SIAM J. OPTIM.
Vol. 25, No. 4, pp. 2182–2211

ACCELERATING BLOCK-DECOMPOSITION FIRST-ORDER METHODS FOR SOLVING COMPOSITE SADDLE-POINT AND TWO-PLAYER NASH EQUILIBRIUM PROBLEMS*

YUNLONG HE† AND RENATO D. C. MONTEIRO‡

Abstract. This article considers the (two-player) composite Nash equilibrium (CNE) problem with a separable nonsmooth part, which is known to include the composite saddle-point (CSP) problem as a special case. Due to its two-block structure, this problem can be solved by any algorithm belonging to the block-decomposition hybrid proximal-extragradient (BD-HPE) framework proposed in [R. D. C. Monteiro and B. F. Svaiter, *SIAM J. Optim.*, 23 (2013), pp. 475–507]. The framework consists of a family of inexact proximal point methods for solving a more general two-block structured monotone inclusion problem which, at every iteration, solves two prox subinclusions according to a certain relative error criterion. By exploiting the fact that the two prox subinclusions in the context of the CNE problem are equivalent to two composite convex programs, this article proposes a new instance of the BD-HPE framework that approximately solves them using an accelerated gradient method. It is shown that the new instance is able to take significantly larger prox stepsizes than other instances from this framework that perform single composite gradient steps for solving the subinclusions. As a result, it is shown that the first instance has better iteration-complexity than the latter ones. Finally, it is also shown that the new accelerated BD-HPE instance computationally outperforms several state-of-the-art algorithms on many relevant classes of CSP and CNE instances.

Key words. Nash equilibrium, monotone variational inequality, saddle point, block decomposition, hybrid proximal-extragradient, accelerated method, complexity

AMS subject classifications. 90C60, 90C25, 90C30, 47H05, 47J20, 65K10, 65K05

DOI. 10.1137/130943649

1. Introduction.

The proximal point (PP) method proposed by Rockafellar [20] is a classical iterative scheme for solving the *monotone inclusion problem*, namely, finding x such that $0 \in T(x)$, where T is a maximal monotone point-to-set operator. Its exact version generates a sequence $\{z_k\}$ according to $z_k = (\lambda_k T + I)^{-1}(z_{k-1})$ or, equivalently, z_k as the unique solution of the inclusion $0 \in \lambda_k T(z) + z - z_{k-1}$. Inexact versions of the PP method have also been studied first by Rockafellar [20] based on an absolute error criterion and later by Solodov and Svaiter [22, 23, 24, 25] based on different relative error criteria. In particular, the variant studied in [22], namely, the hybrid proximal-extragradient (HPE) framework, was used to develop and analyze block-decomposition (BD) algorithms in [13]. The main purpose of this paper is to present and study the computational complexity of a special class of BD algorithms for solving the composite Nash equilibrium (CNE) problem in which the two prox inner subproblems are approximately solved at every iteration by an accelerated gradient method (e.g., one of the methods studied in [15, 17, 27]).

Given proper closed convex functions g_1 and g_2, and differentiable real functions Ψ_1 and Ψ_2 on $X \times Y := \operatorname{dom} g_1 \times \operatorname{dom} g_2$, the (two-player) CNE problem consists of

*Received by the editors October 31, 2013; accepted for publication (in revised form) August 26, 2015; published electronically November 10, 2015.
http://www.siam.org/journals/siopt/25-4/94364.html

†School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332 (heyunlong@gatech.edu).

‡School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0205 (monteiro@isye.gatech.edu). The work of this author was partially supported by NSF grant CMMI-1300221 and ONR grant ONR N00014-11-1-0062.

finding a pair $(x, y) \in X \times Y$ such that

$$x \in \operatorname{Argmin}\{\Psi_1(\tilde{x}, y) + g_1(\tilde{x}) : \tilde{x} \in X\}, \quad y \in \operatorname{Argmin}\{\Psi_2(x, \tilde{y}) + g_2(\tilde{y}) : \tilde{y} \in Y\}.$$

When $\Psi_1 = -\Psi_2$, the CNE problem reduces to the more familiar composite saddle-point (CSP) problem. Clearly, the above CNE problem is equivalent to the two-block structured inclusion problem

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \in \begin{pmatrix} \nabla_x \Psi_1(x, y) + \partial g_1(x) \\ \nabla_y \Psi_2(x, y) + \partial g_2(y) \end{pmatrix}. \tag{1.1}$$

A framework of BD-HPE algorithms for solving the above problem was introduced in [13]. Given an iterate $((x_{k-1}, y_{k-1})$ and stepsize $\lambda_k > 0$, the exact version of the BD-HPE framework applied to problem (1.1) computes the next iterate (x_k, y_k) by solving the two decoupled inclusions

$$0 \in \lambda_k \left[\nabla_x \Psi_1(\tilde{x}_k, y_{k-1}) + \partial g_1(\tilde{x}_k)\right] + \tilde{x}_k - x_{k-1}, \tag{1.2}$$

$$0 \in \lambda_k \left[\nabla_y \Psi_2(\tilde{x}_k, y_k) + \partial g_2(y_k)\right] + y_k - y_{k-1} \tag{1.3}$$

sequentially for $\tilde{x}_k$ and y_k, and then setting $x_k = \tilde{x}_k - \lambda \nabla_x \Psi_1(\tilde{x}_k, \tilde{y}_k) + \lambda \nabla_x \Psi_1(\tilde{x}_k, y_{k-1})$. Clearly, inclusion (1.2) (resp., (1.3)) is a special case of the inclusion

$$0 \in \lambda \left[\nabla \tilde{f}(z) + \nabla \tilde{h}(z)\right] + z - w_0, \tag{1.4}$$

or equivalently the optimality condition for the composite (strongly) convex optimization problem

$$\min_{z \in \mathcal{Z}} \left\{ \frac{1}{2}\|z - w_0\|^2 + \lambda[\tilde{f}(z) + \tilde{h}(z)] \right\}, \tag{1.5}$$

where $\lambda = \lambda_k$, $\tilde{f} = \Psi_1(\cdot, y_{k-1})$ (resp., $\tilde{f} = \Psi_2(\tilde{x}_k, \cdot)$), $\tilde{h} = g_1$ (resp., $\tilde{h} = g_2$), and $w_0 = x_{k-1}$ (resp., $w_0 = y_{k-1}$). When both prox subinclusions are viewed in this manner, inexact versions of the BD-HPE method inexactly solve them by computing an approximate solution of (1.5), i.e., a triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$ satisfying the relative error criterion

$$\tilde{s} \in \partial_{\tilde{\varepsilon}} \tilde{h}(\tilde{z}), \qquad \|\lambda(\nabla \tilde{f}(\tilde{z}) + \tilde{s}) + \tilde{z} - w_0\|^2 + 2\lambda\tilde{\varepsilon} \le \sigma_z^2 \|\tilde{z} - w_0\|^2, \tag{1.6}$$

for some constant $\sigma_z \in (0, 1)$. In regards to solving (1.4) according to (1.6), we make the following two important observations under the assumption that $\tilde{f}$ has $\tilde{L}$-Lipschitz continuous gradient (see (1.8)). First, a triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$ satisfying (1.6) can be computed by performing a single composite gradient step from w_0 with respect to (1.5) whenever the prox stepsize λ is sufficiently small, i.e., it satisfies $\lambda\tilde{L} \le \sigma_z$ (see Proposition 2.4). Second, for an arbitrary stepsize $\lambda > 0$, we show in this paper that an accelerated gradient method applied to (1.5) and started from w_0 can find a triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$ satisfying (1.6) in $\mathcal{O}((\lambda\tilde{L} + 1)^{1/2} \log(\lambda\tilde{L} + 1))$ iterations (see Corollary 3.6).

Based on the first observation above, we review a BD-HPE method for solving (1.1) which was essentially introduced in subsection 5.2 of [13]. We note that this BD-HPE instance chooses the prox stepsize as $\lambda = 1/\mathcal{O}(\max\{L_{xx}, L_{yy}, L_{xy}\})$, where L_{xx} (resp., L_{xy}) denotes the uniform Lipschitz constant of $\nabla_x \Psi_1(x, y)$ with respect to x (resp., y) and L_{yy} denotes the uniform Lipschitz constant of $\nabla_y \Psi_2(x, y)$ with respect to y.

Our main contribution in this paper is to present an accelerated instance of the BD-HPE framework which solves the two prox subinclusions by means of an accelerated gradient method (see the second observation above). The resulting accelerated BD-HPE method is able to choose the prox stepsize as large as allowed by the BD-HPE framework, i.e., as $\lambda = 1/\mathcal{O}(L_{xy})$, and as a result performs substantially fewer HPE (outer) iterations than the first BD-HPE method for instances of (1.1) in which $\max\{L_{xx}, L_{yy}\} >> L_{xy}$. Moreover, for these same instances of (1.1), it is also shown here both theoretically (see section 4) and computationally (see section 5) that the overall number of accelerated gradient (inner) iterations performed by the accelerated method is substantially smaller than the number of HPE iterations performed by the first BD-HPE method.

Our paper is organized as follows. Section 2 contains two subsections. The first one describes the two-player Nash equilibrium problem and the notion of a (ρ, ε)-Nash equilibrium. The second one describes the CNE problem and a specialization of the BD-HPE framework, referred to as the CNE-BD-HPE framework, to the CNE context together with iteration-complexity bounds for it to obtain a (ρ, ε)-Nash equilibrium. Sections 3 and 4 develop an instance of the CNE-BD-HPE framework which uses an accelerated gradient method to approximately solve the subproblems. The iteration-complexity for solving the subproblems is derived in section 3. The resulting accelerated BD-HPE instance and its complexity in terms of gradient, projection, and resolvent evaluations are described in section 4. Finally, section 5 evaluates the performance of the new accelerated instance on four classes of CSP and/or CNE problems.

1.1. Previous most related works. Development and analysis of splitting and BD methods is by now a well-developed area, although algorithms which allow a relative error tolerance in the solution of the proximal subproblems have been studied in just a few papers. In particular, Ouorou [18] discusses an ε-proximal decomposition using the ε-subdifferential and a relative error criterion on ε. Projection splitting methods for the sum of arbitrary maximal monotone operators using a particular case of the HPE error tolerance for solving the proximal subproblems were presented in [5, 6]. The use of the HPE method for studying BD methods was first presented in [21]. We observe, however, that none of these works deal with the derivation of iteration-complexity bounds. More recently, Chambolle and Pock [3] have developed and established iteration-complexity bounds for a splitting method, which solves the proximal subproblems exactly, in the context of saddle-point problems with a bilinear coupling.

Special instances of the HPE framework for solving monotone variational inequalities and monotone inclusions (resp., saddle-point problems) are discussed in [11] and in the subsequent papers [12, 13]. More specifically, by viewing Korpelevich's method [7] as well as Tseng's modified forward-backward splitting (MFBS) method [26] as special cases of the HPE method, papers [11, 12] established in the pointwise and ergodic iteration-complexities of these methods applied to either monotone variational inequalities, monotone inclusions consisting of the sum of a Lipschitz continuous monotone map and a maximal monotone operator with an easily computable resolvent, and convex-concave saddle-point problems. In the context of variational inequalities, we should mention that prior to [11, 12], Nemirovski [14] established the ergodic iteration-complexity of his mirror-prox algorithm, which includes Korpelevich's method as a special case, under the assumption that the feasible set of the problem is bounded. Moreover, Nesterov [16] has established the ergodic iteration-complexity of a new dual

extrapolation algorithm whose termination depends on the guess of a ball centered at the initial iterate and containing a solution of the problem.

1.2. Notation and basic definitions. We denote the sets of real numbers by $\Re$, nonnegative numbers by $\Re_+$, and positive numbers by $\Re_{++}$. For a matrix $W \in \Re^{m\times n}$, we denote its Frobenius norm by $\|W\|_F$, the sum of the absolute values of its entries by $\|W\|_1$, and the sum of its singular values by $\|W\|_*$. Let $\mathcal{S}^n$ denote the space of $n \times n$ real symmetric matrices and $\mathcal{S}^n_+$ denote the subset of $\mathcal{S}^n$ consisting of the positive semidefinite matrices. We denote the largest eigenvalue of a matrix $W \in \mathcal{S}^n$ by $\theta_{\max}(W)$. For any $z > 0$, define $\log^+(z) := \max(0, \log(z))$. Let $\lceil z \rceil$ denote the smallest integer not less than $z \in \Re$. The nth unit simplex $\Delta_n \subseteq \Re^n$ is defined as

$$\Delta_n := \left\{ z \in \Re^n : \sum_{i=1}^n z_i = 1, z_i \geq 0, i = 1, \ldots, n \right\}. \tag{1.7}$$

Let $\mathcal{Z}$ denote a finite dimensional inner product space with inner product and associated norm denoted by $\langle \cdot, \cdot \rangle$ and $\| \cdot \|$. The effective domain of a function $f : \mathcal{Z} \to [-\infty, \infty]$ is defined as $\operatorname{dom} f := \{z \in \mathcal{Z} : f(z) < \infty\}$. The conjugate f^* of f is the function $f^* : \mathcal{Z} \to [-\infty, \infty]$ defined as

$$f^*(v) := \sup_{z\in\mathcal{Z}} \langle v, z\rangle - f(z) \quad \forall v \in \mathcal{Z}.$$

The indicator function $\mathcal{I}_\Omega : \mathcal{Z} \to (-\infty, \infty]$ is defined as

$$\mathcal{I}_\Omega(z) := \begin{cases} 0, & z \in \Omega, \\ \infty, & z \notin \Omega. \end{cases}$$

The *orthogonal projection* $P_\Omega : \mathcal{Z} \to \mathcal{Z}$ onto Ω is defined as

$$P_\Omega(z) := \operatorname{argmin}_{\tilde{z}\in\Omega} \|\tilde{z} - z\| \quad \forall z \in \mathcal{Z}.$$

The domain of a point-to-point map F is denoted by $\operatorname{Dom} F$. For a constant $L \geq 0$, a map $F : \operatorname{Dom} F \subseteq \mathcal{Z} \to \mathcal{Z}$ is said to be L-Lipschitz continuous on $\Omega \subseteq \operatorname{Dom} F$ if

$$\|F(z) - F(\tilde{z})\| \leq L\|z - \tilde{z}\| \quad \forall z, \tilde{z} \in \Omega; \tag{1.8}$$

moreover, if in addition $\Omega = \operatorname{Dom} F$, we will simply say that F is L-Lipschitz continuous. Also, a map $F : \operatorname{Dom} F \subseteq \mathcal{Z} \to \mathcal{Z}$ is said to be L-co-coercive on $\Omega \subseteq \operatorname{Dom} F$ if

$$L\langle z - \tilde{z}, F(z) - F(\tilde{z})\rangle \geq \|F(z) - F(\tilde{z})\|^2 \quad \forall z, \tilde{z} \in \Omega.$$

Clearly, in view of the Cauchy–Schwarz inequality, every L-co-coercive map F on $\Omega \subseteq \operatorname{Dom} F$ is L-Lipschitz continuous on Ω.

