A PRIMAL-DUAL PARALLEL METHOD WITH $O(1/\epsilon)$ CONVERGENCE FOR CONSTRAINED COMPOSITE CONVEX PROGRAMS

HAO YU† AND MICHAEL J. NEELY†

Abstract. This paper considers large scale constrained convex (possibly composite and non-separable) programs, which are usually difficult to solve by interior point methods or other Newton-type methods due to the non-smoothness or the prohibitive computation and storage complexity for Hessians and matrix inversions. Instead, they are often solved by first order gradient based methods or decomposition based methods. The conventional primal-dual subgradient method, also known as the Arrow-Hurwicz-Uzawa subgradient method, is a low complexity algorithm with an $O(1/\epsilon^2)$ convergence time. Recently, a new Lagrangian dual type algorithm with a faster $O(1/\epsilon)$ convergence time is proposed in Yu and Neely (2017). However, if the objective or constraint functions are not separable, each iteration of the Lagrangian dual type method in Yu and Neely (2017) requires to solve an unconstrained convex program, which can have huge complexity. This paper proposes a new primal-dual type algorithm with a faster $O(1/\epsilon)$ convergence for general constrained convex programs. Each iteration of the new algorithm can be implemented in parallel with low complexity even when the original problem is composite and non-separable.

Key words. constrained convex programs, composite convex programs, parallel methods, convergence time

AMS subject classifications. 90C25, 90C30

1. Introduction. Recall that a function $h(x)$ is said to be separable (with respect to its vector variable $x$) if it can be written as the summation of multiple smaller functions, each of which only involves disjoint components or blocks of $x$, e.g., $h(x) = \sum_{i=1}^{n} h^{(i)}(x_i)$. Fix positive integers $n$ and $m$, which are typically large. Consider the following constrained convex program:

$$\begin{align*}
(1) & \quad \min \ F(x) \triangleq f(x) + \bar{f}(x) \\
(2) & \quad \text{s.t.} \quad G_k(x) \triangleq g_k(x) + \tilde{g}_k(x) \leq 0, \forall k \in \{1, 2, \ldots, m\} \\
(3) & \quad x \in \mathcal{X}
\end{align*}$$

where set $\mathcal{X} \subseteq \mathbb{R}^n$ is a closed convex set; function $f(x)$ is convex and smooth (but possibly non-separable) on $\mathcal{X}$; function $\bar{f}(x)$ is convex and separable (but possibly non-smooth) on $\mathcal{X}$; functions $g_k(x), \forall k \in \{1, 2, \ldots, m\}$ are convex, Lipschitz continuous and smooth (but possibly non-separable) on $\mathcal{X}$, and functions $\tilde{g}_k(x)$ are convex, Lipschitz continuous and separable (but possibly non-smooth) on $\mathcal{X}$. The convex program (1)-(3) is called a constrained composite convex program since either its objective function $F(x)$ or each of its constraint functions $G_k(x)$ is in general the sum of a smooth function and a non-smooth function.

Denote the stacked vector of functions via $g(x) = [g_1(x), g_2(x), \ldots, g_m(x)]^T$; $\tilde{g}(x) = [\tilde{g}_1(x), \tilde{g}_2(x), \ldots, \tilde{g}_m(x)]^T$ and $G(x) = [G_1(x), G_2(x), \ldots, G_m(x)]^T$. The Lipschitz continuity of each $g_k(x)$ and $\tilde{g}_k(x)$ implies that $g(x) + \tilde{g}(x)$ is Lipschitz continuous on $\mathcal{X}$. Throughout this paper, we use $\|x\|$ to denote the Euclidean norm.

†Department of Electrical Engineering, University of Southern California, Los Angeles, CA (yuhao@usc.edu, mjneely@usc.edu).
of vector $x$, also known as the $l_2$ norm, and have the following assumptions on convex program (1)-(3):

**Assumption 1 (Basic Assumptions).**
- There exists a (possibly non-unique) optimal solution $x^* \in X$ that solves convex program (1)-(3).
- There exists $\beta > 0$ such that $\|G(x) - G(y)\| \leq \beta \|x - y\|$ for all $x, y \in X$, i.e., $G(x)$ is Lipschitz continuous with modulus $\beta$.

