A Novel Iterative Convex Approximation Method

Yang Yang and Marius Pesavento
Communication Systems Group, Darmstadt University of Technology, Darmstadt, Germany.
Emails: {yang, pesavento}@nt.tu-darmstadt.de

***Abstract*—In this paper, we propose a novel iterative algorithm based on convex approximation for a large class of possibly nonconvex optimization problems. The stationary points of the original problem are found by solving a sequence of successively refined approximate problems. To achieve convergence, the approximate problem only needs to be pseudo-convex while the stepsizes are determined by the exact or successive line search. The proposed method not only includes as special cases a variety of existing methods, for example, the gradient projection method and the Jacobi algorithm, but also leads to new algorithms which enjoy easier implementation and faster convergence speed, as illustrated (both theoretically and numerically) by the example application of the sum capacity computation problem of the MIMO broadcast channel.**

I. Introduction

In this paper, we consider the following problem:

$$\underset{\mathbf{x}\in\mathcal{X}}{\text{minimize}} \quad f(\mathbf{x}), \tag{1}$$

where $f(\mathbf{x}) : \mathcal{C}^n \to \mathcal{R}$ is a proper and differentiable function with a continuous gradient, and $\mathcal{X} \subseteq \mathcal{C}^n$ is a closed and convex set. We assume that problem (1) has a solution.

Since we do not assume that $f(\mathbf{x})$ is convex, problem (1) is in general nonconvex. We thus focus on iterative algorithms that can solve (1) efficiently in the sense of stationary points [1]. As example applications consider the MIMO broadcast channel (BC) [2], where $f(\mathbf{x})$ is the sum-rate function of multiple users (to be maximized) while the set $\mathcal{X}$ characterizes the users' power constraints.

Commonly used iterative algorithms belong, e.g., to the class of descent direction methods such as the conditional gradient and gradient projection method [1] which often suffer from slow convergence. To speed up the convergence, algorithms based on nonlinear best-response have been widely studied. These methods are applicable if $\mathcal{X}$ in (1) has a Cartesian product structure, i.e.,

$$\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_K. \tag{2}$$

The resulting problem can be solved in a distributed manner as the change of one variable does not affect the feasibility of other variables [3]. The block coordinate descent (BCD) method [1, Sec. 2.7] is an example of distributed optimization algorithms: in each iteration, only one variable is updated by the solution that minimizes $f(\mathbf{x})$ with respect to (w.r.t.) that variable while the remaining variables are fixed, and the variables are updated sequentially. This method and its variants have been successfully adopted to many problems [4].

When the number of variables is large, the BCD method may suffer from slow convergence due to the sequential update. Existing parallel update seems more desirable, but their convergence conditions are rather restrictive, cf. the diagonal dominance condition on the objective function $f(\mathbf{x})$ [3] or the sufficiently small stepsize condition [1], [2]. A recent progress in parallel algorithms has been made in [5], [6], and it was shown that, for a large class of nonconvex optimization problems, a stationary point can be found by solving a sequence of successively refined approximate problems. This algorithm is essentially an iterative descent direction method and convergence is established if, among other conditions, the approximate function and stepsizes are properly selected.

Despite its novelty, the parallel algorithm proposed in [5]–[7] suffers from a limitation, namely, the approximate function must be strongly convex. This is usually guaranteed by adding an additional regularization term to the original function $f(\mathbf{x})$, which however may destroy the desirable structure that could otherwise be exploited, e.g., to obtain computationally efficient closed-form solution of the approximate problems [4].

Based on the idea first presented in [5]–[7], we develop a novel iterative convex approximation method in which the approximate function only needs a weak form of convexity, namely, pseudo-convexity. This iterative method not only includes as special cases many existing methods, for example, [4], [5], [7]–[9], but also opens new possibilities to construct approximate problems that are easier to solve. For example, in the MIMO IC, MAC and BC sum-rate maximization problems, the new approximate problems can be solved in closed-form. Besides, we show by a counterexample that the assumption on pseudo-convexity is tight in the sense that if it is not satisfied, the algorithm may not converge.

To guarantee the convergence of the proposed algorithm, the exact/successive line search procedure is used to determine the stepsize, and its implementation may depend on the existence of a centralized controller. The existence of such a centralized controller can be justified in many scenarios, e.g., the base station in MIMO BC.