A relation $T \subseteq \mathcal{Z} \times \mathcal{Z}$ can be identified with a point-to-set operator $T : \mathcal{Z} \rightrightarrows \mathcal{Z}$ in which

$$T(z) := \{v \in \mathcal{Z} : (z, v) \in T\} \quad \forall z \in \mathcal{Z}.$$

Note that the relation T is then the same as the graph of the point-to-set operator T defined as

$$Gr(T) := \{(z, v) \in \mathcal{Z} \times \mathcal{Z} : v \in T(z)\}.$$

The domain of T, denoted by $\mathrm{Dom}\, T$, is defined as

$$\mathrm{Dom}\, T := \{z \in \mathcal{Z} : T(z) \neq \emptyset\}.$$

An operator $T : \mathcal{Z} \rightrightarrows \mathcal{Z}$ is *monotone* if

$$\langle v - \tilde{v}, z - \tilde{z} \rangle \geq 0 \quad \forall (z, v), (\tilde{z}, \tilde{v}) \in Gr(T).$$

Moreover, T is *maximal monotone* if it is monotone and maximal in the family of monotone operators with respect to the partial order of inclusion, i.e., $S : \mathcal{Z} \rightrightarrows \mathcal{Z}$ monotone and $Gr(S) \supseteq Gr(T)$ implies that $S = T$.

For a scalar $\varepsilon \geq 0$, the ε-subdifferential of a function $f : \mathcal{Z} \to [-\infty, +\infty]$ is the operator $\partial_\varepsilon f : \mathcal{Z} \rightrightarrows \mathcal{Z}$ defined as

$$\partial_\varepsilon f(z) := \{v \mid f(\tilde{z}) \geq f(z) + \langle \tilde{z} - z, v \rangle - \varepsilon,\ \forall \tilde{z} \in \mathcal{Z}\} \quad \forall z \in \mathcal{Z}. \tag{1.9}$$

When $\varepsilon = 0$, the operator $\partial_\varepsilon f$ is simply denoted by ∂f and is referred to as the subdifferential of f. The operator ∂f is trivially monotone if f is proper. If f is a proper lower semicontinuous convex function, then ∂f is maximal monotone [19].

2. BD-HPE framework for CNE problems. This section contains two subsections. The first one describes the two-player Nash equilibrium problem and the notion of a (ρ, ε)-Nash equilibrium. The second subsection introduces the CNE problem and a specialization of the BD-HPE framework [13] to its context and describes iteration-complexity bounds for an instance of the specialized framework to obtain a (ρ, ε)-Nash equilibrium. It also briefly reviews a specific instance of the BD-HPE framework in the context of the CNE problem which was proposed in subsection 5.2 of [13].

2.1. Nash equilibrium and saddle-point problems. Let $\mathcal{X}$ and $\mathcal{Y}$ denote finite dimensional inner product spaces with associated inner products both denoted by $\langle \cdot, \cdot \rangle$ and associated norms both denoted by $\|\cdot\|$. We endow the product space $\mathcal{X} \times \mathcal{Y}$ with the canonical inner product defined as

$$\langle (x, y), (\tilde{x}, \tilde{y}) \rangle := \langle x, \tilde{x} \rangle + \langle y, \tilde{y} \rangle \quad \forall (x, y), (\tilde{x}, \tilde{y}) \in \mathcal{X} \times \mathcal{Y}.$$

The associated norm, also denoted by $\|\cdot\|$ for shortness, is then given by

$$\|(x, y)\| := \sqrt{\|x\|^2 + \|y\|^2} \quad \forall (x, y) \in \mathcal{X} \times \mathcal{Y}.$$

We next introduce the Nash equilibrium problem. Let nonempty closed convex sets $X \subseteq \mathcal{X}$ and $Y \subseteq \mathcal{Y}$ be given and consider functions $\widehat{\Psi}_1 : X \times Y \to \Re$ and $\widehat{\Psi}_2 : X \times Y \to \Re$. The (two-player) Nash equilibrium problem determined by $(\widehat{\Psi}_1, \widehat{\Psi}_2)$, denoted by $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$, consists of finding a pair $(x, y) \in X \times Y$ such that

$$\widehat{\Psi}_1(x, y) \leq \widehat{\Psi}_1(\tilde{x}, y), \quad \widehat{\Psi}_2(x, y) \leq \widehat{\Psi}_2(x, \tilde{y}) \quad \forall (\tilde{x}, \tilde{y}) \in X \times Y. \tag{2.1}$$

Clearly, (x, y) is a Nash equilibrium of $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$ if and only if $(x, y) \in X \times Y$ and

$$(0, 0) \in \partial[\widehat{\Psi}_1(\cdot, y) + \widehat{\Psi}_2(x, \cdot)](x, y). \tag{2.2}$$

Let $\phi_1 : Y \to [-\infty, \infty)$ and $\phi_2 : X \to [-\infty, \infty)$ be defined as

$$\phi_1(y) := \inf_{\tilde{x} \in X} \widehat{\Psi}_1(\tilde{x}, y), \quad \phi_2(x) := \inf_{\tilde{y} \in Y} \widehat{\Psi}_2(x, \tilde{y}) \quad \forall (x, y) \in X \times Y. \tag{2.3}$$

Clearly,

$$\mathrm{gap}(x, y) := \widehat{\Psi}_1(x, y) + \widehat{\Psi}_2(x, y) - \phi_1(y) - \phi_2(x) \geq 0 \quad \forall (x, y) \in X \times Y. \tag{2.4}$$

Moreover, (x, y) is a Nash equilibrium if and only if $\mathrm{gap}(x, y) = 0$ or, equivalently, (x, y) is an optimal solution with optimal value equal zero for the problem of minimizing the gap function $\mathrm{gap}(\tilde{x}, \tilde{y})$ on $X \times Y$.

Now we introduce the following definition of an approximate Nash equilibrium for $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$.

DEFINITION 2.1. *Given $(\rho, \varepsilon) \in \Re_+ \times \Re_+$, $z = (x, y) \in X \times Y$, $v \in \mathcal{X} \times \mathcal{Y}$, and $\tilde{\varepsilon} \in \Re_+$, the triple $(z, v, \tilde{\varepsilon})$ is called a (ρ, ε)-Nash equilibrium of $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$ if $\|v\| \leq \rho$, $\tilde{\varepsilon} \leq \varepsilon$ and*

$$v \in \partial_{\tilde{\varepsilon}}[\widehat{\Psi}_1(\cdot, y) + \widehat{\Psi}_2(x, \cdot)](x, y). \tag{2.5}$$

Moreover, any such pair $(v, \tilde{\varepsilon})$ will be called an NE-residual *for (x, y) with respect to $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$.*

It is worth pointing out the relationship between a (ρ, ε)-Nash equilibrium and the more popular notion of an ε-Nash equilibrium. Recall that $z = (x, y) \in X \times Y$ is called an ε-Nash equilibrium if it satisfies (2.5) with $(v, \tilde{\varepsilon}) = (0, \varepsilon)$. Clearly, $z = (x, y)$ is an ε-Nash equilibrium if and only if $\mathrm{gap}(x, y) \leq \varepsilon$. Note that the use of the latter condition as a means of determining whether a pair (x, y) is an ε-Nash equilibrium assumes that the functions ϕ_1 and ϕ_2 defined in (2.3) can be easily evaluated at (x, y). However, there are many applications for which the latter two functions are not necessarily easy to evaluate (see, for example, subsections 5.1, 5.2, and 5.3). On the other hand, under the assumption that $X \times Y$ is bounded, the following result shows that an algorithm which generates a sequence of iterates $\{(z_k, v_k, \tilde{\varepsilon}_k)\}$ such that $(z, v, \tilde{\varepsilon}) = (z_k, v_k, \tilde{\varepsilon}_k)$ satisfies (2.5) and $\{(v_k, \tilde{\varepsilon}_k)\}$ converges to zero can easily compute and detect an ε-Nash equilibrium.

LEMMA 2.2. *Assume that $X \times Y$ is bounded and that $z = (x, y) \in X \times Y$, $v \in \mathcal{X} \times \mathcal{Y}$, and $\tilde{\varepsilon} \in \Re_+$ are such that $(z, v, \tilde{\varepsilon})$ satisfies* (2.5). *Then, z is an ε'-Nash equilibrium of $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$, where*

$$\varepsilon' = \varepsilon'(z, v, \tilde{\varepsilon}) := \tilde{\varepsilon} + \max_{(\tilde{x}, \tilde{y}) \in X \times Y} \langle v, (x - \tilde{x}, y - \tilde{y}) \rangle.$$

Hence, under the assumption that $X \times Y$ is bounded, it follows from Lemma 2.2 that if the goal is to compute an ε-Nash equilibrium, then one can evaluate $\varepsilon_k := \varepsilon'(z_k, v_k, \tilde{\varepsilon}_k)$ and terminate the aforementioned hypothetical algorithm whenever $\varepsilon_k \leq \varepsilon$. Note that the assumption that $X \times Y$ is bounded guarantees that ε_k is finite for every k. Moreover, since the hypothetical algorithm generates $\{(z_k, v_k, \tilde{\varepsilon}_k)\}$ in such a way that $\{(v_k, \tilde{\varepsilon}_k)\}$ converges to zero, it will always find an ε-Nash equilibrium in this manner.

We now make some comments about the notion of a (ρ, ε)-Nash equilibrium and its specialization to the context of saddle-point problems (see, for example, subsection 3.2 of [12]), which also pertains to the case where $X \times Y$ is unbounded. First, being weaker than the notion of an ε-Nash equilibrium in the sense that v can be nonzero, it is suitable for analyzing many algorithms for solving saddle-point and

Nash equilibrium problems. In particular, it is a natural notion to consider in the context of HPE-type algorithms such as the ones studied in this paper since they generate a sequence $\{(z_k, v_k, \tilde{\varepsilon}_k)\}$ such that $(z, v, \tilde{\varepsilon}) = (z_k, v_k, \tilde{\varepsilon}_k)$ satisfies (2.5) for every k. Second, although it is based on two errors, namely, v and $\tilde{\varepsilon}$, instead of just one single scalar error, these errors naturally arise in the sense that v usually expresses the infeasibility error while ε expresses some sort of functional gap (see, for example, section 3 of [11]).

We observe that the more familiar saddle-point problem is the special case of the Nash equilibrium problem in which $\hat{\Psi}_1 = -\hat{\Psi}_2$. Naturally, a Nash equilibrium is referred to as a saddle point in this context. Observe that in such a case, (2.4) reduces to $-\phi_2(x) \geq \phi_1(y)$ for every $(x, y) \in X \times Y$, which is the well-known weak duality result that says that the primal value $p(x) := -\phi_2(x)$ is greater than or equal to the dual value $d(y) := \phi_1(y)$ for any $(x, y) \in X \times Y$. Moreover, (x, y) is a saddle point (resp., ε-saddle point) if and only if $(x, y) \in X \times Y$ and $p(x) = d(y)$ (resp., $p(x) - d(y) \leq \varepsilon$). In view of the weak duality, the latter condition is equivalent to $x \in X$ and $y \in Y$ being optimal solutions of $p_* := \inf_{\tilde{x} \in X} p(\tilde{x})$ and $d_* := \sup_{\tilde{y} \in Y} d(\tilde{y})$, respectively, and the optimal duality gap $p_* - d_*$ being equal to zero.

2.2. A BD framework for CNE problems. This subsection describes the CNE problem and a specialization of the BD-HPE framework of [13] to the CNE context, which we refer to as the CNE-BD-HPE framework. It also establishes the iteration-complexity for the latter framework to find a (ρ, ε)-Nash equilibrium. Moreover, it briefly discusses the generic problem underlying the prox subinclusions of the CNE-BD-HPE framework and describes a recipe for solving it, when the stepsize is sufficiently small, based on performing a single composite gradient step on (1.5). Finally, it briefly describes a specific CNE-BD-HPE instance based on this recipe and its corresponding ergodic iteration-complexity. This instance can be viewed as a BD version of Tseng's MFBS algorithm in [26] (see also [12, 11]). Except for our slightly more general way of choosing the stepsize, this instance is the same as the method stated in subsection 5.2 of [13] when the latter one is specialized to the context of (1.1).

We start by describing our problem of interest in this paper, namely, the Nash equilibrium problem endowed with a composite structure. More specifically, consider a Nash equilibrium problem $NE(\widehat{\Psi}_1, \widehat{\Psi}_2)$, where $\operatorname{Dom} \widehat{\Psi}_1 = \operatorname{Dom} \widehat{\Psi}_2 = X \times Y$ (see subsection 2.1). The composite structure consists of the existence of proper closed convex functions $g_i : \mathcal{X} \to (-\infty, \infty]$, $i = 1, 2$, and functions $\Psi_i : \operatorname{Dom} \Psi_i \subseteq \mathcal{X} \times \mathcal{Y} \to \Re$, $i = 1, 2$, satisfying

$$
\begin{aligned}
&\operatorname{dom} g_1 \times \operatorname{dom} g_2 = X \times Y \subseteq \operatorname{Dom} \Psi_1 \cap \operatorname{Dom} \Psi_2, \\
&\widehat{\Psi}_1(x, y) = \Psi_1(x, y) + g_1(x) - g_2(y), \\
&\widehat{\Psi}_2(x, y) = \Psi_2(x, y) + g_2(y) - g_1(x) \quad \forall (x, y) \in X \times Y
\end{aligned} \tag{2.6}
$$

and (or some of) the following additional conditions:

A.1. There exists a closed convex set $\Omega_x \times \Omega_y \supseteq \operatorname{dom} g_1 \times \operatorname{dom} g_2$ such that Ψ_1 and Ψ_2 are differentiable on $\Omega_x \times \Omega_y$.

A.2. There exists $L_{xy} > 0$ such that

$$
\|\nabla_x \Psi_1(x, \tilde{y}) - \nabla_x \Psi_1(x, y)\| \leq L_{xy} \|\tilde{y} - y\| \quad \forall x \in \Omega_x, \, \forall y, \tilde{y} \in \Omega_y.
$$

A.3. The map $(x, y) \mapsto (\nabla_x \Psi_1(x, y), \nabla_y \Psi_2(x, y)) \in \mathcal{X} \times \mathcal{Y}$ is monotone on $X \times Y$.

A.3′. $\Psi_1(x,\cdot)+\Psi_2(\cdot,y)$ is concave on $X\times Y$ for every $(x,y)\in X\times Y$ and $\Psi_1+\Psi_2$ is convex on $X\times Y$.

For the sake of future reference, we denote the Nash equilibrium problem $NE(\widehat{\Psi}_1,\widehat{\Psi}_2)$ endowed with the above composite structure by $CNE(\Psi_1,\Psi_2;g_1,g_2)$ and refer to it as the CNE problem.

We now make some remarks about the above conditions. First, if condition A.1 holds, then A.3′ implies A.3 (see Proposition A.1 in Appendix A) but the reverse implication does not hold, as the following example shows: $\Psi_1(x,y)=x^2-xy$ and $\Psi_2(x,y)=-3xy+y^2$ for every $(x,y)\in\Re^2$. Hence, A.3′ is a stronger condition than A.3 and it will be used in the derivation of ergodic complexity bounds for the CNE-BD-HPE framework discussed later in this subsection. Second, the aforementioned reverse implication holds in the special case of CSP problems (i.e., when $\Psi_1=-\Psi_2$).

In view of (2.2), $CNE(\Psi_1,\Psi_2;g_1,g_2)$ is equivalent to (1.1), which is a maximal monotone inclusion of the same type considered in [13] with the exception that $F_1(x,y):=\nabla_x\Psi_1(x,y)$ and $F_2(x,y):=\nabla_y\Psi_2(x,y)$ are defined in $\Omega_x\times\Omega_y$ instead of $\Omega_x\times\mathcal{Y}$ as in [13]. This issue can be resolved by extending $F_i(x,y)$, $i=1,2$, to $\Omega_x\times\mathcal{Y}$ as $(x,y)\in\Omega_x\times\mathcal{Y}\mapsto F_i(x,P_{\Omega_y}(y))$. The latter extension can then be shown to satisfy all the conditions assumed in [13] and, as a consequence, we can use the BD-HPE framework studied in [13] to solve (1.1), and hence $CNE(\Psi_1,\Psi_2;g_1,g_2)$.

