcal{O}(N\log(N))$. The Lipschitz constants required for determining the step sizes are described in the following lemma.

Lemma 1 *Suppose $\boldsymbol{A} \neq \mathbf{0}$.*[1] *The Lipschitz constant of $\nabla_{\boldsymbol{S}} f(\boldsymbol{A}, \boldsymbol{S})$ w.r.t. $\boldsymbol{S}$ is*

$$\lambda_{\max}\left(\theta_G \boldsymbol{A}^T\boldsymbol{A} + (\boldsymbol{F}\boldsymbol{A})^T(\boldsymbol{F}\boldsymbol{A})\right),$$

where $\theta_G = \lambda_{\max}(\boldsymbol{G}^T\boldsymbol{G})$. Similarly, suppose $\boldsymbol{S} \neq \mathbf{0}$. The Lipschitz constant of $\nabla_{\boldsymbol{A}} f(\boldsymbol{A}, \boldsymbol{S})$ w.r.t. $\boldsymbol{A}$ is

$$\lambda_{\max}\left(\theta_F \boldsymbol{S}\boldsymbol{S}^T + (\boldsymbol{S}\boldsymbol{G})(\boldsymbol{S}\boldsymbol{G})^T\right),$$

where $\theta_F = \lambda_{\max}(\boldsymbol{F}\boldsymbol{F}^T)$.

We skip the proof of Lemma 1 owing to space limitation. We should mention that Lemma 1 is not a standard result and its proof requires exploitation of the problem structures in (3). It is also an efficient solution to compute the Lipschitz constants.

3.2. Inexact BCD by the Frank-Wolfe Method

Our second idea has the same spirit as the previous PG inexact BCD scheme. The difference is that we replace the PG updates with the Frank-Wolfe (FW) method:

$$\boldsymbol{S}^{k+1} = \boldsymbol{S}^k + \alpha_{S,k}(\boldsymbol{P}_S^k - \boldsymbol{S}^k), \tag{6a}$$

$$\boldsymbol{A}^{k+1} = \boldsymbol{A}^k + \alpha_{A,k}(\boldsymbol{P}_A^k - \boldsymbol{A}^k), \tag{6b}$$

where $\alpha_{S,k}, \alpha_{A,k} \in (0, 1]$ are step sizes;

$$\boldsymbol{P}_S^k = \arg\min_{\boldsymbol{Z}\in\mathcal{S}} \langle \nabla_{\boldsymbol{S}} f(\boldsymbol{A}^k, \boldsymbol{S}^k), \boldsymbol{Z}\rangle, \tag{7a}$$

$$\boldsymbol{P}_A^k = \arg\min_{\boldsymbol{Z}\in\mathcal{A}} \langle \nabla_{\boldsymbol{A}} f(\boldsymbol{A}^k, \boldsymbol{S}^{k+1}), \boldsymbol{Z}\rangle \tag{7b}$$

are FW directions; see the literature for the details of FW [18]. We determine the step sizes by exact line search

$$\alpha_{S,k} = \arg\min_{\alpha\in(0,1]} f(\boldsymbol{A}^k, \boldsymbol{S}^k + \alpha(\boldsymbol{P}_S^k - \boldsymbol{S}^k)), \tag{8a}$$

$$\alpha_{A,k} = \arg\min_{\alpha\in(0,1]} f(\boldsymbol{A}^k + \alpha(\boldsymbol{P}_A^k - \boldsymbol{A}^k), \boldsymbol{S}^{k+1}). \tag{8b}$$