II. The Proposed Iterative Convex Approximation Method

We start with the definition of a *pseudo-convex function:* A function $g(\mathbf{x})$ is said to be pseudo-convex if

$$g(\mathbf{y}) < g(\mathbf{x}) \Longrightarrow (\mathbf{y}-\mathbf{x})^T \nabla g(\mathbf{x}) < 0. \tag{3}$$

In other words, $g(\mathbf{y}) < g(\mathbf{x})$ implies $\mathbf{y}-\mathbf{x}$ is a *descent direction* of $g(\mathbf{x})$. We remark that the (strong) convexity of a function implies that the function is pseudo-convex, which

978-1-4799-1963-5/15/$31.00 ©2015 IEEE

in turn implies that the function is quasi-convex. That is:

$$\begin{array}{c} g(\mathbf{x}) \text{ is strongly convex} \rightarrow g(\mathbf{x}) \text{ is convex} \\ \downarrow \\ g(\mathbf{x}) \text{ is quasi-convex} \leftarrow g(\mathbf{x}) \text{ is pesudo-convex} \end{array} \quad (4)$$

We solve (1) as a sequence of successively refined approximate problems, each of which is presumably much easier to solve than the original problem (1). In iteration t, let $\tilde{f}(\mathbf{x};\mathbf{x}^t)$ be the approximate function of $f(\mathbf{x})$ around the point $\mathbf{x}^t$. Then the approximate problem is

$$\underset{\mathbf{x}\in\mathcal{X}}{\text{minimize}} \quad \tilde{f}(\mathbf{x};\mathbf{x}^t).$$

We assume that the approximate function $\tilde{f}(\mathbf{x};\mathbf{y})$ satisfies several technical conditions:
(A1) $\tilde{f}(\mathbf{x};\mathbf{y})$ is pseudo-convex in $\mathbf{x}\in\mathcal{X}$ for any $\mathbf{y}\in\mathcal{X}$;
(A2) $\tilde{f}(\mathbf{x};\mathbf{y})$ is continuously differentiable in both $\mathbf{x}\in\mathcal{X}$ and $\mathbf{y}\in\mathcal{X}$, and $\nabla_{\mathbf{x}}\tilde{f}(\mathbf{x};\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x})$;

Let us define the operator $\mathbb{B}\mathbf{x}^t$ as the minimizer of the approximate function $\tilde{f}(\mathbf{x};\mathbf{x}^t)$ over $\mathbf{x}\in\mathcal{X}$:

$$\mathbb{B}\mathbf{x}^t \in \mathcal{S}(\mathbf{x}^t) \triangleq \left\{\mathbf{x}^\star\in\mathcal{X}:\ \tilde{f}(\mathbf{x}^\star;\mathbf{x}^t) = \min_{\mathbf{x}\in\mathcal{X}} \tilde{f}(\mathbf{x};\mathbf{x}^t)\right\}. \quad (5)$$

We shall exploit the following property of $\mathbb{B}\mathbf{x}$ [10].

Proposition 1. *A vector $\mathbf{x}$ is a stationary point of (1) if and only if $\mathbf{x}\in\mathcal{S}(\mathbf{x})$. If $\mathbf{x}$ is not a stationary point of (1), then $\mathbb{B}\mathbf{x}-\mathbf{x}$ is a descent direction of $f(\mathbf{x})$:*

$$(\mathbb{B}\mathbf{x}-\mathbf{x})^T\nabla f(\mathbf{x}) < 0. \quad (6)$$

With the descent direction $\mathbb{B}\mathbf{x}^t-\mathbf{x}^t$, the vector update $\mathbf{x}^{t+1}$ in the $(t+1)$-th iteration is defined as follows:

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \gamma^t(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t), \quad (7)$$

where $\gamma^t\in(0,1]$ is an appropriate stepsize that is determined by the following standard rules.
Exact line search: The stepsize is selected such that the function $f(\mathbf{x})$ is decreased to the largest extent along the descent direction $\mathbb{B}\mathbf{x}^t-\mathbf{x}^t$:

$$\gamma^t \in \underset{0\le\gamma\le 1}{\arg\min}\ f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)). \quad (8)$$