**Assumption 2 (Existence of Lagrange multipliers).** There exists a Lagrange multiplier vector $\lambda^* = [\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*] \geq 0$ attaining the strong duality for problem (1)-(3), i.e.,

$$q(\lambda) = \min_{x \in X} \{F(x) + \sum_{k=1}^m \lambda_k G_k(x)\}$$

where $q(\lambda) = \min_{x \in X} \{F(x) : G_k(x) \leq 0, \forall k \in \{1, 2, \ldots, m\}\}$ is the Lagrangian dual function of problem (1)-(3).

Under Assumption 1 and Assumption 2, this paper proposes a new primal dual type algorithm which can solve convex program (1)-(3) with $O(1/\epsilon)$ convergence. That is, the new algorithm only requires $O(1/\epsilon)$ iterations to achieve an $\epsilon$-approximate solution. Furthermore, each iteration of this new algorithm can be decomposed into multiple smaller independent subproblems and hence can be implemented in parallel with low complexity even though the original convex program (1)-(3) involves non-separable $f(x)$ and $g(x)$.

### 1.1. Example Problems

The general convex program (1)-(3) considered in this paper includes many difficult convex programs as special cases.

**1.1.1. Large Scale Constrained Smooth Convex Programs.** If $\tilde{f}(x) \equiv 0$ and $\tilde{g}(x) \equiv 0$, then problem (1)-(3) is a constrained smooth convex program. In general, constrained smooth convex program (1)-(3) can be solved via interior point methods (or other Newton type methods) which involve the computation of Hessians and matrix inversions at each iteration. The associated computation complexity and memory space complexity at each iteration is between $O(n^2)$ and $O(n^3)$, which is prohibitive when $n$ is extremely large. For example, if $n = 10^5$ and each floating point number uses 4 bytes, then 40 Gbytes of memory space is required even to save the Hessian at each iteration. Thus, large scale convex programs are usually solved by first order gradient based methods or decomposition based methods. The primal-dual subgradient algorithm, also known as the Arrow-Hurwicz-Uzawa subgradient method, is a first order method with a slow $O(1/\epsilon^2)$ convergence time for large scale convex programs [12].

**1.1.2. Constrained Composite Convex Programs.** If $\tilde{f}(x) \neq 0$ and/or $\tilde{g}(x) \neq 0$, then problem (1)-(3) is a constrained composite convex program. Due to the non-differentiability, interior points methods (or other Newton type methods) are usually not applicable. Such a non-smooth convex program can be solved by a mirror descent based method in [1] with a slow $O(1/\epsilon^2)$ convergence time. In the special case when there is only one smooth constraint given by $g_1(x) \leq 0$, i.e., $\tilde{g}_1(x) \equiv 0$, work [16] proposes a dual method with an $O(1/\epsilon)$ convergence time. In the special case when $g(x) = Ax - b$ is linear and $\tilde{g}(x) \equiv 0$, work [6] proposes a random primal-dual method that can converges to a solution whose expected error is $\epsilon$ with an $O(1/\epsilon)$ convergence time.
One representative example of constrained composite convex programs is the constrained LASSO problem from machine learning applications [9] and financial portfolio optimization [4] as follows:

\[
\begin{align*}
\min_{x} & \quad \|Ax - b\|^2 + \lambda \|x\|_1 \\
\text{s.t.} & \quad Cx - d \leq 0 \\
\text{x} & \in X
\end{align*}
\]

where \( \|x\|_1 = \sum_{i=1}^{n} |x_i| \) denotes the \( l_1 \) norm of vector \( x \). The constrained LASSO problem is a special case of convex program (1)-(3) with \( f(x) = \|Ax - b\|^2 \), \( \tilde{f}(x) = \lambda \|x\|_1 \), \( g(x) = Cx - d \) and \( \tilde{g}(x) \equiv 0 \). In fact, many constrained optimization problems from machine learning, compressed sensing and financial portfolio optimization involve a non-smooth but separable \( l_1 \) norm \( \|x\|_1 \) term in the objective or constraint functions and can hence be cast as a special case of convex program (1)-(3).