We next state a special case of the BD-HPE framework of [13], referred to as the CNE-BD-HPE framework, specialized to the context of (1.1). In contrast to [13], it assumes for simplicity that the proximal stepsizes are constant and denoted by λ. Also, in contrast to the BD-HPE framework of [13] which uses the ε-enlargement of ∂g_i, the CNE-BD-HPE framework works with the ε-subdifferential of ∂g_i, which is known to be a smaller enlargement than the first one.

(CNE-BD-HPE) BD-HPE framework for solving CNE$(\boldsymbol{\Psi_1},\boldsymbol{\Psi_2};\mathbf{g_1},\mathbf{g_2})$.

0. Let $(x_0,y_0)\in\mathcal{X}\times\mathcal{Y}$, $\sigma\in(0,1]$, and $\sigma_x,\sigma_y\in[0,\sigma)$ be given, choose $\lambda>0$ such that
$$\lambda\le\frac{\sqrt{(\sigma^2-\sigma_x^2)(\sigma^2-\sigma_y^2)}}{\sigma L_{xy}},\tag{2.7}$$
and set $k=1$;
1. compute a triple $(\tilde{x},\tilde{a},\varepsilon^x)\in\mathcal{X}\times\mathcal{X}\times\Re_+$ such that
$$\tilde{a}\in\partial_{\varepsilon^x}g_1(\tilde{x}),\quad \|\lambda(\nabla_x\Psi_1(\tilde{x},y'_{k-1})+\tilde{a})+\tilde{x}-x_{k-1}\|^2+2\lambda\varepsilon^x\le\sigma_x^2\|\tilde{x}-x_{k-1}\|^2,\tag{2.8}$$
where $y'_{k-1}=P_{\Omega_y}(y_{k-1})$, and set $(\tilde{x}_k,\tilde{a}_k,\varepsilon_k^x)=(\tilde{x},\tilde{a},\varepsilon^x)$;
2. compute a triple $(\tilde{y},\tilde{b},\varepsilon^y)\in\mathcal{Y}\times\mathcal{Y}\times\Re_+$ such that
$$\tilde{b}\in\partial_{\varepsilon^y}g_2(\tilde{y}),\quad \|\lambda(\nabla_y\Psi_2(\tilde{x}_k,\tilde{y})+\tilde{b})+\tilde{y}-y_{k-1}\|^2+2\lambda\varepsilon^y\le\sigma_y^2\|\tilde{y}-y_{k-1}\|^2,\tag{2.9}$$
and set $(\tilde{y}_k,\tilde{b}_k,\varepsilon_k^y)=(\tilde{y},\tilde{b},\varepsilon^y)$;
3. let $(\tilde{v}_k^x,\tilde{v}_k^y)=(\nabla_x\Psi_1(\tilde{x}_k,\tilde{y}_k)+\tilde{a}_k,\nabla_y\Psi_2(\tilde{x}_k,\tilde{y}_k)+\tilde{b}_k)$, set
$$(x_k,y_k)=(x_{k-1},y_{k-1})-\lambda(\tilde{v}_k^x,\tilde{v}_k^y),$$
and $k\leftarrow k+1$, and go to step 1.

end

We now state the convergence rate result for the CNE-BD-HPE framework whose proof uses Theorems 3.2 and 3.3 of [13] and arguments similar to (but more general than) the ones used in section 5 of [12]. For the sake of completeness, its proof is given in Appendix B.

THEOREM 2.3. *Assume that conditions* A.1 *and* A.2 *hold and consider the sequences* $\{(\tilde{x}_k, \tilde{y}_k)\}$, $\{(\tilde{v}_k^x, \tilde{v}_k^y)\}$, *and* $\{(\varepsilon_k^x, \varepsilon_k^y)\}$ *generated by the CNE-BD-HPE framework. For every* $k \in \mathbb{N}$, *define*

$$(\tilde{x}_k^a, \tilde{y}_k^a) := \frac{1}{k}\sum_{i=1}^{k}(\tilde{x}_i, \tilde{y}_i), \qquad \tilde{v}_k^a := \frac{1}{k}\sum_{i=1}^{k}(\tilde{v}_i^x, \tilde{v}_i^y), \tag{2.10}$$

and

$$\tilde{\varepsilon}_k^a := \frac{1}{k}\sum_{i=1}^{k}[\varepsilon_i^x + \varepsilon_i^y + \langle(\tilde{x}_i - \tilde{x}_k^a, \tilde{y}_i - \tilde{y}_k^a), (\tilde{v}_i^x, \tilde{v}_i^y)\rangle], \tag{2.11}$$

and let d_0 *denote the distance of* (x_0, y_0) *to the set of Nash equilibriums of* $CNE(\Psi_1, \Psi_2; g_1, g_2)$. *Then, for every* $k \in \mathbb{N}$, *the following statements hold:*

(a) *if* A.3 *holds, then the pair* $((\tilde{v}_k^x, \tilde{v}_k^y), \varepsilon_k^x + \varepsilon_k^y)$ *is an NE-residual for* $(\tilde{x}_k, \tilde{y}_k)$ *and there exists* $i \le k$ *such that*

$$\|(\tilde{v}_i^x, \tilde{v}_i^y)\| \le \frac{d_0}{\lambda}\sqrt{\frac{1+\sigma}{k(1-\sigma)}}, \quad \varepsilon_i^x + \varepsilon_i^y \le \frac{\sigma^2 d_0^2}{2(1-\sigma^2)k\lambda}; \tag{2.12}$$

(b) *if* A.3′ *holds, then the pair* $(\tilde{v}_k^a, \tilde{\varepsilon}_k^a)$ *is an NE-residual for* $(\tilde{x}_k^a, \tilde{y}_k^a)$ *and*

$$\|\tilde{v}_k^a\| \le \frac{2d_0}{k\lambda}, \qquad \tilde{\varepsilon}_k^a \le \frac{2d_0^2}{k\lambda}(1+\bar{\eta}), \tag{2.13}$$

where

$$\bar{\eta} := \frac{2\sqrt{2}\sigma}{1-\max(\sigma_x, \sigma_y)}\left(1 + \frac{1}{(1-\sigma_y)^2}\right)^{1/2}. \tag{2.14}$$

We end this section by describing a generic problem underlying the computation of the triples as in steps 1 and 2 of the CNE-BD-HPE framework. Let $\mathcal{Z}$ be an inner product space and $\tilde{f} : \operatorname{Dom}\tilde{f} \to \Re$ and $\tilde{h} : \mathcal{Z} \to (-\infty, \infty]$ be functions such that

B.1. $\tilde{h}$ is a proper closed convex function;

B.2. $\tilde{f}$ is differentiable and convex on a nonempty closed convex set $\Omega \supseteq \operatorname{dom}\tilde{h}$.

The generic problem mentioned above is as follows:

> ($P0$) Given $w_0 \in \mathcal{Z}$, $\lambda > 0$, and $\sigma_z \ge 0$, find a triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$ such that (1.6) is satisfied.

We now make two remarks about ($P0$). First, steps 1 and 2 of the CNE-BD-HPE framework are clearly special cases of the above generic problem. Indeed, steps 1 and 2 are special cases of ($P0$) in which

$$\tilde{f}(\cdot) := \Psi_1(\cdot, y'_{k-1}) : \Omega_x \mapsto \Re, \quad \tilde{h} = g_1, \quad \Omega = \Omega_x, \tag{2.15}$$

$$\tilde{f}(\cdot) := \Psi_2(\tilde{x}_k, \cdot) : \Omega_y \mapsto \Re, \quad \tilde{h} = g_2, \quad \Omega = \Omega_y, \tag{2.16}$$

respectively. Second, the above problem is related to the problem of finding an approximate solution of the (strongly) convex optimization problem (1.5). Clearly, an exact solution $\tilde{z}$ of (1.5) satisfies $0 \in \tilde{z} - w_0 + \lambda(\nabla \tilde{f}(\tilde{z}) + \partial \tilde{h}(\tilde{z}))$, and hence the triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$, where $\tilde{s} = (w_0 - \tilde{z})/\lambda - \nabla \tilde{f}(\tilde{z})$ and $\varepsilon = 0$ satisfies (1.6) with $\sigma_z = 0$. Therefore, the situation in which (1.5) can be solved exactly immediately yields a solution of problem $(P0)$.

The following result, whose proof can be found in Proposition 4.3 of [13], shows that a single composite gradient step from w_0 with respect to (1.5) yields a solution of $(P0)$ when $\nabla \tilde{f}$ is Lipschitz continuous on Ω, $\lambda > 0$ is sufficiently small, and the resolvents of $\partial \tilde{h}$, i.e., vectors of the form

$$(I + \lambda \partial \tilde{h})^{-1}(z) = \operatorname{argmin}_{u \in \mathcal{Z}} \left\{ \tilde{h}(u) + \frac{1}{2\lambda} \|u - z\|^2 \right\} \quad \forall z \in \mathcal{Z},$$

can be easily computed. Clearly, when $\tilde{h}$ is the indicator of a nonempty closed convex set $\Omega \subseteq \mathcal{Z}$, we have $(I + \lambda \partial \tilde{h})^{-1}(\cdot) = P_\Omega(\cdot)$ for any $\lambda > 0$.

PROPOSITION 2.4. *For some $\tilde{L} \geq 0$, assume that $\nabla \tilde{f}$ is $\tilde{L}$-Lipschitz continuous on Ω. Then, for any $w_0 \in \mathcal{Z}$, $\sigma_z \geq 0$, and $\lambda > 0$ such that $\lambda \tilde{L} \leq \sigma_z$, the triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$ given by*

(2.17)
$$\tilde{z} := (I + \lambda \partial \tilde{h})^{-1} \left(w_0 - \lambda \nabla \tilde{f}(P_\Omega(w_0)) \right), \quad \tilde{s} := \frac{1}{\lambda}(w_0 - \tilde{z}) - \nabla \tilde{f}(P_\Omega(w_0)), \quad \tilde{\varepsilon} := 0,$$

solves problem $(P0)$.

In addition to conditions A.1, A.2, and A.3′, we further assume that the following two additional conditions hold:

A.4. There exists $L_{xx} \geq 0$ such that $\nabla_x \Psi_1(\cdot, y)$ is L_{xx}-co-coercive on Ω_x for every $y \in \Omega_y$.

A.5. There exists $L_{yy} \geq 0$ such that $\nabla_y \Psi_2(x, \cdot)$ is L_{yy}-co-coercive on Ω_y for every $x \in \Omega_x$.

Since L-co-coercive maps are L-Lipschitz continuous, it follows from A.4. and A.5. that the maps $\tilde{f}$ in (2.15) and (2.16) satisfy the assumption of Proposition 2.4. As a result, if we set

$$\lambda = \bar{\lambda}(\sigma_x, \sigma_y) := \min \left\{ \frac{\sigma_x}{L_{xx}}, \frac{\sigma_y}{L_{yy}}, \frac{\sqrt{(\sigma^2 - \sigma_x^2)(\sigma^2 - \sigma_y^2)}}{\sigma L_{xy}} \right\}, \tag{2.18}$$

we can use the constructive recipe of Proposition 2.4 to obtain the triples $(\tilde{x}_k, \tilde{a}_k, \varepsilon_k^x)$ and $(\tilde{y}_k, \tilde{b}_k, \varepsilon_k^y)$ as in steps 1 and 2 of the CNE-BD-HPE framework. For the sake of future reference, we refer to the special instance of the CNE-BD-HPE framework which generates the triples $(\tilde{x}_k, \tilde{a}_k, \varepsilon_k^x)$ and $(\tilde{y}_k, \tilde{b}_k, \varepsilon_k^y)$ in this manner as the Tseng-BD algorithm. Using Definition 2.1 and Theorem 2.3(b), we easily see that for every pair of positive scalars (ρ, ε), there exists an index

$$k_0 = \mathcal{O}\left(\max\{L_{xx}, L_{xy}, L_{yy}\} \max\left[\frac{d_0^2}{\varepsilon}, \frac{d_0}{\rho} \right] \right) \tag{2.19}$$

such that for every $k \geq k_0$, the kth ergodic iterate $(\tilde{x}_k^a, \tilde{y}_k^a)$ generated by the Tseng-BD algorithm is a (ρ, ε)-Nash equilibrium of $CNE(\Psi_1, \Psi_2; g_1, g_2)$ and the pair $(\tilde{v}_k^a, \tilde{\varepsilon}_k^a)$ is a NE residual for $(\tilde{x}_k^a, \tilde{y}_k^a)$.

Observe that the third bound in (2.18) is due to condition (2.7), while the first two bounds guarantee that λ satisfies the assumption of Proposition 2.4 for the two $(P0)$ instances described in (2.15) and (2.16). Clearly, the stepsize λ given by (2.18) is $1/\mathcal{O}(M)$, where $M := \max\{L_{xx}, L_{xy}, L_{yy}\}$. On the other hand, the largest stepsize λ that can be chosen in the context of the CNE-BD-HPE framework, i.e., the right-hand side of (2.7), is $1/\mathcal{O}(L_{xy})$. Clearly, when $M >> L_{xy}$, or equivalently $\max\{L_{xx}, L_{yy}\} >> L_{xy}$, the latter stepsize is considerably larger than the first one. As a consequence, the number of iterations performed by a CNE-BD-HPE instance which sets λ equal to the right-hand side of (2.7) can be considerably smaller than the number of iterations performed by the Tseng-BD algorithm which chooses λ as in (2.18). Clearly, a CNE-BD-HPE instance which sets λ equal to the right-hand side of (2.7) requires an approach for solving $(P0)$ that is different from the one described in Proposition 2.4. This alternative approach will be the subject of our study in the next section.

3. An accelerated method for problem $(P0)$. The main goal of this section is to derive the iteration-complexity of solving problem $(P0)$ using a variant of the Nesterov's accelerated method [17] applied to (1.5). This section contains three subsections. The first subsection introduces two new problems and discusses their relationship with $(P0)$. The second subsection discusses a variant of the accelerated method introduced by Nesterov in [17] for minimizing a general convex composite function. Finally, the third subsection establishes the iteration-complexities of solving the two problems related to $(P0)$ and, as a consequence, problem $(P0)$ itself, using the aforementioned variant applied to (1.5).

Let $\mathcal{Z}$ denote a finite dimensional inner product space with inner product and associated norm denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$. Throughout this section, we assume that f and h are functions satisfying the following conditions:

C.1. h is a proper closed convex function.

C.2. f is differentiable on a closed convex set $\Omega \supseteq \operatorname{dom} h$.

C.3. There exists $L > 0$ such that ∇f is L-co-coercive on Ω.

Note that C.3 implies that ∇f is L-Lipschitz continuous on Ω, which in turn implies that

$$0 \le f(u') - f(u) - \langle \nabla f(u), u' - u\rangle \le \frac{L}{2}\|u' - u\|^2 \quad \forall u, u' \in \Omega. \tag{3.1}$$

3.1. Generalization and related formulations of problem $(P0)$. This subsection introduces a more general version of $(P0)$, as well as a relaxed version of the new problem. It also discusses the relationship between these three problems and, in particular, how solutions to the relaxed version yield solutions to $(P0)$.