If $f(\mathbf{x})$ is convex and $\gamma^\star$ nulls the gradient of $f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t))$, i.e., $\nabla_\gamma f(\mathbf{x}^t+\gamma^\star(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)) = 0$, then γ^t in (8) is simply the projection of $\gamma^\star$ onto the interval $[0,1]$:

$$\gamma^t = \begin{cases} 1, & \text{if } \nabla_\gamma f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t))|_{\gamma=1} \ge 0, \\ 0, & \text{if } \nabla_\gamma f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t))|_{\gamma=0} \le 0, \\ \gamma^\star, & \text{otherwise.} \end{cases}$$

In some applications it is possible to compute $\gamma^\star$ analytically, e.g., if $f(\mathbf{x})$ is convex quadratic. Otherwise, for general convex functions, $\gamma^\star$ can be found efficiently by the bisection method: since $f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t))$ is convex in γ, it follows that $\nabla_\gamma f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)) < 0$ if $\gamma<\gamma^\star$ and $\nabla_\gamma f(\mathbf{x}^t+\gamma(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)) > 0$ if $\gamma>\gamma^\star$; we thus can find $\gamma^\star$ simply by bisection: given an interval $[\gamma_{\text{low}},\gamma_{\text{up}}]$ containing $\gamma^\star$ (the initial value of γ_{low} and γ_{up} is 0 and 1, respectively), set $\gamma_{\text{mid}} = (\gamma_{\text{low}}+\gamma_{\text{up}})/2$ and refine γ_{low} and γ_{up} as follows: $\gamma_{\text{low}} = \gamma_{\text{mid}}$ if $\nabla_\gamma f(\mathbf{x}^t+\gamma_{\text{mid}}(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)) > 0$ or $\gamma_{\text{up}} = \gamma_{\text{mid}}$ otherwise. This procedure is repeated for a finite number of times until the gap $\gamma_{\text{up}}-\gamma_{\text{low}}$ is smaller than a prescribed precision.

Algorithm 1 The iterative convex approximation algorithm

Data: $t=0$ and $\mathbf{x}^0\in\mathcal{X}$; stop criterion δ.
S1: Compute $\mathbb{B}\mathbf{x}^t$ according to (5).
S2: Determine the stepsize γ^t by exact/successive line search.
S3: Update $\mathbf{x}$ according to (7).
S4: If $\left|(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)^T\nabla f(\mathbf{x}^t)\right| \le \delta$, STOP; otherwise go to **S1**.

Successive line search: If no structure in $f(\mathbf{x})$ can be exploited to efficiently compute γ^t according to the exact line search (8), one can instead employ the successive line search: given scalars $0<\alpha<1$ and $0<\beta<1$, we set $\gamma^t=\beta^{m_t}$, where m_t is the first nonnegative integer m for which the following inequality is satisfied:

$$f(\mathbf{x}^t+\beta^m(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)) \le f(\mathbf{x}^t)+\alpha\beta^m(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)^T\nabla f(\mathbf{x}^t). \quad (9)$$

The existence of a finite m_t is always guaranteed if $(\mathbb{B}\mathbf{x}^t-\mathbf{x}^t)^T\nabla f(\mathbf{x}^t) < 0$ [1].

The algorithm is summarized in Algorithm 1 and its convergence properties are given in the following theorem [10].

Theorem 2. *Consider the sequence $\{\mathbf{x}^t\}$ generated by Algorithm 1. Assume Assumptions (A1)-(A2) as well as the following assumptions are satisfied:*

(A3) The solution set $\mathcal{S}(\mathbf{x}^t)$ is nonempty for $t=1,2,\ldots$;
(A4) Given any convergent subsequence $\{\mathbf{x}^t\}_{t\in\mathcal{T}}$ where $\mathcal{T}\subseteq\{1,2,\ldots\}$, the sequence $\{\mathbb{B}\mathbf{x}^t\}_{t\in\mathcal{T}}$ is bounded.

Then any limit point of $\{\mathbf{x}^t\}$ is a stationary point of (1).

Other structures in $\tilde{f}(\mathbf{x};\mathbf{x}^t)$, if any, can be further exploited to assist in the selection of the stepsize. For example, if $\tilde{f}(\mathbf{x};\mathbf{x}^t)$ is an upper bound of $f(\mathbf{x})$ that is exact at $\mathbf{x}=\mathbf{x}^t$:
(A5) $\tilde{f}(\mathbf{x};\mathbf{x}^t)\ge f(\mathbf{x})$ and $\tilde{f}(\mathbf{x}^t;\mathbf{x}^t)=f(\mathbf{x}^t)$,
then we can set $\gamma^t=1$ and (7) reduces to [10]

$$\mathbf{x}^{t+1} = \mathbb{B}\mathbf{x}^t = \underset{\mathbf{x}\in\mathcal{X}}{\arg\min}\ \tilde{f}(\mathbf{x};\mathbf{x}^t).$$

In the following we discuss some properties of the proposed Algorithm 1.