1.2. The Primal-Dual Subgradient Method. The primal-dual subgradient method, also known as Arrow-Hurwicz-Uzawa subgradient method, with primal averaging is a first order method that can be applied to solve convex program (1)-(3) as described in Algorithm 1. In this paper, we use \( \nabla h(x(t-1)) \) to denote either the gradient (when \( h(\cdot) \) is differentiable) or a subgradient (when \( h(\cdot) \) is non-differentiable) of function \( h(x) \) at point \( x = x(t-1) \).

The primal-dual subgradient method can solve constrained non-smooth convex programs. The updates of \( x(t) \) and \( \lambda(t) \) only involve the computation of subgradients and simple projection operations. For large scale constrained smooth convex programs with large \( n \) value, the computation of subgradients is much simpler than the computation of Hessians and matrix inversions and hence the primal dual subgradient has lower complexity computations at each iteration and is more suitable when compared with the interior point method. However, Algorithm 1 is known to have a slow \( O(1/\epsilon^2) \) convergence time [12]. Another drawback of Algorithm 1 is that its implementation requires \( \lambda^\text{max}_k \), which are upper bounds of each component of the Lagrange multiplier vector \( \lambda^* \) that attains the strong duality. In practice, \( \lambda^* \) is usually unavailable.

1.3. The Dual Subgradient Method and Its Variations. The classical dual subgradient algorithm is a Lagrangian dual type iterative method that can solve constrained strictly convex programs [3]. By averaging the resulting primal estimates from the classical dual subgradient algorithm, we can solve general constrained convex programs (possibly without strict convexity) with an \( O(1/\epsilon^2) \) convergence time [13, 11, 14]. The dual subgradient algorithm with primal averaging is more suitable to separable convex programs because the updates of each component \( x_i(t) \) are independent and parallel if both the objective function and the constraint function are separable \( x \).

Recently, a new Lagrangian dual type algorithm with \( O(1/\epsilon) \) convergence for general convex programs is proposed in [18]. This algorithm can solve convex program (1)-(3) following the steps described in Algorithm 2. Similar to the dual subgradient algorithm with primal averaging, Algorithm 2 can decompose the updates of \( x(t) \) into smaller independent subproblems if functions \( F(x) \) and \( G_k(x) \) are separable. Moreover, Algorithm 2 has faster \( O(1/\epsilon) \) convergence when compared with the primal-dual subgradient algorithm or the dual subgradient algorithm with primal averaging.

In this paper, however, objective function \( F(x) \) involves possibly non-separable \( f(x) \) and each constraint function \( G_k(x) \) involves possibly non-separable \( g_k(x) \). As a
Algorithm 1 The Primal-Dual Subgradient Algorithm
Let $c > 0$ be a constant step size. Choose any $x(0) \in X$. Initialize Lagrangian multipliers $\lambda_k(0) = 0, \forall k \in \{1, 2, \ldots, m\}$. At each iteration $t \in \{1, 2, \ldots\}$, observe $x(t-1)$ and $\lambda(t-1)$ and do the following:

- Choose $x(t)$ via
  \[
x(t) = \mathcal{P}_X \left[ x(t-1) - c (\nabla F(x(t-1)) + \sum_{k=1}^{m} \lambda_k(t-1) \nabla G_k(x(t-1))) \right],
  \]
  where $\mathcal{P}_X[\cdot]$ is the projection onto convex set $X$.
- Update Lagrangian multipliers $\lambda(t)$ via
  \[
  \lambda_k(t) = [\lambda_k(t-1) + cG_k(x(t-1))]^{\lambda_k^{\text{max}}}, \forall k \in \{1, 2, \ldots, m\},
  \]
  where $\lambda_k^{\text{max}} > \lambda_k^* \text{ and } [\cdot]^{\lambda_k^{\text{max}}}$ is the projection onto interval $[0, \lambda_k^{\text{max}}]$.
- Update the running averages $\bar{x}(t)$ via
  \[
  \bar{x}(t+1) = \frac{1}{t+1} \sum_{\tau=0}^{t} x(\tau) = \bar{x}(t) \frac{t}{t+1} + x(t) \frac{1}{t+1}.
  \]

Algorithm 2 Algorithm 1 in [18]
Let $\alpha > 0$ be a constant parameter. Choose any $x(-1) \in X$. Initialize virtual queues $Q_k(0) = \max \{0, -G_k(x(-1))\}, \forall k \in \{1, 2, \ldots, m\}$. At each iteration $t \in \{0, 1, 2, \ldots\}$, observe $x(t-1)$ and $Q(t)$ and do the following:

- Choose $x(t)$ as
  \[
  x(t) = \arg\min_{x \in X} \left\{ F(x) + \left[ Q(t) + G(x(t-1)) \right]^T G(x) + \alpha \|x - x(t-1)\|^2 \right\}.
  \]
- Update virtual queue vector $Q(t)$ via
  \[
  Q_k(t+1) = \max \{-G_k(x(t)), Q_k(t) + G_k(x(t))\}, \forall k \in \{1, 2, \ldots, m\}.
  \]
- Update the running averages $\bar{x}(t)$ via
  \[
  \bar{x}(t+1) = \frac{1}{t+1} \sum_{\tau=0}^{t} x(\tau) = \bar{x}(t) \frac{t}{t+1} + x(t) \frac{1}{t+1}.
  \]

result, the update of $x(t)$ is not decomposable and requires to solve a set constrained non-smooth strongly convex program which is typically solved via a subgradient based method. However, the subgradient based method for set constrained convex programs is an iterative technique and involves at least one projection operation at each iteration. To obtain an $\epsilon$-approximate solution to the set constrained convex program, the projected subgradient method in general requires $O(1/\epsilon^2)$ iterations and can be improved to only require $O(1/\epsilon)$ or $O(1/\sqrt{\epsilon})$ iterations for certain special problems [15, 2].
1.4. New Algorithm. In this paper, we propose a new primal-dual type algorithm to solve convex program (1)-(3) as described in Algorithm 3. The new algorithm uses the same virtual queue update as in Algorithm 2, however, the update of $x(t)$ is fundamentally different. The modification enable us to update each component of $x$ in parallel even if $f(x)$ or each $g_k(x)$ is non-separable. Later, we will further show that the $O(1/e)$ convergence of Algorithm 2 is preserved in the new algorithm.

**Algorithm 3 New Algorithm**

Let $\{\alpha(t), t \geq 0\}$ be a sequence of positive algorithm parameters (defined in Section 3). Choose any $x(-1) \in \mathcal{X}$. Initialize virtual queues $Q_k(0) = \max\{0, -G_k(x(-1))\}, \forall k \in \{1, 2, \ldots, m\}$. At each iteration $t \in \{0, 1, \ldots\}$, observe $x(t-1)$ and $Q(t)$ and do the following:

- Choose $x(t)$ to solve $\min_{x \in \mathcal{X}} \left\{ [\nabla f(x(t-1))]^T x + \hat{f}(x) + \sum_{k=1}^m \left[ Q_k(t) + G_k(x(t-1)) \right] \left[ [\nabla g_k(x(t-1))]^T x + \hat{g}_k(x) \right] + \alpha(t) \|x - x(t-1)\|^2 \right\}$.
- Update virtual queue vector $Q(t)$ via $Q_k(t+1) = \max\{-G_k(x(t)), Q_k(t) + G_k(x(t))\}, \forall k \in \{1, 2, \ldots, m\}$.
- Update the running averages $\bar{x}(t)$ via

$$\bar{x}(t+1) = \frac{1}{t+1} \sum_{\tau=0}^t x(\tau) = \bar{x}(t) + x(t) \frac{1}{t+1}.$$ 

At each iteration $t \in \{0, 1, 2, \ldots\}$, $Q_k(t) + G_k(x(t-1)), \forall k \in \{1, 2, \ldots, m\}$ are given constants. Note that $[\nabla f(x(t-1))]^T x + \sum_{k=1}^m \left[ Q_k(t) + G_k(x(t-1)) \right] \left[ [\nabla g_k(x(t-1))]^T x \right]$ is a linear function and hence is separable; $\hat{f}(x) + \sum_{k=1}^m \left[ Q_k(t) + G_k(x(t-1)) \right] \hat{g}_k(x)$ is separable by assumption; and $\alpha(t) \|x - x(t-1)\|^2$ is also separable. Thus, the update of $x(t)$ requires to minimize a separable convex function. It follows that each component of $x(t)$ can be updated independently by solving a scalar convex program. Thus, each iteration of Algorithm 3 is parallel and has low complexity.