We start by introducing the following generalization of $(P0)$:

$(P1)$ Given $w_0 \in \mathcal{Z}$ and $\tau_1 > 0$, find a triple $(z, v, \varepsilon) \in \mathcal{Z} \times \mathcal{Z} \times \mathbb{R}_+$ such that

$$v \in \nabla f(z) + \partial_\varepsilon h(z), \quad \|v\|^2 + 2\varepsilon \le \tau_1 \|z - w_0\|^2. \tag{3.2}$$

The following simple result shows that a solution of a specific instance of $(P1)$ yields a solution of $(P0)$.

PROPOSITION 3.1. *Let $\tilde{f}$ and $\tilde{h}$ satisfy conditions* B.1 *and* B.2 *and $(w_0, \lambda, \sigma_z) \in \mathcal{Z} \times \mathbb{R}_{++} \times \mathbb{R}_{++}$ determine an instance of $(P0)$ and define the functions f and h as*

$$f(u) := \lambda \tilde{f}(u) + \frac{1}{2}\|u - w_0\|^2, \quad h(u) := \lambda \tilde{h}(u) \quad \forall u \in \mathcal{Z}. \tag{3.3}$$

Then, if (z, v, ε) *is a solution of the instance of* $(P1)$ *with* f *and* h *as above and* $\tau_1 = \sigma_z^2$*, then the triple* $(\tilde{z}, \tilde{s}, \tilde{\varepsilon})$ *defined as*

$$\tilde{z} := z, \quad \tilde{s} := \frac{v - (z - w_0)}{\lambda} - \nabla \tilde{f}(z), \quad \tilde{\varepsilon} := \frac{\varepsilon}{\lambda},$$

is a solution of the above instance of $(P0)$.

Proof. The proof follows immediately from the definition of f and h and the fact that $\lambda \partial_{\tilde{\varepsilon}} \tilde{h}(u) = \partial_{\lambda\tilde{\varepsilon}}(\lambda \tilde{h})(u)$ for every $u \in \mathcal{Z}$. ∎

It follows from the above result that any algorithm that solves $(P1)$ can also be used to solve $(P0)$. On the other hand, problem $(P1)$ is naturally related to the following weaker problem:

$(P2)$ Given $w_0 \in \mathcal{Z}$ and $\tau_2 > 0$, find a triple $(z, r, \varepsilon) \in \mathcal{Z} \times \mathcal{Z} \times \mathbb{R}_+$ such that

$$r \in \partial_\varepsilon (f + h)(z), \quad \|r\|^2 + 2\varepsilon \le \tau_2 \|z - w_0\|^2. \tag{3.4}$$

Due to the inclusion $\nabla f(z) + \partial_\varepsilon h(z) \subseteq \partial_\varepsilon (f + h)(z)$, any solution of $(P1)$ is also a solution of $(P2)$ with $\tau_2 = \tau_1$, thereby showing that $(P2)$ is a weaker version of $(P1)$. However, we will show below that a solution of $(P2)$ with τ_2 sufficiently small can be used to construct a solution of $(P1)$.

LEMMA 3.2. *Assume that* f *and* h *satisfy* C.1, C.2, *and* C.3. *Then, if* $(z, r, \varepsilon) \in \mathcal{Z} \times \mathcal{Z} \times \Re_+$ *satisfy*

$$r \in \partial_\varepsilon (f + h)(z), \tag{3.5}$$

then for any positive scalar $c > L/2$*, the vector*

$$\delta_c = \delta(z, r, c) := c[z - (I + c^{-1}\partial h)^{-1}(z - c^{-1}\nabla f(z) + c^{-1} r)] \tag{3.6}$$

satisfies

$$r + \delta_c \in (\nabla f + \partial_\varepsilon h)(z), \quad \|\delta_c\| \le c\sqrt{\frac{2\varepsilon}{2c - L}}. \tag{3.7}$$

Proof. First note that the assumptions imply that $z \in \operatorname{dom} h \subseteq \Omega$. It is easy to see that (3.6) implies that

$$z - c^{-1}\delta_c \in \operatorname{dom} h, \quad r + \delta_c - \nabla f(z) \in \partial h(z - c^{-1}\delta_c).$$

The above inclusion implies

$$h(u) - h(z - c^{-1}\delta_c) \ge \langle r + \delta_c - \nabla f(z), u - z + c^{-1}\delta_c \rangle \quad \forall u \in \mathcal{Z}. \tag{3.8}$$

On the other hand, it follows from (3.5) that

$$f(u) + h(u) \ge f(z) + h(z) + \langle r, u - z \rangle - \varepsilon \quad \forall u \in \mathcal{Z}.$$

This inequality with $u = z - c^{-1}\delta_c$ and (3.1) with $u' = z$ and $u = z - c^{-1}\delta$ then imply that

$$\begin{aligned} h(z - c^{-1}\delta_c) - h(z) &\ge -[f(z - c^{-1}\delta_c) - f(z) - \langle \nabla f(z), -c^{-1}\delta_c \rangle] \\ &\quad + \langle r - \nabla f(z), -c^{-1}\delta_c \rangle - \varepsilon \\ &\ge -\frac{L}{2}\|c^{-1}\delta_c\|^2 + \langle r - \nabla f(z), -c^{-1}\delta_c \rangle - \varepsilon. \end{aligned} \tag{3.9}$$

Adding up (3.8) and (3.9), we conclude that

$$h(u) - h(z) \geq \langle r + \delta_c - \nabla f(z), u - z \rangle + c^{-2}\left(c - \frac{L}{2}\right)\|\delta_c\|^2 - \varepsilon \quad \forall u \in \mathcal{Z},$$

which clearly implies the inclusion in (3.7) when $c \geq L/2$. Moreover, this same inequality with $u = z$ implies the inequality in (3.7) when $c > L/2$. ∎

PROPOSITION 3.3. *If (z, r, ε) is a solution of problem $(P2)$, then the triple (z, v, ε) where*

$$v := r + \delta, \quad \delta := \delta(z, r, L) \tag{3.10}$$

is a solution of problem $(P1)$ with $\tau_1 = 2(L+1)\tau_2$.

Proof. The assumption of the proposition implies that (3.4) holds. It then follows from Lemma 3.2 with $c = L$ and the definition of v that the inclusion in (3.2) holds and $\delta \leq \sqrt{2L\varepsilon}$. Hence, we conclude that

$$\begin{aligned}\|v\|^2 + 2\varepsilon = \|r + \delta\|^2 + 2\varepsilon &\leq 2\left(\|r\|^2 + \|\delta\|^2\right) + 2\varepsilon \leq 2(\|r\|^2 + 2L\varepsilon) + 2\varepsilon \\ &\leq 2(L+1)\tau_2\|z - w_0\|^2,\end{aligned}$$

where the last inequality is due to (3.4). We have thus proved that (z, v, ε) is a solution of problem $(P1)$ with $\tau_1 = 2(L+1)\tau_2$. ∎

Proposition 3.3 implies that a solution of $(P1)$ can be obtained by a suitable solution of $(P2)$ and an additional evaluation of the resolvent $(I + L^{-1}\partial h)^{-1}$ of ∂h. In the next subsection, we discuss how an accelerated gradient variant applied to the composite optimization problem

$$\phi_* := \min_{u \in \mathcal{Z}} \phi(u) := f(u) + h(u) \tag{3.11}$$

immediately yields a solution of $(P2)$ and hence $(P1)$ in view of Proposition 3.3.

3.2. Accelerated method for minimizing strongly convex composite functions. This subsection describes a variant of the accelerated method introduced by Nesterov in [17] for minimizing a (possibly, strongly) convex composite function, i.e., a function of the form (3.11).

In what follows, we refer to convex functions as 0-strongly convex functions. This terminology has the benefit of allowing us to treat both the convex and strongly convex cases simultaneously. In addition to assuming that conditions C.1–C.3 hold, we also assume the following condition:

C.4. For some known constant $\mu \geq 0$, the function ϕ is μ-strongly convex.

We now study a variant of the accelerated algorithm introduced by Nesterov in [17]. Similar to the variant in [17], it is based on an aggressive update of the accelerated parameters (see (3.12) below). However, in contrast to the first one, the latter one performs one less resolvent per iteration and can start from an arbitrary (instead of feasible) initial point. The last feature is important when the new variant is used in the context of the accelerated BD-HPE presented in section 4.

Algorithm 1. A variant of the accelerated algorithm of [17].

0. Let $w_0 \in \mathcal{Z}$ be given and set $A_0 = 0, z_0 = u_0 = P_\Omega(w_0)$ and $k = 1$;
1. let $A_k > A_{k-1}$ be such that

$$2A_k(A_{k-1}\mu + 1) = L(A_k - A_{k-1})^2 \tag{3.12}$$

and compute

$$\tilde{u}_k := \frac{A_{k-1}}{A_k} z_{k-1} + \left(1 - \frac{A_{k-1}}{A_k}\right) u_{k-1}, \tag{3.13}$$

$$z_k := \operatorname{argmin}_{u \in \mathcal{Z}} \left\{ f(\tilde{u}_k) + \langle \nabla f(\tilde{u}_k), u - \tilde{u}_k \rangle + h(u) + \frac{L}{2}\|u - \tilde{u}_k\|^2 \right\}, \tag{3.14}$$

$$q_k := L(\tilde{u}_k - z_k) + \nabla f(z_k) - \nabla f(\tilde{u}_k), \tag{3.15}$$

$$w_k := \frac{A_{k-1}\mu + 1}{A_k\mu + 1} w_{k-1} + \frac{A_k - A_{k-1}}{A_k\mu + 1}(\mu z_k - q_k), \tag{3.16}$$

$$u_k := P_\Omega(w_k); \tag{3.17}$$

2. compute

$$r_k := \frac{1}{A_k}(w_0 - u_k), \quad \delta_k := \delta(z_k, r_k, L), \tag{3.18}$$

$$\varepsilon_k := \frac{1}{2A_k}\|z_k - w_0\|^2 - \frac{1}{2A_k}\|z_k - u_k\|^2, \quad v_k := r_k + \delta_k, \tag{3.19}$$

where $\delta(\cdot, \cdot, \cdot)$ is defined in (3.6);

3. set $k \leftarrow k + 1$ and go to step 1.

end

We now make a few remarks about Algorithm 1. First, if one is interested in finding an ε-solution of (3.11), i.e., a point $z \in \mathcal{Z}$ such that $0 \in \partial_\varepsilon(f + h)(z)$, then there is no need to perform step 2, whose only purpose is to generate a solution of $(P1)$. Second, Algorithm 1 (with step 2 included) requires two gradient evaluations (see (3.15)), two resolvent evaluations, i.e., one in (3.14) and one in (3.18), and one projection onto Ω. Third, step 2 can be performed every fixed number of iterations in order to save one resolvent evaluation at those iterations which skip this step.

It is also worthwhile comparing Algorithm 1 (without step 2 included) with Nesterov's accelerated method in [17]. First, while both methods require two gradient evaluations per iteration, the first one requires one resolvent evaluation, while the latter one requires two resolvent evaluations. Note, however, that Algorithm 1 requires a projection evaluation onto Ω, which is not required by the method in [17]. Hence, an iteration of Algorithm 1 should be cheaper than that of the method in [17] in those instances of (3.11) for which Ω is a simple set, e.g., $\Omega = \mathcal{Z}$, or Ω is a closed ball in $\mathcal{Z}$. Second, both methods update A_k by means of (3.12), which is better than the update formula used by other accelerated methods (see, for example, [1, 4, 8, 27]), namely, (3.12) without the constant 2. As a result, both methods update A_k more aggressively than ones in the latter list of papers. Third, while the method in [17] assumes that the strong convexity of ϕ is entirely contained in h, Algorithm 1 does

not make such an assumption. We note, however, that the extra assumption that the strong convexity of ϕ is all in h is not at all restrictive since it can be easily enforced by moving any strong convexity of f to the function h (e.g., by subtracting from and/or adding to these functions a suitable positive multiple of the quadratic function $\|\cdot\|^2$).

The following result describes the convergence rate of Algorithm 1. Its proof closely follows the one given in [17] and is included in the online version of this work.[1]

THEOREM 3.4. *Consider the sequences $\{u_k\}$, $\{z_k\}$, and $\{A_k\}$ generated by Algorithm* 1. *Then, for every $k \geq 1$,*

$$A_k\phi(z_k) + \frac{A_k\mu + 1}{2}\|u - u_k\|^2 \leq A_k\phi(u) + \frac{1}{2}\|u - w_0\|^2 \quad \forall u \in \mathcal{Z} \tag{3.20}$$

and

$$A_k \geq \frac{1}{L}\max\left\{\frac{k^2}{2},\ \ 2\left(1 + \sqrt{\frac{\mu}{2L}}\right)^{2(k-1)}\right\}. \tag{3.21}$$

In regards to the third observation in the paragraph preceding Theorem 3.4, we conclude from (3.21) that the convergence rate of Algorithm 1 improves by moving the strong convexity (if any) from f to h since the latter preprocessing reduces the value of L and hence forces the sequence $\{A_k\}$ to be updated more aggressively.

3.3. Iteration-complexity bounds for solving ($P0$), ($P1$), and ($P2$). In this subsection, we derive the iteration-complexity of Algorithm 1 for solving problem ($P2$). As a consequence, its iteration-complexity for solving ($P1$) is established with the aid of Proposition 3.3. Moreover, by applying Algorithm 1 to a specific instance of (3.11), we also derive with the aid of Proposition 3.1 the iteration-complexity of solving ($P0$).

We start by deriving the iteration-complexity of Algorithm 1 to obtain solutions for problems ($P1$) and ($P2$).

PROPOSITION 3.5. *Let $w_0 \in \mathcal{Z}$, $\tau_1 > 0$, and $\tau_2 > 0$ be given and consider the sequences $\{z_k\}$, $\{r_k\}$, $\{v_k\}$, and $\{\varepsilon_k\}$ generated by Algorithm* 1. *Then, the following statements hold:*

(a) *for every $k \geq 1$, the triples $(z, r, \varepsilon) = (z_k, r_k, \varepsilon_k)$ and $(z, v, \varepsilon) = (z_k, v_k, \varepsilon_k)$ satisfy the inclusion in* (3.4) *and the inclusion in* (3.2)*, respectively;*

(b) *there exists an index*

$$k_0 = \mathcal{O}\left(\left\lceil \min\left\{\sqrt{L\left\lceil\tau_2^{-1}\right\rceil},\ 1 + \left(1 + \sqrt{\frac{L}{\mu}}\right)\log^+\left(L\left\lceil\tau_2^{-1}\right\rceil\right)\right\}\right\rceil\right) \tag{3.22}$$

such that, for every $k \geq k_0$, the triple $(z, r, \varepsilon) = (z_k, r_k, \varepsilon_k)$ also satisfies the inequality in (3.4) *and hence is a solution of problem* ($P2$)*;*

(c) *there exists an index*

$$\hat{k}_0 = \mathcal{O}\left(\left\lceil \min\left\{\sqrt{L(L+1)\left\lceil\tau_1^{-1}\right\rceil},\ 1 + \left(1 + \sqrt{\frac{L}{\mu}}\right)\log^+\left(L(L+1)\left\lceil\tau_1^{-1}\right\rceil\right)\right\}\right\rceil\right) \tag{3.23}$$

[1] http://www.optimization-online.org/DB_HTML/2013/10/4101.html.

such that, for every $k \geq \hat{k}_0$, the triple $(z, v, \varepsilon) = (z_k, v_k, \varepsilon_k)$ also satisfies the inequality in (3.2) *and hence is a solution of problem* $(P1)$.