On the pseudo-convexity of $f(\mathbf{x};\mathbf{x}^t)$. Assumption (A1) is tight in the sense that if (A1) is not satisfied, Proposition 1 does not hold any more. Consider the following simple example: $f(x)=x^3$, $-1\le x\le 1$. It is easy to see that $x=0$ is a stationary point. If $x^t=0$ and we set $\tilde{f}(x;x^t)=x^3$, which is quasi-convex but not pseudo-convex, all assumptions except Assumption (A1) are satisfied and $\mathbb{B}x^t=\mathcal{S}(x^t)=-1$. However, $0=x^t\notin\mathcal{S}(x^t)=-1$, and thus the conclusions of Proposition 1 no longer hold.

On the convergence conditions. A sufficient condition for Assumptions (A3)-(A4) is that the feasible set $\mathcal{X}$ is bounded or the approximate function $\tilde{f}(\mathbf{x};\mathbf{x}^t)$ is strongly convex [11].

On the approximate problem (5). The only requirement on the approximate function $\tilde{f}(\mathbf{x};\mathbf{x}^t)$ is that it is pseudo-convex

(cf. Assumption (A1)), and to the best of our knowledge, this is the weakest condition that can be found in the literature. As a result, it enables us to construct new approximate functions that can often be optimized more easily or even in closed-form. This results in a significant reduction of the computational cost if the approximate problem must otherwise only be optimized by iterative algorithms as in standard solvers.

III. Special Cases of the Proposed Method

In this section, we interpret some existing methods in the context of our algorithm and show that how they represent special cases of Algorithm 1.

Conditional gradient method. In this method, the approximate function is just the first-order approximation of $f(\mathbf{x})$ at $\mathbf{x}=\mathbf{x}^t$, i.e.,

$$\tilde{f}(\mathbf{x};\mathbf{x}^t)=f(\mathbf{x}^t)+(\mathbf{x}-\mathbf{x}^t)^T\nabla f(\mathbf{x}^t). \tag{10}$$

The stepsize is selected by exact or successive line search.

Gradient projection method. In this method, $\mathbb{B}\mathbf{x}^t$ is given by [1, Sec. 2.3]

$$\mathbb{B}\mathbf{x}^t=\left[\mathbf{x}^t-s^t\nabla f(\mathbf{x}^t)\right]_{\mathcal{X}},$$

where $s^t>0$ and $[\mathbf{x}]_{\mathcal{X}}$ denotes the projection of $\mathbf{x}$ onto $\mathcal{X}$. This is equivalent to defining $\tilde{f}(\mathbf{x};\mathbf{x}^t)$ in (5) as follows:

$$\tilde{f}(\mathbf{x};\mathbf{x}^t)=(\mathbf{x}-\mathbf{x}^t)^T\nabla f(\mathbf{x}^t)+\tfrac{1}{2s^k}\left\|\mathbf{x}-\mathbf{x}^t\right\|_2^2,$$

which is the first-order approximation of $f(\mathbf{x})$ plus a quadratic regularization term introduced for numerical stability [3]. Then the stepsize is selected by the exact/successive line search.

Jacobi algorithm. If $f(\mathbf{x})$ is componentwise convex, the approximate function can be

$$\tilde{f}(\mathbf{x};\mathbf{x}^t)=\textstyle\sum_{k=1}^K\big(f(\mathbf{x}_k,\mathbf{x}_{-k}^t)+\frac{\tau_k}{2}\left\|\mathbf{x}_k-\mathbf{x}_k^t\right\|_2^2\big), \tag{11}$$

where $\tau_k\geq 0$ for all k. The k-th component function in (11) is obtained from the original function $f(\mathbf{x})$ by fixing all variables except $\mathbf{x}_k$, i.e., $\mathbf{x}_{-k}=\mathbf{x}_{-k}^t$, and further adding an (optional) regularization term. The resulting approximate problem is

$$\begin{array}{ll}\underset{\mathbf{x}=(\mathbf{x}_k)_{k=1}^K}{\text{minimize}} & \sum_{k=1}^K\big(f(\mathbf{x}_k,\mathbf{x}_{-k}^t)+\frac{\tau_k}{2}\left\|\mathbf{x}_k-\mathbf{x}_k^t\right\|_2^2\big)\\ \text{subject to} & \mathbf{x}\in\mathcal{X}.\end{array} \tag{12}$$