A reason for considering the FW inexact BCD scheme is that the FW method is "projection-free." In the PG step in (5a), the proximal mapping requires us to perform unit simplex projections. This results in a complexity of $\mathcal{O}(LN\log(N))$. In the FW step in (6) and (7), the operations are computationally simpler. It can be shown that

$$[\boldsymbol{P}_S^k]_i = \boldsymbol{e}_{j_i^\star}, \quad j_i^\star = \arg\min_{j=1,\ldots,N} [\nabla_{\boldsymbol{S}} f(\boldsymbol{A}^k, \boldsymbol{S}^k)]_{ji},$$

for $i = 1, \ldots, L$. As seen above, the computations of $\boldsymbol{P}_S^k$ do not require floating point operations. Hence, the FW step with $\boldsymbol{S}$ is cheaper than the PG step on a per-iteration basis. Also, the FW direction of $\boldsymbol{A}$ in (7b) equals

$$[\boldsymbol{P}_A^k]_{ij} = \begin{cases} 0, & [\nabla_{\boldsymbol{A}} f(\boldsymbol{A}^k, \boldsymbol{S}^{k+1})]_{ij} \geq 0 \\ 1, & [\nabla_{\boldsymbol{A}} f(\boldsymbol{A}^k, \boldsymbol{S}^{k+1})]_{ij} < 0 \end{cases}$$

for all i, j. Furthermore, it can be shown that the step-size rule in (8) has a closed form

$$\alpha_{S,k} = \min\left\{1, \frac{\langle \nabla_{\boldsymbol{S}} f(\boldsymbol{A}^k, \boldsymbol{S}^k), \boldsymbol{S}^k - \boldsymbol{P}_S^k\rangle}{\|\boldsymbol{A}^k(\boldsymbol{P}_S^k - \boldsymbol{S}^k)\boldsymbol{G}\|_F^2 + \|\boldsymbol{F}\boldsymbol{A}^k(\boldsymbol{P}_S^k - \boldsymbol{S}^k)\|_F^2}\right\},$$

$$\alpha_{A,k} = \min\left\{1, \frac{\langle \nabla_{\boldsymbol{A}} f(\boldsymbol{A}^k, \boldsymbol{S}^{k+1}), \boldsymbol{A}^k - \boldsymbol{P}_A^k\rangle}{\|(\boldsymbol{P}_A^k - \boldsymbol{A}^k)\boldsymbol{S}^{k+1}\boldsymbol{G}\|_F^2 + \|\boldsymbol{F}(\boldsymbol{P}_A^k - \boldsymbol{A}^k)\boldsymbol{S}^{k+1}\|_F^2}\right\};$$

they are obtained by utilizing the convex quadratic structure of f.

It is interesting to compare the complexities of the FW and PG updates. We carefully evaluated the complexities of each operation, and summarize the results by means of big O w.r.t. L, N, M in Table 1. Note that $\mathrm{nnz}(\boldsymbol{G})$ denotes the number of nonzero elements of $\boldsymbol{G}$; as the spatial degradation matrix, it has $\mathrm{nnz}(\boldsymbol{G}) = \mathcal{O}(L_{\mathrm{H}}B^2)$ where B is the width of the spatial point spread response (see Fig. 1). We see that the FW updates are more efficient than the PG updates.

PG	$\boldsymbol{S}$-update (5a)	$\mathcal{O}(LN(M + \log(N)) + N\cdot\mathrm{nnz}(\boldsymbol{G}) + N^2M)$
	$\boldsymbol{A}$-update (5b)	$\mathcal{O}(LNM + N\cdot\mathrm{nnz}(\boldsymbol{G}) + N^2L)$
FW	$\boldsymbol{S}$-update (6a)	$\mathcal{O}(LNM + N\cdot\mathrm{nnz}(\boldsymbol{G}))$
	$\boldsymbol{A}$-update (6b)	$\mathcal{O}(LNM + N\cdot\mathrm{nnz}(\boldsymbol{G}))$

Table 1. Big O complexities of the PG and FW updates.

3.3. Hybrid Inexact BCD

We can also consider a hybrid inexact BCD (HiBCD) scheme

$$\boldsymbol{S}^{k+1} = \mathsf{UD}_S(\boldsymbol{A}^k, \boldsymbol{S}^k), \quad \boldsymbol{A}^{k+1} = \mathsf{UD}_A(\boldsymbol{A}^k, \boldsymbol{S}^{k+1}), \tag{9}$$

where UD_S is either the PG step in (5a) or the FW step in (6a); UD_A is either the PG step in (5b) or the FW step in (6b). For example, we can use the PG step for $\boldsymbol{A}$, but use the FW step for $\boldsymbol{S}$ to avoid unit simplex projections.