The next lemma shows that if $\hat{f}(x)$ and each $\hat{g}_k(x)$ are $l_1$ norms, then the $x(t)$ update in Algorithm 3 has a closed-form update equation for each coordinate.

**Lemma 1.** If $\hat{f}(x) = c_0 \|x\|_1$, $\hat{g}_k(x) = c_k \|x\|_1$, $\forall k \in \{1, 2, \ldots, m\}$ and $\mathcal{X} = \prod_{i=1}^n [x_i^{\min}, x_i^{\max}]$, then the following holds:

1. The update of $x(t)$ in Algorithm 3 can be decomposed into two scalar convex programs and each $x_i(t), i \in \{1, 2, \ldots, n\}$ is the solution to a scalar convex program given by

$$\min_{x_i \in [x_i^{\min}, x_i^{\max}]} \left\{ \alpha x_i - x_i(t-1) \right\}^2 + d_i x_i + e_i \cdot x_i,$$

where $\alpha = \alpha(t) > 0$, $d_i = \frac{\partial \hat{f}(x(t-1))}{\partial x_i} + \sum_{k=1}^m \left[ Q_k(t) + G_k(x(t-1)) \right] \frac{\partial \hat{g}_k(x(t-1))}{\partial x_i}$, and $e_i = c_0 + \sum_{k=1}^m \left[ Q_k(t) + G_k(x(t-1)) \right] c_k$ are constants. Note that we use $\frac{\partial \phi(x(t-1))}{\partial x_i}$ to denote the partial gradient of $\phi(x)$ with respect to the $i$-th component $x_i$ at point $x = x(t-1)$.
2. Scalar convex program (4) has a closed-form solution given by

\[ x^*_i = \begin{cases} 
    x_i(t-1) - \frac{d_i}{2\alpha} - \frac{c_i}{2\alpha}t_{x_i}^{\max}, & \text{if } x_i(t-1) - \frac{d_i}{2\alpha} > \frac{c_i}{2\alpha}, \\
    x_i(t-1) - \frac{d_i}{2\alpha} + \frac{c_i}{2\alpha}t_{x_i}^{\min}, & \text{if } x_i(t-1) - \frac{d_i}{2\alpha} < -\frac{c_i}{2\alpha}, \\
    [0]_{x_i}^{\max}, & \text{else.}
\end{cases} \]

where \([.]_{a}^{b}\) is the projection onto the interval \([a, b]\).

**Proof.**
1. The first part follows trivially by combining terms of each component \(x_i\) in the convex program with vector variable \(x\).
2. The second part follows by recalling that the subgradient of \(|x|\) is 1 when \(x > 0\); is -1 when \(x < 0\); and is the interval \([-1, 1]\) when \(x = 0\). The closed-form solution is obtained by considering different ranges of parameters. \(\square\)

The next lemma summarizes that if \(
\tilde{f}(x) \equiv 0 \text{ and } \tilde{g}(x) \equiv 0\)
, i.e., problem (1)-(3)
is a constrained smooth convex program, then \(x(t)\) update in Algorithm 3 follows a simple projected gradient update, which is parallel for each component as long as \(X\) is a Cartesian product.

**Lemma 2.** If \(\tilde{f}(x) \equiv 0 \text{ and } \tilde{g}(x) \equiv 0\), then the update of \(x(t)\) in Algorithm 3 is given by

\[ x(t) = \mathcal{P}_X \left[ x(t-1) - \frac{1}{2\alpha(t)} d(t) \right] \]

where \(d(t) = \nabla f(x(t-1)) + \sum_{k=1}^{m} (Q_k(t) + g_k(x(t-1))) \nabla g_k(x(t-1))\) and \(\mathcal{P}_X[.]\) is the projection onto convex set \(X\).