If the constraint set has a Cartesian product structure as in (2), problem (12) can be decomposed into a set of smaller subproblems, one for each variable, which are then solved *in parallel* (i.e., Jacobi update): $\mathbb{B}\mathbf{x}^t=(\mathbb{B}_k\mathbf{x}^t)_{k=1}^K$ where

$$\mathbb{B}_k\mathbf{x}^t\in\underset{\mathbf{x}_k\in\mathcal{X}_k}{\arg\min}\left\{f(\mathbf{x}_k,\mathbf{x}_{-k}^t)+\tfrac{\tau_k}{2}\left\|\mathbf{x}_k-\mathbf{x}_k^t\right\|_2^2\right\},\ \forall k.$$

To guarantee the convergence, the condition proposed in [9] is that $\tau_k>0$ for all k in (11). However, this may destroy the convenient structure that could otherwise be exploited. In the proposed method, the convergence is guaranteed even when $\tau_k=0$ in (11). To our knowledge, this is the weakest convergence condition available in literature. We will show by an example application in the MIMO BC in Sec. IV that the proposed relaxation in approximate function yields new approximate problems that are much easier to solve.

IV. Computation of MIMO BC Sum Capacity

In the MIMO BC, assume $\mathbf{H}_k$ is the channel from the base station to user k, and $\mathbf{Q}_k$ is the transmit covariance matrix of the signal from the base station to user k. Then the sum capacity of MIMO BC is [2]

$$\begin{array}{ll}\underset{(\mathbf{Q}_k\succeq\mathbf{0})_{k=1}^K}{\text{maximize}} & \log\det\left|\mathbf{I}+\sum_{k=1}^K\mathbf{H}_k\mathbf{Q}_k\mathbf{H}_k^H\right|\\ \text{subject to} & \sum_{k=1}^K\text{tr}(\mathbf{Q}_k)\leq P,\end{array} \tag{13}$$

where P is the power budget at the base station.

To apply Algorithm 1, we invoke (11)-(12) and the approximate problem at iteration t is

$$\begin{array}{ll}\underset{(\mathbf{Q}_k\succeq\mathbf{0})_{k=1}^K}{\text{maximize}} & \sum_{k=1}^K\log\left|\mathbf{R}_k(\mathbf{Q}_{-k}^t)+\mathbf{H}_k\mathbf{Q}_k\mathbf{H}_k^H\right|\\ \text{subject to} & \sum_{k=1}^K\text{tr}(\mathbf{Q}_k)\leq P,\end{array} \tag{14}$$

where $\mathbf{R}_k(\mathbf{Q}_{-k})\triangleq\mathbf{R}_n+\sum_{j\neq k}\mathbf{H}_j\mathbf{Q}_j\mathbf{H}_j^H$ is the convariance matrix of noise plus interference for user k. Problem (14) is convex and can be solved from the dual domain by relaxing the sum-power constraint into the Lagrangian [1]:

$$\mathbb{B}\mathbf{Q}^t\triangleq\underset{(\mathbf{Q}_k\succeq\mathbf{0})_{k=1}^K}{\arg\max}\left\{\begin{array}{l}\sum_{k=1}^K\log\left|\mathbf{R}_k(\mathbf{Q}_{-k}^t)+\mathbf{H}_k\mathbf{Q}_k\mathbf{H}_k^H\right|\\ -\lambda^\star(\sum_{k=1}^K\text{tr}(\mathbf{Q}_k)-P)\end{array}\right\}. \tag{15}$$

where $\mathbb{B}\mathbf{Q}^t=(\mathbb{B}_k\mathbf{Q}^t)_{k=1}^K$ and $\lambda^\star$ is the optimal Lagrange multiplier which satisfies the following conditions: $\lambda^\star\geq 0$, $\sum_{k=1}^K\text{tr}(\mathbb{B}_k\mathbf{Q}^t)-P\leq 0$, $\lambda^\star(\sum_{k=1}^K\text{tr}(\mathbb{B}_k\mathbf{Q}^t)-P)=0$, and can be found efficiently by bisection.