3.4. Convergence Guarantees

Our development has been intuitive with an emphasis on computational aspects. Now we turn to fundamental issues. Our main problem in (3) is non-convex. Despite such difficulty, it is useful to understand whether our schemes have guarantees on convergence to a stationary point. First, let us recognize two facts.

Fact 1 *The PG inexact BCD scheme in* (5) *is an instance of an optimization framework called block successive upper-bound minimization (BSUM) [11]. It is also an instance of the block multiconvex optimization framework in [12]. Following the aforementioned frameworks,* (5) *guarantees convergence to a stationary point of Problem* (3).

Fact 2 *The FW inexact BCD scheme in* (6) *is an instance of an optimization framework called cyclic block conditional gradient (CBCG) [13]. However, CBCG was developed for convex problems.*

We skip the details of the above facts for conciseness. Our next question is whether the HiBCD scheme also has the same or similar convergence guarantees. The answer is yes.

Theorem 1 *The HiBCD scheme in* (9) *guarantees convergence to a stationary point of Problem* (3)*. Also, its convergence rate, measured by means of the FW gap [19, 20], is $\mathcal{O}(1/\sqrt{k})$.*

Theorem 1 is our result. The proof follows that of CBCG, and the significant contribution of our result lies in extending the CBCG analysis to include PG. Also, the result is not limited to Problem (3) and applies to a wider class of problems. We should point out that since both the PG inexact BCD scheme in (5) and the FW inexact BCD scheme in (6) are instances of the HiBCD in (9), it follows that the convergence guarantees in Theorem 1 apply to the PG and FW inexact BCD schemes. We skip the proof in view of space constraint and will reveal them in the journal version.

[1] Note that for the case of $\boldsymbol{A} = \mathbf{0}$, which is rare in practice, we can choose the Lipschitz constant as any positive number.

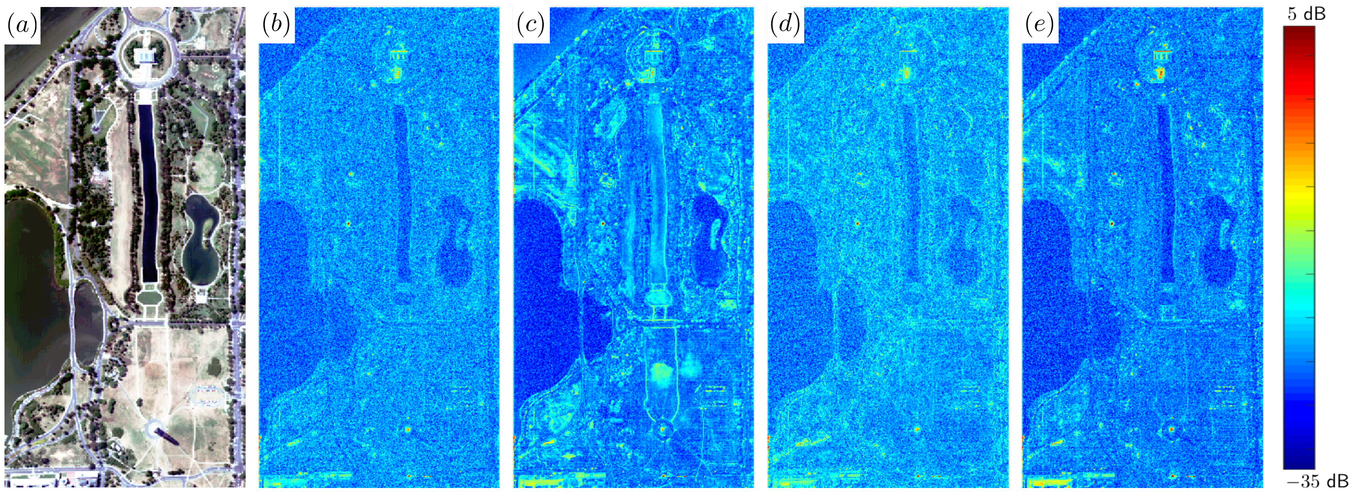

Fig. 2. From left to right: (a) Washington DC RGB image, MSE maps of (b) FUMI, (c) PGiBCD, (d) FWiBCD and (e) HiBCD.