**Proof.** The projection operator can be reinterpreted as an optimization problem as follows:

\[
\begin{align*}
    x(t) &= \mathcal{P}_X \left[ x(t-1) - \frac{1}{2\alpha(t)} d(t) \right] \\
    &= \left( \arg\min_{x \in X} \left\| x - [x(t-1) - \frac{1}{2\alpha(t)} d(t)] \right\|^2 \right) \\
    &= \left( \arg\min_{x \in X} \left\| x - x(t-1) \right\|^2 + \frac{1}{\alpha(t)} d^T(t) [x - x(t-1)] + \frac{1}{4\alpha^2(t)} \|d(t)\|^2 \right) \\
    &= \left( \arg\min_{x \in X} \left[ d^T(t)x + \alpha(t) \|x - x(t-1)\|^2 \right] \right),
\end{align*}
\]

where (a) follows from the definition of the projection onto a convex set; and (b) follows from the fact the minimizing solution does not change when we remove constant term \(-\frac{1}{\alpha(t)} d^T(t)x(t-1) + \frac{1}{4\alpha^2(t)} \|d(t)\|^2\) and multiply positive constant \(\alpha(t)\) in the objective function.

Recall that \(d(t) = \nabla f(x(t-1)) + \sum_{k=1}^{m} (Q_k(t) + g_k(x(t-1))) \nabla g_k(x(t-1))\). This lemma follows because (5) is identical to the update of \(x(t)\) in Algorithm 3 when \(\tilde{f}(x) \equiv 0 \text{ and } \tilde{g}(x) \equiv 0\). \(\square\)

For constrained smooth convex programs, Lemma 2 suggests that Algorithm 3 has a similar per-iteration complexity when compared with Algorithm 1. However, Algorithm 3 can be more easily implemented since it does not require any upper bound of \(X^*\) as required by Algorithm 1. Moreover, we shall show that Algorithm 2 has faster \(O(1/\epsilon)\) convergence in comparison with the slow \(O(1/\epsilon^2)\) convergence of Algorithm 1.
2. Preliminaries and Basis Analysis. This section presents useful preliminaries on convex analysis and important facts of Algorithm 3.

2.1. Preliminaries.

**Definition 1 (Lipschitz Continuity).** Let \( \mathcal{X} \subseteq \mathbb{R}^n \) be a convex set. Function \( \phi : \mathcal{X} \to \mathbb{R}^m \) is said to be Lipschitz continuous on \( \mathcal{X} \) with modulus \( L > 0 \) if there exists such that \( \|\phi(y) - \phi(x)\| \leq L\|y - x\| \) for all \( x, y \in \mathcal{X} \).

**Definition 2 (Smooth Functions).** Let \( \mathcal{X} \subseteq \mathbb{R}^n \) and function \( \phi(x) \) be continuously differentiable on \( \mathcal{X} \). Function \( \phi(x) \) is said to be smooth on \( \mathcal{X} \) with modulus \( L \) if \( \nabla \phi(x) \) is Lipschitz continuous on \( \mathcal{X} \) with modulus \( L \).

Note that linear function \( \phi(x) = a^Tx \) is smooth with modulus 0. If a function \( \phi(x) \) is smooth with modulus \( L \), then \( c\phi(x) \) is smooth with modulus \( cL \) for any \( c > 0 \).

**Lemma 3 (Descent Lemma, Proposition A.24 in [3]).** If \( h \) is smooth on \( \mathcal{X} \) with modulus \( L \), then \( \phi(y) \leq \phi(x) + \nabla \phi(x)^T(y - x) + \frac{L}{2}\|y - x\|^2 \) for all \( x, y \in \mathcal{X} \).

**Definition 3 (Strongly Convex Functions).** Let \( \mathcal{X} \subseteq \mathbb{R}^n \) be a convex set. Function \( \phi \) is said to be strongly convex on \( \mathcal{X} \) with modulus \( \alpha > 0 \) if there exists a constant \( \alpha > 0 \) such that \( \phi(x) - \frac{1}{2}\alpha\|x\|^2 \) is convex on \( \mathcal{X} \).

By the definition of strongly convex functions, it is easy to show that if \( \phi(x) \) is convex and \( \alpha > 0 \), then \( \phi(x) + \alpha\|x - x_0\|^2 \) is strongly convex with modulus \( 2\alpha \) for any constant \( x_0 \).

**Lemma 4 (Corollary 1 in [18]).** Let \( \mathcal{X} \subseteq \mathbb{R}^n \) be a convex set. Let function \( \phi \) be strongly convex on \( \mathcal{X} \) with modulus \( \alpha \) and \( x^{\text{opt}} \) be a global minimum of \( h \) on \( \mathcal{X} \). Then, \( \phi(x^{\text{opt}}) \leq \phi(x) - \frac{\alpha}{2}\|x^{\text{opt}} - x\|^2 \) for all \( x \in \mathcal{X} \).