Since the optimization problem in (15) is uncoupled among different variables $\mathbf{Q}_k$ in both the objective function and the constraint set, it can be solved in parallel:

$$\mathbb{B}_k\mathbf{Q}^t=\underset{\mathbf{Q}_k\succeq\mathbf{0}}{\arg\max}\left\{\log\left|\mathbf{R}_k(\mathbf{Q}_{-k}^t)+\mathbf{H}_k\mathbf{Q}_k\mathbf{H}_k^H\right|-\lambda^\star\text{tr}(\mathbf{Q}_k)\right\}, \tag{16}$$

and $\mathbb{B}_k\mathbf{Q}^t$ has a closed-form expression based on waterfilling solution [2], so problem (14) has a closed-form solution up to a Lagrange multiplier that can be found efficiently by bisection. Given the update direction $\mathbb{B}\mathbf{Q}^t-\mathbf{Q}^t$, the base station can implement the exact line search to determine the stepsize.

The proposed algorithm outperforms [6] and [2] from the perspective of approximate problem and stepsize, respectively. On the one hand, in the iterative algorithm proposed in [6], an additional quadratic regularization term is required in (14) and thus (16) (cf. (12)), which would destroy the existence of a closed-form solution of (16) and increase the complexity of the algorithm dramatically. On the other hand, the iterative algorithm [2] has the same approximate problem (14), but a fixed stepsize $\gamma^t=1/K$ is used. The exact line search performs better than the fixed stepsize because, by definition, the former returns the largest increase in the objective function.

Simulations. We set the parameters as follows: the number of users is $K=$20 and 100. The number of transmit antennas at the base station is 5 and the number of receive antennas of each user is 4. The power constraint is $P=10$.

We compare the proposed algorithm with the iterative algorithm proposed in [2] in Fig. 1, where the sum-rate versus

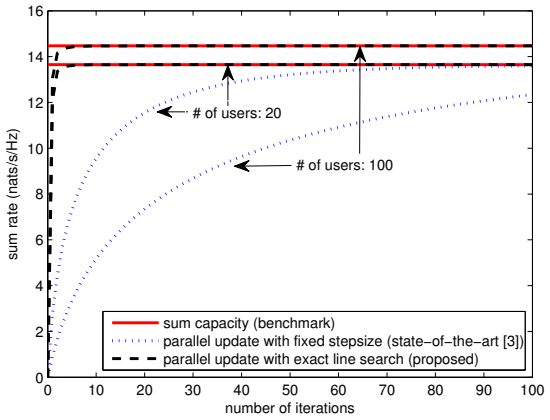

Fig. 1. MIMO BC: sum-rate versus the number of iterations.

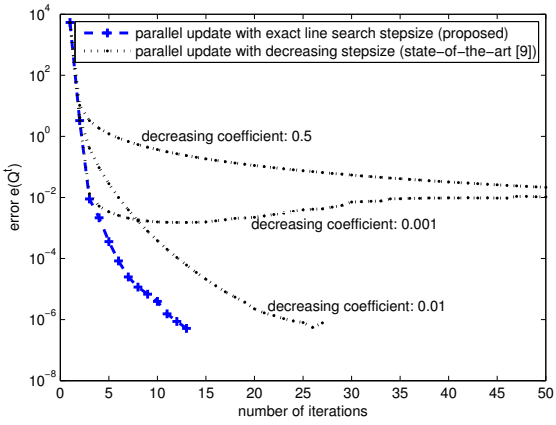

Fig. 2. MIMO BC: error $e(\mathbf{Q}^t) \triangleq \langle \mathbb{B}\mathbf{Q}^t - \mathbf{Q}^t, \nabla f(\mathbf{Q}^t) \rangle$ versus the number of iterations.

the number of iterations is achieved. It is easy to see that the proposed algorithm converges very fast (in less than 10 iterations) to the sum capacity, while [2] requires many more iterations when $K = 20$ and it does not even converge in a reasonable number of iterations when $K = 100$. This is because the exact line search yields the largest increase in the objective function in each iteration while the fixed stepsize tends to be overly conservative. Employing the exact line search adds complexity as compared to the simple choice of a fixed stepsize, however, since the objective function is convex, the exact line search can be implemented efficiently using the bisection method and the additional complexity is usually affordable. Specifically, it takes 0.0023 seconds to solve (14) and 0.0018 seconds to perform the exact line search (the software/hardware environment is specified in [10]), but the saving in the number of iterations is more than enough to compensate the loss incurred by the exact line search.