4. SIMULATIONS

In this section we provide numerical results to demonstrate the efficiencies of our inexact BCD schemes. For convenience, we will call the PG inexact BCD scheme in (5) "PGiBCD", the FW inexact BCD scheme in (6) "FWiBCD", and the HiBCD scheme with FW $\boldsymbol{S}$-update and PG $\boldsymbol{A}$-update "HiBCD". The baseline algorithm is FUMI (recall that FUMI is an exact BCD). The algorithm settings of FUMI follow those of the original paper, except that we stop the algorithm when its relative change of the objective value is below $1e^{-4}$. The same stopping rule applies to the three inexact BCD schemes. Also, all the algorithms are initialized by a hyperspectral unmixing algorithm called SVMAX [21]; more specifically, we use SVMAX to estimate $\boldsymbol{A}$ from the HS image $\boldsymbol{Y}_{\mathrm{H}}$, and use that estimation to initialize the algorithms. We benchmark the algorithms by runtime, the number of iterations used, peak SNR (PSNR) and spectral angle mapper (SAM); see [5] for the definitions. The evaluation of runtime was performed on a desktop computer with Intel Core i7 3.6GHz CPU and 16GB memory, and under MATLAB R2015a.

4.1. Synthetic Data Experiment

First, we consider synthetic data simulations. We set $N = 9$, $L = 100^2$. At each simulation trial, the columns of $\boldsymbol{A}$ are randomly selected from the USGS spectral signature library [22], which has $M = 224$. The matrix $\boldsymbol{S}$ is randomly cropped from the abundance map retrieved from the AVIRIS Cuprite dataset (which is a real HS image); the retrieval is done by applying a hyperspectral unmixing algorithm on the dataset. After obtaining $\boldsymbol{A}, \boldsymbol{S}$, we use (1)–(2) to generate $\boldsymbol{X}, \boldsymbol{Y}_M, \boldsymbol{Y}_H$; noise is randomly generated following an i.i.d. white Gaussian distribution. The spatial degradation matrix $\boldsymbol{G}$ corresponds to 11×11 Gaussian point spreading with variance $\sigma^2 = 1.7^2$, followed by downsampling on every 4 pixels horizontally and vertically. The spectral degradation matrix $\boldsymbol{F}$ corresponds to the band-average relative spectral response of the LANDSAT specification [23]. Subsequently we have $(L_H, M_M) = (25^2, 6)$. For each SNR point, we ran 100 independently generated trials to evaluate the aforementioned performance measures on average.

The results are shown in Table 2. We observe that the recovery performance of all the tested algorithms is similar; e.g., the differences of PSNRs of the tested algorithms are no greater than 1.3dB. However, all the three inexact BCD schemes run faster than FUMI—with FWiBCD and HiBCD being particularly fast.

Table 2. HSR Performance on the Synthetic Data

SNR	Method	Runtime (sec.)	Iterations	PSNR (dB)	SAM (deg.)
20	FUMI	8.31 ± 1.75	214.44 ± 51.59	16.39 ± 0.41	16.72 ± 1.56
	PGiBCD	3.41 ± 0.43	503.04 ± 62.81	$\mathbf{17.68 \pm 0.51}$	$\mathbf{14.41 \pm 1.61}$
	FWiBCD	$\mathbf{1.01 \pm 0.09}$	$\mathbf{164.30 \pm 13.55}$	17.59 ± 0.49	14.58 ± 1.59
	HiBCD	1.59 ± 0.18	272.40 ± 32.08	17.66 ± 0.51	14.46 ± 1.60
30	FUMI	15.80 ± 4.48	435.70 ± 125.92	22.72 ± 0.81	5.87 ± 0.75
	PGiBCD	7.01 ± 0.96	1057.14 ± 140.57	$\mathbf{24.33 \pm 0.89}$	$\mathbf{4.83 \pm 0.71}$
	FWiBCD	$\mathbf{1.44 \pm 0.20}$	$\mathbf{237.12 \pm 34.17}$	24.19 ± 0.85	4.90 ± 0.71
	HiBCD	2.51 ± 043	434.34 ± 75.09	24.27 ± 0.87	4.86 ± 0.71
40	FUMI	19.45 ± 5.89	566.60 ± 181.69	32.41 ± 0.98	1.69 ± 0.25
	PGiBCD	14.78 ± 3.25	2235.18 ± 496.21	$\mathbf{33.02 \pm 1.08}$	$\mathbf{1.57 \pm 0.25}$
	FWiBCD	$\mathbf{3.11 \pm 0.53}$	$\mathbf{515.12 \pm 87.38}$	32.61 ± 1.04	1.65 ± 0.26
	HiBCD	5.29 ± 1.07	932.14 ± 191.53	32.72 ± 1.05	1.63 ± 0.26