Proof. (a) It follows from (3.20) that for any $u \in \mathcal{Z}$ and $k \geq 1$,

$$\begin{aligned}\phi(u) - \phi(z_k) &\geq \frac{1}{2A_k}\left(\|u - u_k\|^2 - \|u - w_0\|^2\right)\\ &= \frac{1}{A_k}\langle w_0 - u_k, u - z_k\rangle - \frac{1}{2A_k}(\|z_k - w_0\|^2 - \|z_k - u_k\|^2).\end{aligned}$$

The above inequality together with (3.19) and the definition of ε-subdifferential in (1.9) then imply that $r_k \in \partial_{\varepsilon_k}\phi(z_k) = \partial_{\varepsilon_k}(f + h)(z_k)$ for every $k \geq 1$. Now, the last conclusion, the definition of v_k in (3.19), and Lemma 3.2 with $c = L$, imply that $(z, v, \varepsilon) = (z_k, v_k, \varepsilon_k)$ satisfies the inclusion in (3.2) for every $k \geq 1$.

(b) We now claim that for every $k \geq 1$ such that

$$A_k \geq \max\{2, 2\tau_2^{-1}\}, \tag{3.24}$$

the triple $(z, r, \varepsilon) = (z_k, r_k, \varepsilon_k)$ also satisfies the inequality in (3.4). Indeed, by (3.18) and the inequality $\|a + b\|^2 \leq 2(\|a\|^2 + \|b\|^2)$, we have

$$\|r_k\|^2 = \frac{1}{A_k^2}\|u_k - w_0\|^2 \leq \frac{2}{A_k^2}\|z_k - w_0\|^2 + \frac{2}{A_k^2}\|z_k - u_k\|^2$$

and hence that

$$\|r_k\|^2 + 2\varepsilon_k \leq \left(\frac{2}{A_k^2} + \frac{1}{A_k}\right)\|z_k - w_0\|^2 + \left(\frac{2}{A_k^2} - \frac{1}{A_k}\right)\|z_k - u_k\|^2. \tag{3.25}$$

Since condition (3.24) is easily seen to imply that

$$\frac{2}{A_k^2} + \frac{1}{A_k} \leq \tau_2, \quad \frac{2}{A_k^2} - \frac{1}{A_k} \leq 0,$$

we conclude from (3.25) that the triple $(z, r, \varepsilon) = (z_k, r_k, \varepsilon_k)$ satisfies the inequality in (3.4) for every $k \geq 1$ satisfying (3.24). We have thus shown that the above claim holds. Now, define

$$k_0 := \left\lceil \min\left\{ 2\sqrt{L\left\lceil \tau_2^{-1}\right\rceil},\, 1 + \frac{1 + \sqrt{\mu/(2L)}}{2\sqrt{\mu/(2L)}}\log^+\left(L\left\lceil \tau_2^{-1}\right\rceil\right)\right\}\right\rceil \tag{3.26}$$

and note that k_0 clearly satisfies (3.22) and $k_0 \geq 1$. To end the proof, it suffices to show in view of the above claim that $k \geq k_0$ implies (3.24). Indeed, in view of the inequality $t/(1 + t) \leq \log(1 + t)$ for $t > -1$, we have

$$\frac{1 + \sqrt{\mu/(2L)}}{\sqrt{\mu/(2L)}} \geq \frac{1}{\log(1 + \sqrt{\mu/(2L)})}.$$

Thus, $k \geq k_0$ implies that either

$$k \geq 2\sqrt{L\left\lceil \tau_2^{-1}\right\rceil} \quad \text{or} \quad k \geq 1 + \frac{\log\left(L\left\lceil \tau_2^{-1}\right\rceil\right)}{2\log(1 + \sqrt{\mu/(2L)})}$$

and hence that

$$A_k \geq \max\left\{\frac{k^2}{2L},\, \frac{2}{L}\left(1 + \sqrt{\frac{\mu}{2L}}\right)^{2(k-1)}\right\} \geq 2\left\lceil \tau_2^{-1}\right\rceil \geq \max\{2, 2\tau_2^{-1}\},$$

where the first inequality is due to (3.21).

(c) Letting $\tau_2 := \tau_1/[2(L+1)]$ and using the fact that $\lceil \tau_2^{-1} \rceil \leq 2(L+1)\lceil \tau_1^{-1} \rceil$, statement (c) follows immediately from statement (b), the definition of v_k in (3.19), and Proposition 3.3. ∎

We now make some remarks about two possible termination criteria for Algorithm 1 which guarantee the computation of a triple $(z_k, r_k, \varepsilon_k)$ (resp., $(z_k, v_k, \varepsilon_k)$) which is a solution of problem $(P2)$ (resp., $(P1)$). The first, and less efficient, one is to simply perform k_0 (resp., $\hat{k}_0$) iterations where k_0 (resp., $\hat{k}_0$) is given by (3.26) (resp., (3.26) with τ_2 set to $\tau_2 = \tau_1/[2(L+1)]$). The second, and most efficient, one is to terminate Algorithm 1 at the first iteration k such that $(z, r, \varepsilon) = (z_k, r_k, \varepsilon_k)$ (resp., $(z, v, \varepsilon) = (z_k, v_k, \varepsilon_k)$) satisfies the inequality in (3.4) (resp., (3.2)). Note that for a general triple (z, r, ε) (resp., (z, v, ε)), it is not easy to check whether the inclusion in (3.4) (resp., (3.2)) is satisfied. However, when this triple is one of the triples $(z_k, r_k, \varepsilon_k)$ (resp., $(z_k, r_k, \varepsilon_k)$) generated by Algorithm 1, then Proposition 3.5(a) guarantees that (z, r, ε) (resp., (z, v, ε)) automatically satisfies the inclusion in (3.4) (resp., (3.2)).

The following result, which follows as an immediate consequence of Propositions 3.1 and 3.5, establishes the iteration-complexity of a specialization of Algorithm 1 for solving problem $(P0)$.

COROLLARY 3.6. *Let $\tilde{f}$ and $\tilde{h}$ satisfying conditions* B.1 *and* B.2 *and $(w_0, \lambda, \sigma_z) \in \mathcal{Z} \times \mathbb{R}_{++} \times (0, 1]$ determine an instance of $(P0)$ and define the functions f and h as in* (3.3). *Assume also that, for some constant $\tilde{L} \geq 0$, $\nabla \tilde{f}$ is $\tilde{L}$-co-coercive on Ω, where Ω is as in* B.2. *Consider the sequence $\{(z_k, v_k, \varepsilon_k)\}$ generated by Algorithm* 1 *with functions f and h as above and initial point w_0, and define*

$$(\tilde{z}_k, \tilde{s}_k, \tilde{\varepsilon}_k) := \left(z_k, \frac{v_k - (z_k - w_0)}{\lambda} - \nabla \tilde{f}(z_k), \frac{\varepsilon_k}{\lambda} \right) \quad \forall k \geq 1. \tag{3.27}$$

Then, the following statements hold:

(a) *for every $k \geq 1$, the triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon}) = (\tilde{z}_k, \tilde{s}_k, \tilde{\varepsilon}_k)$ satisfies the inclusion in* (1.6)*;*

(b) *there exists*

$$k_1 = \mathcal{O}\left(1 + \sqrt{\lambda \tilde{L} + 1} \log\left((\lambda \tilde{L} + 1) \sigma_z^{-1} \right) \right) \tag{3.28}$$

such that, for every $k \geq k_1$, the triple $(\tilde{z}, \tilde{s}, \tilde{\varepsilon}) = (\tilde{z}_k, \tilde{s}_k, \tilde{\varepsilon}_k)$ satisfies the inequality in (1.6) *and hence is a solution of $(P0)$.*

Proof. (a) This follows straightforwardly from Proposition 3.5(a), relation (3.27), and the observations made in the proof of Proposition 3.1.

(b) It is easy to verify that the assumptions of the corollary imply that f and h defined in (3.3) satisfy conditions C.1–C.4 with $\mu = 1$ and $L = \lambda \tilde{L} + 1$. Hence, from Proposition 3.5(c), the second bound in (3.23), and the fact that $\sigma_z^{-1} \geq 1$, we conclude that there exists an index k_1 satisfying (3.23) such that the triple $(z_k, v_k, \varepsilon_k)$ is a solution of $(P1)$ with $\tau_1 = \sigma_z^2$ for any $k \geq k_1$. The conclusion of the corollary now follows immediately from Proposition 3.1. ∎

We end this section by making two remarks about Corollary 3.6. First, when $\sigma_z \in (0, 1]$ is such that $\sigma_z^{-1} = \mathcal{O}(1)$, the iteration-complexity of solving $(P0)$ by means of the special case of Algorithm 1 described in Corollary 3.6 reduces to $\mathcal{O}(1 + (\lambda \tilde{L} + 1)^{1/2} \log(\lambda \tilde{L} + 1))$. Second, an observation similar to the one made after Proposition 3.5 can be made with respect to termination criteria for the special case of Algorithm 1 of Corollary 3.6 for solving $(P0)$. Third, our computational experiments use the

second termination criterion, i.e., the one which verifies whether the triple $(\tilde{z}_k, \tilde{s}_k, \tilde{\varepsilon}_k)$ defined in (3.27) satisfies the inequality in (1.6).

4. Accelerated BD algorithm for the CNE problem. This section considers a special accelerated instance of the CNE-BD-HPE framework for solving the CNE problem $CNE(\Psi_1, \Psi_2; g_1, g_2)$ in which the triples $(\tilde{x}_k, \tilde{a}_k, \varepsilon_k^x)$ and $(\tilde{y}_k, \tilde{b}_k, \varepsilon_k^y)$ in steps 1 and 2 are obtained with the aid of the scheme described in Corollary 3.6 applied to specific instances of problem $(P0)$. It also establishes the complexity of the resulting accelerated instance in terms of gradient, projection, and resolvent evaluations and shows that it is substantially better than that of the Tseng-BD algorithm when $\max\{L_{xx}, L_{yy}\} >> L_{xy}$.

We now describe the aforementioned accelerated instance of the CNE-BD-HPE framework for solving the CNE problem $CNE(\Psi_1, \Psi_2; g_1, g_2)$.

(Acc-BD) An accelerated BD-HPE algorithm for CNE$(\boldsymbol{\Psi_1}, \boldsymbol{\Psi_2}; \mathbf{g_1}, \mathbf{g_2})$.

0. Let $(x_0, y_0) \in \mathcal{X} \times \mathcal{Y}$, $\sigma \in (0, 1]$, and $\sigma_x, \sigma_y \in (0, \sigma)$ be given. Set $k = 1$ and
$$\lambda = \frac{\sqrt{(\sigma^2 - \sigma_x^2)(\sigma^2 - \sigma_y^2)}}{\sigma L_{xy}}; \tag{4.1}$$
1. invoke Algorithm 1 with $w_0 = x_{k-1}$, $\mathcal{Z} = \mathcal{X}$,
$$\Omega = \Omega_x, \quad h(\cdot) = \lambda g_1(\cdot). \quad f(\cdot) = \lambda \Psi_1(\cdot, y'_{k-1}) + \frac{1}{2}\|\cdot - x_{k-1}\|^2,$$
where $y'_{k-1} = P_{\Omega_y}(y_{k-1})$ to obtain a triple $(z, v, \varepsilon) \in \mathcal{X} \times \mathcal{X} \times \Re_+$ as in (3.14) and (3.19) such that
$$(\tilde{x}, \tilde{a}, \varepsilon^x) := \left(z, \frac{v - (z - x_{k-1})}{\lambda} - \nabla_x \Psi_1(z, y'_{k-1}), \frac{\varepsilon}{\lambda}\right) \tag{4.2}$$
satisfies (2.8), and set $(\tilde{x}_k, \tilde{a}_k, \varepsilon_k^x) = (\tilde{x}, \tilde{a}, \varepsilon^x)$;
2. invoke Algorithm 1 with $w_0 = y_{k-1}$, $\mathcal{Z} = \mathcal{Y}$,
$$\Omega = \Omega_y, \quad h(\cdot) = \lambda g_2(\cdot), \quad f(\cdot) = \lambda \Psi_2(\tilde{x}_k, \cdot) + \frac{1}{2}\|\cdot - y_{k-1}\|^2$$
to obtain a triple $(z, v, \varepsilon) \in \mathcal{Y} \times \mathcal{Y} \times \Re_+$ as in (3.14) and (3.19) such that
$$(\tilde{y}, \tilde{b}, \varepsilon^y) := \left(z, \frac{v - (z - y_{k-1})}{\lambda} - \nabla_y \Psi_2(\tilde{x}_k, z), \frac{\varepsilon}{\lambda}\right) \tag{4.3}$$
satisfies (2.9), and set $(\tilde{y}_k, \tilde{b}_k, \varepsilon_k^y) = (\tilde{y}, \tilde{b}, \varepsilon^y)$;
3. set $(\tilde{v}_k^x, \tilde{v}_k^y) = (\nabla_x \Psi_1(\tilde{x}_k, \tilde{y}_k) + \tilde{a}_k, \nabla_y \Psi_2(\tilde{x}_k, \tilde{y}_k) + \tilde{b}_k)$,
$$(x_k, y_k) = (x_{k-1}, y_{k-1}) - \lambda(\tilde{v}_k^x, \tilde{v}_k^y), \tag{4.4}$$
and $k \leftarrow k + 1$, and go to step 1.

end

The following result establishes convergence rate bounds for the Acc-BD algorithm.

THEOREM 4.1. *Algorithm Acc-BD is a special instance of the CNE-BD-HPE framework for solving $CNE(\Psi_1, \Psi_2; g_1, g_2)$. Moreover, assume that conditions* A.1, A.2, A.4, *and* A.5 *hold and consider the sequences $\{(x_k, y_k)\}$, $\{(\tilde{x}_k, \tilde{y}_k)\}$, $\{(\varepsilon_k^x, \varepsilon_k^y)\}$, and $\{(\tilde{v}_k^x, \tilde{v}_k^y)\}$ generated by the Acc-BD algorithm and define $\{(\tilde{x}_k^a, \tilde{y}_k^a)\}$, $\{\tilde{v}_k^a\}$, $\{\tilde{\varepsilon}_k^a\}$, d_0, and $\bar{\eta}$ as in Theorem* 2.3. *Then, the following statements hold for every $k \in \mathbb{N}$:*

(a) *if* A.3 *holds, then the pair* $((\tilde{v}_k^x, \tilde{v}_k^y), \varepsilon_k^x + \varepsilon_k^y)$ *is an NE-residual for* $(\tilde{x}_k, \tilde{y}_k)$ *and there exists* $i \leq k$ *such that*

$$\|(\tilde{v}_i^x, \tilde{v}_i^y)\| \leq \frac{L_{xy}\sigma d_0}{\sqrt{(\sigma^2 - \sigma_x^2)(\sigma^2 - \sigma_y^2)}} \sqrt{\frac{1+\sigma}{k(1-\sigma)}},$$

$$\varepsilon_i^x + \varepsilon_i^y \leq \frac{L_{xy}\sigma^3 d_0^2}{2k(1-\sigma^2)\sqrt{(\sigma^2 - \sigma_x^2)(\sigma^2 - \sigma_y^2)}};$$

(b) *if* A.3′ *holds, then the pair* $(\tilde{v}_k^a, \tilde{\varepsilon}_k^a)$ *is an NE-residual for* $(\tilde{x}_k^a, \tilde{y}_k^a)$ *and*

$$\|\tilde{v}_k^a\| \leq \frac{2L_{xy}\sigma d_0}{k\sqrt{(\sigma^2 - \sigma_x^2)(\sigma^2 - \sigma_y^2)}}, \qquad \tilde{\varepsilon}_k^a \leq \frac{2L_{xy}\sigma d_0^2}{k\sqrt{(\sigma^2 - \sigma_x^2)(\sigma^2 - \sigma_y^2)}}(1+\bar{\eta}).$$

Proof. The Acc-BD algorithm is clearly a special case of the CNE-BD-HPE framework in which (2.7) holds as an equality and the triples of steps 1 and 2 are found by means of Algorithm 1.