We also compare in Fig. 2 the proposed algorithm with the iterative algorithm [6], where the approximate problem is (14) but with an additional quadratic regularization term, cf. (12), where $\tau_k = 10^{-5}$ for all k. Besides, the stepsizes used in [6] are decreasing stepsizes $\gamma^{t+1} = \gamma^t(1 - d\gamma^t)$ where $d \in (0, 1)$ controls how fast the stepsize decreases. We see from Fig. 2 that the convergence behavior of [6] is quite sensitive to the stepsize decreasing rate d. The convergence speed is slow when the decreasing rate is either too large ($d = 0.5$) or too small ($d = 0.001$). The choice $d = 0.01$ works well for this case, but a good decreasing rate is usually dependent on problem parameters and no general rule is equally good for all choices of parameters. Besides, it is more computationally consuming to solve (14) in the presence of an additional quadratic regularization term. Specifically, it takes CVX (version 2.0 [12]) 21.1785 seconds (based on the dual approach (16) while $\lambda^\star$ is found by bisection). Therefore, the overall complexity per iteration of the proposed algorithm is much lower than that of [6].

V. Concluding Remarks

In this paper, we have proposed a new iterative algorithm based on successive convex approximation. The only requirement on the approximate function is that it is pseudo-convex. On one hand, the relaxation on the assumptions of the approximate functions can make the approximate problems much easier to solve. On another hand, the stepsize is selected based on the exact/successive line search method so that notable progress is achieved. Additional structures can be exploited to assist with the selection of the stepsize. The advantages of the proposed algorithm have been demonstrated using the example of the sum-rate maximization problem in the MIMO BC, and they are finally consolidated by numerical results.

References

[1] D. P. Bertsekas, *Nonlinear programming*. Athena Scientific, 1999.
[2] N. Jindal, W. Rhee, S. Vishwanath, S. Jafar, and A. Goldsmith, "Sum Power Iterative Water-Filling for Multi-Antenna Gaussian Broadcast Channels," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1570–1580, Apr. 2005.
[3] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: Numerical methods*. Prentice Hall, 1989.
[4] S.-J. Kim and G. B. Giannakis, "Optimal Resource Allocation for MIMO Ad Hoc Cognitive Radio Networks," *IEEE Transactions on Information Theory*, vol. 57, no. 5, pp. 3117–3131, May 2011.
[5] G. Scutari, F. Facchinei, P. Song, D. P. Palomar, and J.-S. Pang, "Decomposition by Partial Linearization: Parallel Optimization of Multi-Agent Systems," *IEEE Transactions on Signal Processing*, vol. 62, no. 3, pp. 641–656, Feb. 2014.
[6] G. Scutari, F. Facchinei, L. Lampariello, , and P. Song, "Distributed Methods for Constrained Nonconvex Multi-Agent Optimization-Part I: Theory," Oct. 2014, submitted to *IEEE Transactions on Signal Processing*. [Online]. Available: http://arxiv.org/abs/1410.4754
[7] F. Facchinei, G. Scutari, and S. Sagratella, "Parallel Selective Algorithms for Big Data Optimizatio," Dec. 2013, accepted in *IEEE Trans. on Signal Process.* [Online]. Available: http://arxiv.org/abs/1402.5521
[8] M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A Unified Convergence Analysis of Block Successive Minimization Methods for Nonsmooth Optimization," *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 1126–1153, Jun. 2013.
[9] Y. Yang, G. Scutari, P. Song, and D. P. Palomar, "Robust MIMO Cognitive Radio Systems Under Interference Temperature Constraints," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 11, pp. 2465–2482, Nov. 2013.
[10] Y. Yang and M. Pesavento, "A novel iterative convex approximation method," Jun. 2015, submitted to *IEEE Transactions on Signal Processing*. [Online]. Available: http://arxiv.org/abs/1506.04972
[11] S. M. Robinson and R. H. Day, "A sufficient condition for continuity of optimal sets in mathematical programming," *Journal of Mathematical Analysis and Applications*, vol. 45, no. 2, pp. 506–511, Feb. 1974.
[12] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming, version 2.0 beta," http://cvxr.com/, 2012.