Table 3. HSR Performance on Washington DC dataset

Method	Runtime (sec.)	Iterations	PSNR (dB)	SAM (deg.)
FUMI	1162.53 ± 235.89	$\mathbf{950.23 \pm 194.31}$	41.24 ± 0.53	0.71 ± 0.05
PGiBCD	560.64 ± 18.84	2115.23 ± 71.34	$\mathbf{46.88 \pm 0.04}$	$\mathbf{0.59 \pm 0.01}$
FWiBCD	$\mathbf{304.73 \pm 9.71}$	1610.47 ± 51.01	41.34 ± 0.12	0.94 ± 0.01
HiBCD	310.52 ± 8.35	1689.94 ± 46.20	44.25 ± 0.09	0.89 ± 0.01

4.2. Semi-Real Data Experiment

Next, we consider a semi-real data experiment, following a standard procedure called Wald's protocol [24]. In short, we take a real HS image as the SR image to perform the experiment. The image we use is a 520×260 sub-image cropped from the Washington DC image captured by HYDICE sensor [25]; it has $M = 191$ bands. Fig. 2(a) shows the image. The settings for generating $\boldsymbol{Y}_H$ and $\boldsymbol{Y}_M$ are identical to those of the last subsection. The SNR is set at 40dB. We set the model order as $N = 30$.

Fig. 2 (b)–(e) show an instance of the mean square error (MSE) maps of the algorithms. We see that they all yield low MSEs in general, and thus perform HSR reasonably. Table 3 shows more results; they are based on 50 trials. Again, the recovery performances of all the algorithms are comparable, but the inexact BCDs are faster—FWiBCD and HiBCD are almost 4 times faster than FUMI.

5. CONCLUSION

To conclude, inexact BCD schemes based on PG, FW and their hybrids were proposed to tackle SMF for HSR. Computational and convergence issues were dealt with. Numerical results showed that the proposed schemes have promising runtime performance.

6. REFERENCES

[1] R. B. Gomez, A. Jazaeri, and M. Kafatos, "Wavelet-based hyperspectral and multispectral image fusion," in *Proc. SPIE*, vol. 4383, 2001, pp. 36–42.

[2] R. C. Hardie, M. T. Eismann, and G. L. Wilson, "MAP estimation for hyperspectral image resolution enhancement using an auxiliary sensor," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1174–1184, 2004.

[3] T. Akgun, Y. Altunbasak, and R. M. Mersereau, "Super-resolution reconstruction of hyperspectral images," *IEEE Trans. Image Process.*, vol. 14, no. 11, pp. 1860–1875, 2005.

[4] A. Mookambiga and V. Gomathi, "Comprehensive review on fusion techniques for spatial information enhancement in hyperspectral imagery," *Multidimens. Syst. Signal Process.*, vol. 27, no. 4, pp. 863–889, 2016.

[5] N. Yokoya, C. Grohnfeldt, and J. Chanussot, "Hyperspectral and multispectral data fusion: A comparative review," *IEEE Geosci. Remote Sens. Mag.*, 2017.

[6] R. Kawakami, Y. Matsushita, J. Wright, M. Ben-Ezra, Y.-W. Tai, and K. Ikeuchi, "High-resolution hyperspectral imaging via matrix factorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 2329–2336.