(a) This statement follows immediately from Theorem 2.3(a) with λ as in (4.1).

(b) This statement follows immediately from Theorem 2.3(b) with λ as in (4.1). ∎

The following corollary, which for the sake of brevity gives only ergodic complexity bounds for the Acc-BD algorithm, follows immediately from (4.1), Corollary 3.6, Theorem 4.1(b), and the fact that each iteration of Algorithm 1 performs at most two gradient evaluations, two resolvent evaluations of ∂h, and one projection onto Ω. The bounds derived on it are obtained under the assumption that the parameters σ, σ_x, and σ_y are chosen so that the inverses of σ_x, σ_y, $\sigma^2 - \sigma_x^2$, and $\sigma^2 - \sigma_y^2$ are all $\mathcal{O}(1)$.

COROLLARY 4.2. *At each iteration of the Acc-BD algorithm, the number of evaluations of* $\nabla_x \Psi_1(\cdot,\cdot)$ *and the number of resolvent evaluations of* ∂g_1 *and* $\partial \mathcal{I}_{\Omega_x}$ *are both bounded by*

$$\mathcal{O}\left(1 + \sqrt{L_{xx}/L_{xy} + 1}\,\log(L_{xx}/L_{xy} + 1)\right),$$

and the number of evaluations of $\nabla_y \Psi_2(\cdot,\cdot)$ *and the number of resolvent evaluations of* ∂g_2 *and* $\partial \mathcal{I}_{\Omega_y}$ *are bounded by*

$$\mathcal{O}\left(1 + \sqrt{L_{yy}/L_{xy} + 1}\,\log(L_{yy}/L_{xy} + 1)\right).$$

As a consequence, for every pair of positive scalars (ρ, ε)*, the Acc-BD algorithm finds a* (ρ, ε)*-Nash equilibrium of* $CNE(\Psi_1, \Psi_2; g_1, g_2)$ *by performing no more than*

$$\mathcal{O}\left(\left[1 + \sqrt{L_{xx}/L_{xy} + 1}\,\log(L_{xx}/L_{xy} + 1)\right] \max\left\{\frac{L_{xy}d_0^2}{\varepsilon}, \frac{L_{xy}d_0}{\rho}\right\}\right) \tag{4.5}$$

evaluations of $\nabla_x \Psi_1(\cdot,\cdot)$ *and resolvent evaluations of* ∂g_1 *and* $\partial \mathcal{I}_{\Omega_x}$ *and no more than*

$$\mathcal{O}\left(\left[1 + \sqrt{L_{yy}/L_{xy} + 1}\,\log(L_{yy}/L_{xy} + 1)\right] \max\left\{\frac{L_{xy}d_0^2}{\varepsilon}, \frac{L_{xy}d_0}{\rho}\right\}\right) \tag{4.6}$$

evaluations of $\nabla_y \Psi_2(\cdot,\cdot)$ *and resolvent evaluations of* ∂g_2 *and* $\partial \mathcal{I}_{\Omega_y}$.

It is worthwhile to compare the iteration-complexity bound (2.19) obtained for the Tseng-BD algorithm with the bounds (4.5) and (4.6) obtained for the Acc-BD algorithm in Corollary 4.2. Indeed, when $\max\{L_{xx}, L_{yy}\} \approx L_{xy}$, the two bounds in (4.5) and (4.6) are of the same order of magnitude as the one in (2.19). Consider now the relevant case in which $\max\{L_{xx}, L_{yy}\} >> L_{xy}$ and for the sake of concreteness assume that $L_{xx} = \max\{L_{xx}, L_{yy}\}$. Then, bound (2.19) is larger than (4.5) by a factor of

$$\Theta(\sqrt{\xi_x}/\log(\xi_x)),$$

where $\xi_x := L_{xx}/L_{xy} >> 1$. Also, the bound (2.19) is significantly larger than (4.6), i.e., by a factor

$$\tau = \begin{cases} \Theta(\xi_x) & \text{when } L_{yy} = \mathcal{O}(L_{xy}), \\ \Theta\left(\xi_x/[\log(\xi_y)\sqrt{\xi_y}]\right) & \text{when } L_{yy} >> L_{xy}, \end{cases}$$

where $\xi_y := L_{yy}/L_{xy}$.

In subsection 5.1, we describe a relevant class of convex optimization problems corresponding to CSP problems with $L_{yy} = 0$ and the ratio chosen $\xi_x := L_{xx}/L_{xy}$ arbitrarily large. Moreover, the resolvent evaluations of ∂g_2 are much more expensive than the ones for ∂g_1. Note that this class of problems is particularly suitable for Acc-BD in view of the fact that the bound (4.6) on the number of expensive resolvent evaluations of ∂g_2 is significantly smaller than the bound (4.5) on the number of cheap resolvent evaluations of ∂g_1.

5. Numerical experiments. In this section, we conduct experiments to evaluate the performance of the Acc-BD algorithm on a collection of CSP and/or CNE problems.

The numerical performance of the new method is compared with several previous methods, including the Tseng-BD algorithm in section 2.2 (see also subsection 5.2 of [13]), Tseng's MFBS algorithm (Tseng-MFBS) [26], and Korpelevich's extragradient method (Korp) [7]. Since the latter three methods, as well as Acc-BD, are special cases of the HPE framework first proposed in [22], we have used in their implementation an adaptive stepsize strategy (see [10]) which takes the largest extragradient stepsize satisfying the HPE relative error criteria. It is worth noting that this stepsize can be obtained by solving an easy quadratic equation. All four methods can be further accelerated by using a dynamic scaling technique discussed in [9, 10] to properly balance the magnitude of the primal and dual residuals. However, we have not included this technique in our implementation of these four methods (except in the extremely unbalanced CSP problem considered in subsection 5.1) since its implementation is complex and time-consuming. The true values of the Lipschitz constants L_{xx}, L_{yy}, and L_{xy}, all computed with respect to the Euclidean norm, are used for these four methods. We also note our implementation of Acc-BD incorporates the safeguard that the subproblems (2.8) and (2.9) are solved (exactly) using the recipe of Proposition 2.4 whenever $L_{xx} = 0$ and/or $L_{yy} = 0$, respectively.

We have also compared the four methods above with two other well-known methods, namely, Nemirovski's prox-method (referred to as Nemi-prox) [14, 27] and Nesterov's smooth approximation scheme [15] (referred to as Nest-app), where the smooth approximation is solved by a variant of Nesterov's optimal method due to Tseng, namely, Algorithm 3 of [27], based on the update formula (18) there. We observe that Nemi-prox is an extension of Korpelevich's extragradient method which is based on a

general distance-generating function (e.g., the entropy function $\sum_i x_i \log x_i$) instead of the standard one, namely, $\|\cdot\|^2/2$, used by Korpelevich's method. Our implementation of Nemi-prox uses the L_1-norm on $\mathcal{X} \times \mathcal{Y}$ and the entropy distance-generating function (see pp. 15–16 of [14]). Nest-app approximates the nonsmooth max component of the objective function by adding a small positive multiple of the entropy function to the max objective function and then applies the aforementioned Tseng's variant based on the entropy function to solve the resulting smooth approximation. The latter method endows both $\mathcal{X}$ and $\mathcal{Y}$ with the L_1-norm (see pp. 149–150 of [15]). To improve the performance of these two methods, their implementation follows the recipe given in [27], i.e., the initial value of the Lipschitz constant is set to a fraction (1/8 was used in [27] and also in our experiments) of its true value and is increased by a factor of 2 whenever a certain convergence criterion (see (23) and (45) of [27]) is not satisfied.

We now make some observations about the way our computational results are presented. First, for problems with bounded feasible sets $X \times Y$ such as the ones considered in subsections 5.1, 5.2, and 5.3, we have used the duality gap criterion of finding $(x, y) \in X \times Y$ such that $\text{gap}(x, y) \leq \varepsilon$ (see (2.4)) to terminate all methods due to the fact that Nest-app and Nemi-prox were originally designed for the latter termination criterion. Second, we have excluded Nest-app from the comparison in subsection 5.2 due to the fact that it has to solve the perturbed max subproblem exactly in order to compute the gradient of the smooth approximation of the original objective function and the fact that this subproblem is expensive for the CSP problem considered in this subsection. Third, we have also excluded Nest-app from the comparison in subsection 5.3 since it was not designed for the two-player Nash equilibrium problem. Fourth, we have excluded both Nest-app and Nemi-prox from the comparison in subsection 5.4 since the methods considered there are terminated based on the notion of approximate Nash equilibrium of Definition 2.1. The reason for changing the termination criterion in this subsection is due to the fact that its CSP problem has an unbounded feasible set and the fact that none of the six methods compared in this paper has been shown to converge based on the (stronger) duality gap criterion in this situation.

Finally, all the computational results were obtained in MATLAB R2013a on a quad-core Linux machine with 8GB memory.

5.1. A vector-matrix CSP problem. This subsection compares Acc-BD with Tseng-BD, Tseng-MFBS, Korp, Nemi-prox, and Nest-app for solving a collection of instances of the minimization problem

$$\min_{x \in \Delta_m} \frac{1}{2}\|Cx - b\|^2 + \theta_{max}(A(x)), \tag{5.1}$$

where $C \in \Re^{m \times m}$, $b \in \Re^m$, $A_1, \ldots, A_m \in \mathcal{S}^n$, and $A(x) = \sum_{i=1}^m x_i A_i \in \mathcal{S}^{n \times n}$. It is easy to verify that the above problem is equivalent to the following vector-matrix CSP problem:

$$\min_{x \in \Delta_m} \max_{y \in \Omega} \Psi_1(x, y) := \frac{1}{2}\|Cx - b\|^2 + \langle A(x), y \rangle, \tag{5.2}$$

where $\Omega = \{y \in \mathcal{S}^n : tr(y) = 1, y \succeq 0\}$.

Hence, we can apply the above methods on the CSP problem (5.2). In the numerical experiment, the matrices $A_1, \ldots, A_m$ and C are randomly generated such that each entry is generated independently and uniformly in the interval $[-1, 1]$ and

$A_1, \ldots, A_m$ are then symmetrized. All methods are terminated whenever the duality gap at a candidate solution $(\tilde{x}, \tilde{y})$ is less than a given tolerance ϵ, i.e.,

$$\frac{1}{2}\|C\tilde{x} - b\|^2 + \theta_{max}(A(\tilde{x})) - \min_{x \in \Delta_m} \left\{ \frac{1}{2}\|Cx - b\|^2 + \langle A(x), \tilde{y} \rangle \right\} \leq \epsilon. \tag{5.3}$$

Both the current pointwise iterate $(\tilde{x}_k, \tilde{y}_k)$ and the current ergodic iterate $(\tilde{x}_k^a, \tilde{y}_k^a)$ defined in (2.10) are used to check the stopping criterion (5.3) for the methods Acc-BD, Tseng-BD, Tseng-MFBS, Korp, and Nemi-prox. As described in Theorem 3 of [15] (see also Corollary 3 of [27]), the usual dual sequence generated by Nest-app is obtained by taking a weighted average of a sequence of dual maximizers for the perturbed max subproblems. In our experiment, we evaluate the max term of (5.3) at the current (usual) primal iterate and the min term of (5.3) at both the current weighted average dual iterate and the current dual maximizer and choose the largest of the two values in order to obtain the smallest value for (5.3).

Table 1 reports the CPU time and the number of eigen-decompositions (in order to evaluate the resolvent of $\partial \mathcal{I}_\Omega$) for each method. The CPU times reported in this table do not include the time spent to evaluate the left-hand side of stopping criterion (5.3), which is checked every five iterations. Due to space limitations, Table 1 does not specify the number of iterations performed by each method. We note, however, that the number of iterations performed by Tseng-BD, Tseng-MFBS, Korp, Nemi-prox, and Nest-app can be obtained by dividing the corresponding number of eigen-decompositions by $1, 1, 2, 2$, and 2, respectively. Also, the number of outer (HPE) iterations performed by Acc-BD is equal to the number of eigen-decompositions due to the fact that $L_{yy} = 0$ for the CSP considered in this subsection and the fact that the safeguard used in our implementation ensures that the proximal subproblem in the y-variable is solved by means of a single resolvent evaluation of $\partial \mathcal{I}_\Omega$.

Observe from the results reported in Table 1 that the four HPE methods performed better than both Nemi-prox and Nest-app on this collection of CSP problems. We believe that this might be due to the fact that the implementation of these methods incorporates both the adaptive stepsize and scaling strategies mentioned above (see [9] and [10] for details on these strategies). Also, Acc-BD was by far the fastest of the six methods on this collection of CSP instances.

5.2. Quadratic game problem. This subsection compares Acc-BD with Tseng-BD, Tseng-MFBS, Korp, and Nemi-prox for solving a collection of instances of the quadratic game problem

$$\min_{x \in \Delta_m} \max_{y \in \Delta_n} \Psi_1(x, y) := \frac{1}{2}\|Bx\|^2 + x^\top Ay - \frac{1}{2}\|Cy\|^2, \tag{5.4}$$

where $A \in \Re^{m \times n}$, $B \in \Re^{m \times m}$, and $C \in \Re^{n \times n}$.

For this comparison, the matrices A, B, and C are randomly generated such that each entry is nonzero with probability p and each nonzero entry is generated independently and uniformly in the interval $[0, 1]$. The above five methods are terminated whenever the duality gap at the candidate solution $(\tilde{x}, \tilde{y})$ is less than a given tolerance ϵ, i.e.,

$$\max_{y \in \Delta_n} \left\{ \frac{1}{2}\|B\tilde{x}\|^2 + \tilde{x}^\top Ay - \frac{1}{2}\|Cy\|^2 \right\} - \min_{x \in \Delta_m} \left\{ \frac{1}{2}\|Bx\|^2 + x^\top A\tilde{y} - \frac{1}{2}\|C\tilde{y}\|^2 \right\} \leq \epsilon. \tag{5.5}$$

TABLE 1

Computational results for the methods Acc-BD, Tseng-BD, Tseng-MFBS, Korp, Nemi-prox, and Nest-app on vector-matrix CSP problems (5.2) with different sizes. All methods are terminated using criterion (5.3) with $\epsilon = 10^{-4}$ and 10^{-5}. CPU time in seconds and the number of eigen-decompositions are reported for each method.

Tol.	Problem	Acc-BD		Tseng-BD		Tseng-MFBS		Korp		Nemi-prox		Nest-app
ϵ	$m/n/\frac{L_{xx}}{L_{xy}}$	Time	#eigen	Time	#eigen	Time	#eigen	Time	#eigen	Time	#eigen	Time /#eigen
10^{-4}	100/50/4.73	**0.34**	50	0.62	200	1.43	580	0.74	400	4.09	2730	68.04 / 33310
	100/100/2.66	**2.25**	185	3.66	360	4.42	435	5.83	910	42.74	6720	270.63 / 34650
	100/200/1.55	**9.12**	210	18.48	530	19.29	565	11.88	520	77.06	3220	924.61 / 31200
	200/50/9.30	**0.83**	150	1.66	350	2.93	640	2.45	860	5.05	2060	101.02 / 29030
	200/100/5.32	**2.12**	105	6.16	410	6.11	410	9.98	1080	25.88	3080	388.10 / 30520
	200/200/2.72	**10.85**	160	19.60	360	21.97	395	28.22	830	216.60	6620	1877.7 / 32150
	500/50/19.88	**1.38**	80	2.94	315	2.91	315	3.21	600	16.78	3600	147.75 / 19680
	500/100/12.16	**5.31**	130	22.49	790	12.95	455	17.06	1090	199.88	12870	675.36 / 20970
	500/200/6.98	**35.81**	265	141.43	1315	78.49	730	113.68	1890	489.88	8230	3346.9 / 21500
10^{-5}	100/50/4.73	**0.50**	150	0.78	230	2.08	865	1.01	540	11.77	7240	N/A / > 200000
	100/100/2.66	**4.51**	345	5.14	515	5.93	625	8.40	1340	89.70	14520	N/A / > 200000
	100/200/1.55	**19.38**	420	31.11	915	30.02	835	18.06	730	142.10	6570	N/A / > 200000
	200/50/9.30	**1.51**	305	2.15	500	5.67	1175	4.37	1530	9.75	3760	N/A / > 200000
	200/100/5.32	**4.28**	225	8.73	605	8.71	560	14.35	1550	56.70	6290	N/A / > 200000
	200/200/2.72	**15.87**	220	27.19	500	26.76	450	45.64	1310	489.90	15090	N/A/ > 200000
	500/50/19.88	**1.93**	130	4.16	440	5.39	585	4.47	840	28.52	5870	N/A / > 200000
	500/100/12.16	**6.06**	150	26.47	925	16.66	590	22.41	1420	331.80	21430	N/A / > 200000
	500/200/6.98	**66.12**	445	185.11	1650	118.17	1095	147.85	2500	1294.2	21260	N/A / > 200000

TABLE 2

Computational results for the methods Acc-BD, Tseng-BD, Tseng-MFBS, and Korp on two-player quadratic games with different sizes and sparsities. All methods are terminated using criterion (5.5) with $\epsilon = 10^{-3}$ and 10^{-6}. CPU time in seconds and number of gradient evaluations are reported for each method.

Tol.	Problem size	Lip. ratio		Acc-BD		Tseng-BD		Tseng-MFBS		Korp		Nemi-prox	
ε	$m/n/p$	$\frac{L_{xx}}{L_{xy}}$	$\frac{L_{yy}}{L_{xy}}$	Time	#grad./iter.	Time	#grad.	Time	#grad.	Time	#grad.	Time	#grad.
10^{-3}	1000/1000/0.1	48.11	48.03	**0.37**	276/7	0.86	700	0.90	720	0.94	720	1.68	1220
	1000/1000/0.2	91.11	91.46	**0.74**	378/8	2.92	1520	2.81	1500	3.05	1540	1.72	780
	1000/2000/0.1	34.37	135.67	**0.85**	347/7	4.85	2080	4.82	2080	4.98	2080	4.76	1880
	1000/2000/0.2	64.61	257.13	**2.22**	569/9	20.22	5120	20.04	5120	20.53	5120	5.93	1420
	2000/1000/0.1	135.28	34.18	**0.77**	307/7	4.93	2160	4.99	2160	5.20	2180	4.78	1920
	2000/1000/0.2	256.76	64.77	**2.00**	508/8	19.23	4900	18.95	4900	19.55	4900	5.80	1440
	2000/2000/0.1	95.65	96.04	**0.93**	286/6	4.49	1220	4.52	1240	4.57	1240	4.64	1220
	2000/2000/0.2	181.91	181.75	**2.25**	406/6	15.12	2480	15.29	2480	15.64	2500	4.98	780
10^{-6}	1000/1000/0.1	48.11	48.03	**0.91**	802/32	2.47	2120	2.52	2140	2.69	2140	14.24	10840
	1000/1000/0.2	91.11	91.46	**2.07**	1058/28	7.64	4000	7.72	4020	8.15	4020	12.49	6060
	1000/2000/0.1	34.37	135.67	**2.86**	1188/38	17.87	7780	17.94	7780	18.67	7780	36.25	14740
	1000/2000/0.2	64.61	257.13	**5.52**	1400/30	56.91	14540	57.02	14540	58.89	14540	42.69	10100
	2000/1000/0.1	135.28	34.18	**2.07**	844/24	17.07	7480	17.10	7480	17.90	3740	37.28	15120
	2000/1000/0.2	256.76	64.77	**5.10**	1256/26	49.00	15300	59.67	15300	60.81	15300	39.65	9700
	2000/2000/0.1	95.65	96.04	**2.80**	790/20	13.34	3700	13.39	3720	14.11	3720	42.72	11020
	2000/2000/0.2	181.91	181.75	**5.85**	1029/19	41.73	6760	41.54	6780	42.57	6800	37.56	5880

Both the iterate sequence $\{(\tilde{x}_k, \tilde{y}_k)\}$ and the ergodic sequence $\{(\tilde{x}^a_k, \tilde{y}^a_k)\}$ are used to check the stopping criterion (5.5) for the five methods considered in this section.

Table 2 reports the CPU time and the number of gradient evaluations (i.e., evaluations of $\nabla_x \Psi_1(\cdot,\cdot)$ and $\nabla_y \Psi_1(\cdot,\cdot)$, each counted separately) for each method. This table also reports the number of outer (HPE) iterations for the Acc-BD method. The CPU times reported in this table do not include the time spent to evaluate the left-hand side of stopping criterion (5.5), which is checked every outer iteration for Acc-BD and every five iterations for the other four methods. Due to space limitations, Table 2 does not specify the number of iterations performed by the methods Tseng-BD, Tseng-MFBS, Korp, and Nemi-prox. We note, however, that the number of iterations performed by these four methods can be obtained by dividing the corresponding number of gradient evaluations by 4.

Table 2 shows that Tseng-BD had almost the same numerical performance as Tseng-MFBS and Korp on this collection of quadratic game instances and they are outperformed by Nemi-prox on several instances of this collection. Acc-BD was by far the fastest among the four methods on all instances of this collection. The results also confirm our conclusion in the paragraph following Corollary 4.2 that the performance of Acc-BD improves as the ratio $\max\{L_{xx}, L_{yy}\}/L_{xy}$ increases.

5.3. CNE problem. This subsection compares Acc-BD with Tseng-BD, Tseng-MFBS, Korp, and Nemi-prox for solving a collection of instances of $CNE(\Psi_1, \Psi_2, g_1, g_2)$, where

$$X := \Delta_m, \quad Y := \Delta_n, \quad g_1 := \mathcal{I}_X(x), \quad g_2(y) := \mathcal{I}_Y(y),$$
$$\Psi_1(x,y) := \frac{1}{2}x^\top A_1 x + \langle x, B_1 y\rangle, \quad \Psi_2(x,y) := \frac{1}{2}y^\top A_2 y + \langle x, B_2 y\rangle, \tag{5.6}$$

and the matrices $A_1 \in \mathcal{S}^m$, $A_2 \in \mathcal{S}^n$, and $B_1, B_2 \in \Re^{m\times n}$ satisfy the condition that

$$C := \begin{pmatrix} A_1 & B_1 + B_2 \\ (B_1 + B_2)^\top & A_2 \end{pmatrix} \succeq 0. \tag{5.7}$$

TABLE 3

Computational results for the methods Acc-BD, Tseng-BD, Tseng-MFBS, and Korp on CNE problems (5.6) *with different sizes. All methods are terminated whenever the gap function* (2.4) *at the candidate solution is less than* $\epsilon = 10^{-3}$ *and* 10^{-6}*. CPU time in seconds and number of gradient evaluations are reported for each method.*

Tol.	Problem size	Lip. ratio		Acc-BD		Tseng-BD		Tseng-MFBS		Korp		Nemi-prox	
ε	m/n	$\frac{L_{xx}}{L_{xy}}$	$\frac{L_{yy}}{L_{xy}}$	Time	#grad./it.	Time	#grad.	Time	#grad.	Time	#grad.	Time	#grad.
10^{-3}	500/500	44.77	45.19	**0.13**	90/4	0.21	180	0.21	180	0.20	180	4.85	6960
	500/1000	53.63	54.39	**0.13**	94/4	0.21	120	0.21	120	0.25	140	22.46	12240
	1000/500	53.67	53.68	**0.23**	140/6	0.48	300	0.48	300	0.54	320	4.85	6960
	1000/1000	63.08	63.65	**0.29**	104/4	0.49	180	0.48	180	0.56	200	40.41	13400
	2000/2000	76.97	75.94	**0.63**	112/4	1.12	200	1.00	180	1.12	200	218.84	34120
	2000/1000	76.21	75.49	**0.75**	135/5	1.53	280	1.66	280	1.66	300	220.43	34120
	2000/2000	89.28	89.05	**1.45**	153/5	1.68	180	1.68	180	1.89	200	187.85	16680
10^{-6}	500/500	44.77	45.19	**0.23**	253/11	0.62	640	0.58	640	0.61	680	45.88	22740
	500/1000	53.63	54.39	**0.48**	290/12	1.79	1060	1.68	1040	1.91	1100	265.66	67160
	1000/500	53.67	53.68	**3.19**	1832/56	8.41	4960	8.49	4920	9.17	5080	540.10	137400
	1000/1000	63.08	63.65	**0.92**	270/10	2.20	740	2.36	720	2.34	780	259.71	41160
	2000/2000	76.97	75.94	**4.23**	655/19	15.85	2840	15.87	2840	15.98	2860	1790.0	155000
	2000/1000	76.21	75.49	**9.95**	1664/46	32.31	5900	32.81	5840	33.04	5920	1217.1	173520
	2000/2000	89.28	89.05	**3.59**	380/12	6.50	680	6.73	700	7.28	740	738.71	61760

In the numerical experiment, the matrices B_1 and B_2 have their entries generated independently according to the standard normal distribution and the matrices A_1 and A_2 are then set $A_1 = B_1 B_1^\top + I_m$ and $A_2 = B_2^\top B_2 + I_n$, which guarantees that condition (5.7) holds. It is easy to verify that the above randomly generated $CNE(\Psi_1, \Psi_2, g_1, g_2)$ problem satisfies all conditions required by the Tseng-BD and Acc-BD methods.

The five methods considered in this section are terminated whenever the gap function (2.4) at the candidate solution $(\tilde{x}, \tilde{y})$ is less than a given tolerance ϵ. Both the iterate sequence $\{(\tilde{x}_k, \tilde{y}_k)\}$ and the ergodic sequence $\{(\tilde{x}_k^a, \tilde{y}_k^a)\}$ are used as the candidate solutions for all five methods.

Table 3 reports the CPU time and the number of gradient evaluations (i.e., evaluations of $\nabla_x \Psi_1(\cdot,\cdot)$ and $\nabla_y \Psi_2(\cdot,\cdot)$, each counted separately) for each method and the number of outer (HPE) iterations for the Acc-BD method. The CPU times reported in this table do not include the time spent to evaluate the gap function (2.4), which is done every outer iteration for Acc-BD and every five iterations for the other four methods. Due to space limitations, Table 3 does not specify the number of iterations performed by the methods Tseng-BD, Tseng-MFBS, Korp, and Nemi-prox. We note, however, that the number of iterations performed by these four methods can be obtained by dividing the corresponding number of gradient evaluations by 4.

The computational results show that the methods Tseng-BD, Tseng-MFBS, and Korp had comparable numerical performance on this collection of NE instances. Acc-BD was by far the fastest among the five methods on all instances of this collection.

5.4. A regularized least-square problem. This subsection compares Acc-BD with Tseng-BD, Tseng-MFBS, and Korp for solving a collection of instances of the following regularized least-square problem:

$$\min_{x \in \Re^{k \times n}} \frac{1}{2}\|Ax - B\|_F^2 + \beta\|x\|_1 + \gamma\|x\|_*, \tag{5.8}$$

where the matrices $A \in \Re^{m \times k}$, $B \in \Re^{m \times n}$ and the regularization parameters $\beta > 0$ and $\gamma > 0$ are given. Note that the purpose of the regularization term $\beta\|x\|_1 + \gamma\|x\|_*$ in (5.8) is to simultaneously induce sparsity and low-rankness on x.

TABLE 4

Computational results for the methods Acc-BD and Tseng-BD on the regularized least-square problems (5.8) *with different problem sizes. The two methods are terminated using criterion* (5.11) *with* $\epsilon = 10^{-3}$. *CPU time in seconds and the number of SVD computations are reported for each method.*

Problem				Acc-BD		Tseng-BD		Tseng-MFBS		Korp	
m	k	n	$\frac{L_{xx}}{L_{xy}}$	Time	#svd	Time	#svd	Time	#svd	Time	#svd
100	100	100	2.40	**0.31**	35	0.35	72	0.37	81	1.23	276
100	200	200	4.14	**1.70**	36	1.88	93	2.00	96	5.96	356
100	500	500	4.92	**62.71**	99	191.43	465	192.02	479	265.85	752
200	100	100	3.79	**0.61**	50	0.87	165	1.05	174	3.62	636
200	200	200	5.44	**1.60**	39	2.65	106	2.67	108	8.64	398
200	500	500	7.14	**44.63**	100	175.11	682	386.59	1577	376.03	1350
500	100	100	5.70	**0.28**	26	0.61	91	0.67	93	2.48	350
500	200	200	6.61	**1.72**	38	3.23	108	3.32	110	10.85	414
500	500	500	9.49	**57.06**	126	169.56	681	170.18	679	577.10	2558

Clearly, problem (5.8) is a special instance of the class of structured convex optimization problem (see (16) in [9])

$$\min_{x\in\mathcal{X}} f(x) + h_1(x) + h_2(x), \tag{5.9}$$

where $\mathcal{X} = \Re^{k\times n}$, $f(x) = \frac{1}{2}\|Ax - B\|_F^2$, $h_1(x) = \beta\|x\|_1$, and $h_2(x) = \gamma\|x\|_*$, and the resolvents of ∂h_1 and ∂h_2 can be evaluated in closed form (see [2], for example). Problem (5.9) is shown in [9] to be equivalent to the inclusion problem

$$0 \in \nabla f(x) + \partial h_1(x) + y, \quad 0 \in \partial h_2^*(y) - x,$$

or, equivalently, to the CSP problem,

$$\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} f(x) + \langle x, y\rangle + h_1(x) - h_2^*(y). \tag{5.10}$$

Hence we can apply the above four methods to solve problem (5.8). In the numerical experiment, the matrices A and B are generated as sparse matrices with 1% nonzero entries that are independently and uniformly distributed in the interval $[-1, 1]$. The regularization parameters β and γ are set to $0.0005n$. In view of (2.19) and Theorem 4.1, both Tseng-BD and Acc-BD generate an easily computable NE-residual $((\tilde{v}_k^x, \tilde{v}_k^y), \varepsilon_k^x + \varepsilon_k^y)$ at each iteration. We have also implemented versions of Tseng-MFBS and Korp (see, for example, [11]) that generate the above easily computable NE-residuals. The above four methods are then terminated whenever

$$\max\left\{\frac{\|(\tilde{v}_k^x, \tilde{v}_k^y)\|}{\max\{1, \|\tilde{x}_k\|, \|\tilde{y}_k\|\}}, \varepsilon_k^x + \varepsilon_k^y\right\} < \epsilon. \tag{5.11}$$

Table 4 reports the CPU time and the number of SVD computations (in order to evaluate the resolvent of ∂h_2^*) for the above four methods. Due to space limitations, Table 4 does not specify the number of iterations performed by each method. We note, however, that the number of iterations performed by Tseng-BD, Tseng-MFBS, and Korp can be obtained by dividing the corresponding number of SVD computations by $1, 1$, and 2, respectively. Also, the number of outer (HPE) iterations performed by Acc-BD is equal to the number of SVD computations due to the fact that $L_{yy} = 0$ for the CSP considered in this subsection and the fact that the safeguard used in our

implementation ensures that the proximal subproblem in the y-variable is solved by means of a single resolvent evaluation of ∂h_2^*.