[7] N. Yokoya, T. Yairi, and A. Iwasaki, "Coupled nonnegative matrix factorization unmixing for hyperspectral and multispectral data fusion," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 2, pp. 528–537, 2012.

[8] E. Wycoff, T.-H. Chan, K. Jia, W.-K. Ma, and Y. Ma, "A non-negative sparse promoting algorithm for high resolution hyperspectral imaging," in *Proc. IEEE Conf. Acoust. Speech Signal Process.* IEEE, 2013, pp. 1409–1413.

[9] M. Simões, J. Bioucas-Dias, L. B. Almeida, and J. Chanussot, "A convex formulation for hyperspectral image superresolution via subspace-based regularization," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 6, pp. 3373–3388, 2015.

[10] Q. Wei, J. Bioucas-Dias, N. Dobigeon, J.-Y. Tourneret, M. Chen, and S. Godsill, "Multiband image fusion based on spectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7236–7249, 2016.

[11] M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM J. Optim.*, vol. 23, no. 2, pp. 1126–1153, 2013.

[12] Y. Xu and W. Yin, "A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion," *SIAM J. Imaging Sci.*, vol. 6, no. 3, pp. 1758–1789, 2013.

[13] A. Beck, E. Pauwels, and S. Sabach, "The cyclic block conditional gradient method for convex optimization problems," *SIAM J. Optim.*, vol. 25, no. 4, pp. 2024–2049, 2015.

[14] S. Bonettini, M. Prato, and S. Rebegoldi, "A cyclic block coordinate descent method with generalized gradient projections," *Appl. Math. Comput.*, vol. 286, pp. 288–300, 2016.

[15] W.-K. Ma, J. M. Bioucas-Dias, T.-H. Chan, N. Gillis, P. Gader, A. J. Plaza, A. Ambikapathi, and C.-Y. Chi, "A signal processing perspective on hyperspectral unmixing: Insights from remote sensing," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 67–81, 2014.

[16] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends® Optim.*, vol. 1, no. 3, pp. 127–239, 2014.

[17] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the l1-ball for learning in high dimensions," in *Proc. 25th Int. Conf. Mach. Learn.* ACM, 2008, pp. 272–279.

[18] M. Jaggi, "Revisiting Frank-Wolfe: Projection-free sparse convex optimization," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. 427–435.

[19] B. Jiang, T. Lin, S. Ma, and S. Zhang, "Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis," *arXiv preprint arXiv:1605.02408*, 2016.

[20] H.-T. Wai, J. Lafond, A. Scaglione, and E. Moulines, "Decentralized Frank-Wolfe algorithm for convex and non-convex problems," *IEEE Trans. Automat. Contr.*, vol. 62, no. 11, pp. 5522–5537, 2017.

[21] T.-H. Chan, W.-K. Ma, A. Ambikapathi, and C.-Y. Chi, "A simplex volume maximization framework for hyperspectral endmember extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, pp. 4177–4193, 2011.

[22] R. N. Clark, G. A. Swayze, R. Wise, K. E. Livo, T. Hoefen, R. F. Kokaly, and S. J. Sutley, "USGS digital spectral library splib06a," *U.S. Geol. Surv., Digital Data Ser.*, vol. 231, 2007.

[23] G. Chander, B. L. Markham, and D. L. Helder, "Summary of current radiometric calibration coefficients for Landsat MSS, TM, ETM+, and EO-1 ALI sensors," *Remote Sens. Environ.*, vol. 113, no. 5, pp. 893–903, 2009.

[24] L. Wald, "Quality of high resolution synthesised images: Is there a simple criterion?" in *Third conference "Fusion of Earth data: merging point measurements, raster maps and remotely sensed images"*. SEE/URISCA, 2000, pp. 99–103.

[25] R. W. Basedow, D. C. Carmer, and M. E. Anderson, "HYDICE system: Implementation and performance," in *SPIE's 1995 Symp. OE/Aerosp. Sens. Dual Use Photon.* International Society for Optics and Photonics, 1995, pp. 258–267.