Table 4 shows that the methods Tseng-BD and Tseng-MFBS had comparable numerical performance on this collection of regularized least-square problem instances and Korp was the slowest among the four methods on eight of nine instances. The computational results also show that Acc-BD was the fastest in this collection, and it performed especially well when the computational cost of computing an SVD is much larger than that of a matrix-vector multiplication.

Appendix A. A remark about condition A.3′.

PROPOSITION A.1. *Assume condition* A.1 *holds; then* A.3′ *implies* A.3.

Proof. Let $(x,y),(\tilde{x},\tilde{y}) \in X \times Y$ be given. The assumption that A.3′ holds implies that $\Psi := \Psi_1 + \Psi_2$ is convex and hence that

$$\Psi(\tilde{x},\tilde{y}) \geq \Psi(x,y) + \langle \nabla_x \Psi(x,y), \tilde{x} - x\rangle + \langle \nabla_y \Psi(x,y), \tilde{y} - y\rangle.$$

Also, A.3′ implies that $\Psi_1(x,\cdot) + \Psi_2(\cdot,y)$ is concave and hence that

$$\Psi(x,y) + \langle \nabla_y \Psi_1(x,y), \tilde{y} - y\rangle + \langle \nabla_x \Psi_2(x,y), \tilde{x} - x\rangle \geq \Psi_1(x,\tilde{y}) + \Psi_2(\tilde{x},y).$$

Now, the fact that Ψ convex implies that $\Psi(\cdot,\tilde{y}) + \Psi(\tilde{x},\cdot)$ is convex, which, together with the fact that $\Psi_1(\tilde{x},\cdot) + \Psi_2(\cdot,\tilde{y})$ is concave, then implies that $\Psi_1(\cdot,\tilde{y}) + \Psi_2(\tilde{x},\cdot)$ is convex. Hence,

$$\Psi_1(x,\tilde{y}) + \Psi_2(\tilde{x},y) \geq \Psi(\tilde{x},\tilde{y}) + \langle \nabla_x \Psi_1(\tilde{x},\tilde{y}), x - \tilde{x}\rangle + \langle \nabla_y \Psi_2(\tilde{x},\tilde{y}), y - \tilde{y}\rangle.$$

Combining the above three inequalities, we then conclude that

$$\langle \nabla_x \Psi_1(\tilde{x},\tilde{y}) - \nabla_x \Psi_1(x,y), x - \tilde{x}\rangle + \langle \nabla_y \Psi_2(\tilde{x},\tilde{y}) - \nabla_y \Psi_2(x,y), y - \tilde{y}\rangle \geq 0.$$

We have thus shown that A.3 holds. ∎

Appendix B. Proof of Theorem 2.3. The technical result below will be used to show that the corresponding ergodic sequence $(\tilde{v}_k^a, \tilde{\varepsilon}_k^a)$ (see (2.10) and (2.11)) is an NE-residual for the ergodic iterate $(\tilde{x}_k^a, \tilde{y}_k^a)$ (see (2.10)). It is a generalization of Proposition 5.1 of [12].

LEMMA B.1. *Let $X \subseteq \mathcal{X}$ and $Y \subseteq \mathcal{Y}$ be given convex sets and let $\Gamma_i : X \times Y \to \Re, i = 1,2$, be functions such that the following assumptions are satisfied:*

(a) *for each pair $(x,y) \in X \times Y$, the function $\Gamma_1(x,\cdot) + \Gamma_2(\cdot,y) : X \times Y \to \Re$ is concave;*

(b) *$\Gamma_1(\cdot,\cdot) + \Gamma_2(\cdot,\cdot) : X \times Y \to \Re$ is convex.*

Suppose that, for $i = 1,\ldots,k$, $(x_i,y_i) \in X \times Y$ and $(v_{x,i}, v_{y,i}) \in \mathcal{X} \times \mathcal{Y}$ satisfy

$$(v_{x,i}, v_{y,i}) \in \partial_{\varepsilon_i}\Big(\Gamma_1(\cdot,y_i) + \Gamma_2(x_i,\cdot)\Big)(x_i,y_i). \tag{B.1}$$

Let $\alpha_1,\ldots,\alpha_k \geq 0$ be such that $\sum_{i=1}^k \alpha_i = 1$ and define

$$(x^a,y^a) = \sum_{i=1}^k \alpha_i (x_i,y_i), \quad (v_x^a, v_y^a) = \sum_{i=1}^k \alpha_i (v_{x,i}, v_{y,i}), \tag{B.2}$$

$$\varepsilon^a := \sum_{i=1}^{k} \alpha_i [\varepsilon_i + \langle x_i - x^a, v_{x,i}\rangle + \langle y_i - y^a, v_{y,i}\rangle]. \tag{B.3}$$

Then, $\varepsilon^a \geq 0$ *and*

$$(v_x^a, v_y^a) \in \partial_{\varepsilon^a}\Big(\Gamma_1(\cdot, y^a) + \Gamma_2(x^a, \cdot)\Big)(x^a, y^a). \tag{B.4}$$

Proof. By (B.1), we have

$$\Gamma_1(x, y_i) + \Gamma_2(x_i, y) \geq \Gamma_1(x_i, y_i) + \Gamma_2(x_i, y_i) + \langle v_{x,i}, x - x_i\rangle + \langle v_{y,i}, y - y_i\rangle - \varepsilon_i \quad \forall (x, y) \in X \times Y.$$

Using the assumption that $\Gamma_1(x, \cdot) + \Gamma_2(\cdot, y)$ is concave for every $(x, y) \in X \times Y$, $\Gamma_1(\cdot, \cdot)+\Gamma_2(\cdot, \cdot)$ is convex, the assumption that $\sum_{i=1}^k \alpha_i = 1$ and $\alpha_i \geq 0$ for $i = 1, \ldots, k$, and relations (B.2) and (B.3), we conclude that

$$\begin{aligned}
\Gamma_1(x, y^a) + \Gamma_2(x^a, y) &\geq \sum_{i=1}^{k} \alpha_i [\Gamma_1(x, y_i) + \Gamma_2(x_i, y)] \\
&\geq \sum_{i=1}^{k} \alpha_i \left(\Gamma_1(x_i, y_i) + \Gamma_2(x_i, y_i) + \langle v_{x,i}, x - x_i\rangle + \langle v_{y,i}, y - y_i\rangle - \varepsilon_i\right) \\
&\geq \Gamma_1(x^a, y^a) + \Gamma_2(x^a, y^a) + \sum_{i=1}^{k} \alpha_i \left(\langle v_{x,i}, x - x^a\rangle + \langle v_{y,i}, y - y^a\rangle\right) \\
&\quad - \sum_{i=1}^{k} \alpha_i \left(\langle v_{x,i}, x_i - x^a\rangle + \langle v_{y,i}, y_i - y^a\rangle + \varepsilon_i\right) \\
&= \Gamma_1(x^a, y^a) + \Gamma_2(x^a, y^a) + \langle v_x^a, x - x^a\rangle + \langle v_y^a, y - y^a\rangle - \varepsilon^a
\end{aligned}$$

for every $(x, y) \in X \times Y$. We have thus shown that (B.4) holds. The nonnegativity of ε^a follows from the above relation with $(x, y) = (x^a, y^a)$. ∎

With the aid of the Lemma B.1, we now give the proof of Theorem 2.3.

Proof of Theorem 2.3. In view of step 3 of the CNE-BD-HPE framework, we have

$$\begin{aligned}
\tilde{v}_k^x &\in \nabla_x \Psi_1(\tilde{x}_k, \tilde{y}_k) + \partial_{\varepsilon_k^x} g_1(\tilde{x}_k) \subseteq \partial_{\varepsilon_k^x}[\widehat{\Psi}_1(\cdot, \tilde{y}_k)](\tilde{x}_k), \\
\tilde{v}_k^y &\in \nabla_y \Psi_2(\tilde{x}_k, \tilde{y}_k) + \partial_{\varepsilon_k^y} g_2(\tilde{y}_k) \subseteq \partial_{\varepsilon_k^y}[\widehat{\Psi}_2(\tilde{x}_k, \cdot)](\tilde{y}_k),
\end{aligned}$$

from which we conclude that

$$\begin{aligned}
(\tilde{v}_k^x, \tilde{v}_k^y) &\in \Big[\partial_{\varepsilon_k^x}[\widehat{\Psi}_1(\cdot, \tilde{y}_k)](\tilde{x}_k)\Big] \times \Big[\partial_{\varepsilon_k^y}[\widehat{\Psi}_2(\tilde{x}_k, \cdot)](\tilde{y}_k)\Big] \\
&\subseteq \partial_{\varepsilon_k^x + \varepsilon_k^y}[\widehat{\Psi}_1(\cdot, \tilde{y}_k) + \widehat{\Psi}_2(\tilde{x}_k, \cdot)](\tilde{x}_k, \tilde{y}_k),
\end{aligned} \tag{B.5}$$

where the last inclusion follows from the definition of ε-subdifferential. Moreover, (2.12) follows directly from Theorem 3.2 of [13] and the fact that the CNE-BD-HPE framework is a special case of the BD-HPE framework in which $\lambda_k = \lambda$ for all k.

It follows from statement (a) and Lemma B.1 with $\Gamma_1 = \widehat{\Psi}_1|_{X\times Y}$ and $\Gamma_2 = \widehat{\Psi}_2|_{X\times Y}$, and $(x_i, y_i) = (\tilde{x}_i, \tilde{y}_i)$ and $(v_{x,i}, v_{y,i}) = (\tilde{v}_i^x, \tilde{v}_i^y)$ for $i = 1, \ldots, k$, that

$$\tilde{v}_k^a \in \partial_{\tilde{\varepsilon}_k^a}[\Psi_1(\cdot, \tilde{y}_k^a) + \Psi_2(\tilde{x}_k^a, \cdot)](\tilde{x}_k^a, \tilde{y}_k^a).$$

Moreover, (2.13) follows directly from Theorem 3.3 of [13] and the fact that the CNE-BD-HPE framework is a special case of the BD-HPE framework in which $\lambda_k = \lambda$ for all k. □

REFERENCES

[1] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci., 2 (2009), pp. 183–202.

[2] J.-F. Cai, E. J. Candès, and Z. Shen, *A singular value thresholding algorithm for matrix completion*, SIAM J. Optim., 20 (2010), pp. 1956–1982.

[3] A. Chambolle and T. Pock, *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vis., 40 (2011), pp. 120–145.

[4] C. D. Dang, K. Dai, and G. Lan, *A linearly convergent first-order algorithm for total variation minimisation in image processing*, Int. J. Bioinform. Res. Appl., 10 (2014), pp. 4–26.

[5] J. Eckstein and B. F. Svaiter, *A family of projective splitting methods for the sum of two maximal monotone operators*, Math. Program., 111 (2008), pp. 173–199.

[6] J. Eckstein and B. F. Svaiter, *General projective splitting methods for sums of maximal monotone operators*, SIAM J. Control Optim., 48 (2009), pp. 787–811.

[7] G. M. Korpelevič, *An extragradient method for finding saddle points and for other problems*, Èkonom. Mat. Metody, 12 (1976), pp. 747–756.

[8] G. Lan, Z. Lu, and R. D. C. Monteiro, *Primal-dual first-order methods with $\mathcal{O}(1/\epsilon)$ iteration-complexity for cone programming*, Math. Program., 126 (2011), pp. 1–29.

[9] R. D. Monteiro, C. Ortiz, and B. F. Svaiter, *A first-order block-decomposition method for solving two-easy-block structured semidefinite programs*, Math. Program. Comput., 6 (2014), pp. 103–150.

[10] R. D. Monteiro, C. Ortiz, and B. F. Svaiter, *Implementation of a block-decomposition algorithm for solving large-scale conic semidefinite programming problems*, Comput. Optim. Appl., 57 (2014), pp. 45–69.

[11] R. D. C. Monteiro and B. F. Svaiter, *On the complexity of the hybrid proximal extragradient method for the iterates and the ergodic mean*, SIAM J. Optim., 20 (2010), pp. 2755–2787.

[12] R. D. C. Monteiro and B. F. Svaiter, *Complexity of variants of Tseng's modified F-B splitting and Korpelevich's methods for hemivariational inequalities with applications to saddle-point and convex optimization problems*, SIAM J. Optim., 21 (2011), pp. 1688–1720.

[13] R. D. C. Monteiro and B. F. Svaiter, *Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers*, SIAM J. Optim., 23 (2013), pp. 475–507.

[14] A. Nemirovski, *Prox-method with rate of convergence $O(1/T)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems*, SIAM J. Optim., 15 (2004), pp. 229–251.

[15] Y. Nesterov, *Smooth minimization of non-smooth functions*, Math. Program., 103 (2005), pp. 127–152.

[16] Y. Nesterov, *Dual extrapolation and its applications to solving variational inequalities and related problems*, Math. Program., 109 (2007), pp. 319–344.

[17] Y. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161.

[18] A. Ouorou, *Epsilon-proximal decomposition method*, Math. Program., 99 (2004), pp. 89–108.

[19] R. T. Rockafellar, *On the maximal monotonicity of subdifferential mappings*, Pacific J. Math., 33 (1970), pp. 209–216.

[20] R. T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optim., 14 (1976), pp. 877–898.

[21] M. V. Solodov, *A class of decomposition methods for convex optimization and monotone variational inclusions via the hybrid inexact proximal point framework*, Optim. Methods Softw., 19 (2004), pp. 557–575.

[22] M. V. Solodov and B. F. Svaiter, *A hybrid approximate extragradient-proximal point algorithm using the enlargement of a maximal monotone operator*, Set-Valued Anal., 7 (1999), pp. 323–345.

[23] M. V. Solodov and B. F. Svaiter, *A hybrid projection-proximal point algorithm*, J. Convex Anal., 6 (1999), pp. 59–70.

[24] M. V. Solodov and B. F. Svaiter, *An inexact hybrid generalized proximal point algorithm and some new results on the theory of Bregman functions*, Math. Oper. Res., 25 (2000), pp. 214–230.

[25] M. V. Solodov and B. F. Svaiter, *A unified framework for some inexact proximal point algorithms*, Numer. Funct. Anal. Optim., 22 (2001), pp. 1013–1035.
[26] P. Tseng, *A modified forward-backward splitting method for maximal monotone mappings*, SIAM J. Control Optim., 38 (2000), pp. 431–446.
[27] P. Tseng, *On accelerated proximal gradient methods for convex-concave optimization*, J. Optim., submitted.