2.2. Properties of the Virtual Queue Vector and the Drift. The following preliminary results (Lemmas 5-6) on virtual queue vector \( Q(t) \) and its drift are proven for Algorithm 2 in [18] and hold regardless of the update of \( x(t) \). Since Algorithm 3 has the same update equation of \( Q(t) \), these results also hold for Algorithm 3.

**Lemma 5 (Lemma 3 in [18]).** In Algorithm 3, we have

1. At each iteration \( t \in \{0, 1, 2, \ldots\} \), \( Q_k(t) \geq 0 \) for all \( k \in \{1, 2, \ldots, m\} \).
2. At each iteration \( t \in \{0, 1, 2, \ldots\} \), \( Q_k(t) + G_k(x(t-1)) \geq 0 \) for all \( k \in \{1, 2, \ldots, m\} \).
3. At iteration \( t=0 \), \( \|Q(0)\| \leq \|G(x(-1))\| \). At each iteration \( t \in \{1, 2, \ldots\} \), \( \|Q(t)\|^2 \geq \|G(x(t-1))\|^2 \).

**Lemma 6 (Lemma 7 in [18]).** Let \( Q(t), t \in \{0, 1, \ldots\} \) be the sequence generated by Algorithm 3. For any \( t \geq 1 \),

\[ Q_k(t) \geq \sum_{\tau=0}^{t-1} G_k(x(\tau)), \forall k \in \{1, 2, \ldots, m\}. \]

Let \( Q(t) = [Q_1(t), \ldots, Q_m(t)]^T \) be the vector of virtual queue backlogs. Define \( L(t) = \frac{1}{2}\|Q(t)\|^2 \). The function \( L(t) \) shall be called a Lyapunov function. Define the Lyapunov drift as

\[ \Delta(t) = L(t+1) - L(t) = \frac{1}{2}(\|Q(t+1)\|^2 - \|Q(t)\|^2). \]
At each iteration $t \in \{0, 1, 2, \ldots\}$ in Algorithm 3, an upper bound of the Lyapunov drift is given by
\begin{equation}
\Delta(t) \leq Q^T(t)G(x(t)) + \|G(x(t))\|^2.
\end{equation}

2.3. Properties from Strong Duality. The next lemma follows from Lemma 6 and Assumption 2.

**Lemma 8.** Let $x^*$ be an optimal solution of problem (1)-(3) and $\lambda^*$ be a Lagrange multiplier vector satisfying Assumption 2. Let $x(t), Q(t), t \in \{0, 1, \ldots\}$ be sequences generated by Algorithm 3. Then,
\[\sum_{\tau=0}^{t-1} F(x(\tau)) \geq tF(x^*) - \|\lambda^*\|\|Q(t)\|, \quad \forall t \geq 1.\]

**Proof.** The proof is quite similar to the proof of Lemma 8 in [18]. Define Lagrangian dual function $q(\lambda) = \min_{x \in X} \{F(x) + \sum_{k=1}^{m} \lambda_k G_k(x)\}$. For all $\tau \in \{0, 1, \ldots\}$, by Assumption 2, we have
\[F(x^*) = q(\lambda^*) \leq F(x(\tau)) + \sum_{k=1}^{m} \lambda^*_k G_k(x(\tau)), \]
where (a) follows the definition of $q(\lambda^*)$.

Thus, we have $F(x(\tau)) \geq F(x^*) - \sum_{k=1}^{m} \lambda^*_k G_k(x(\tau)), \forall \tau \in \{0, 1, \ldots\}$. Summing over $\tau \in \{0, 1, \ldots, t-1\}$ yields
\[\sum_{\tau=0}^{t-1} F(x(\tau)) \geq tF(x^*) - \sum_{\tau=0}^{t-1} \sum_{k=1}^{m} \lambda^*_k G_k(x(\tau))\]
\[= tF(x^*) - \sum_{k=1}^{m} \lambda^*_k \left[ \sum_{\tau=0}^{t-1} G_k(x(\tau)) \right]\]
\[\geq tF(x^*) - \sum_{k=1}^{m} \lambda^*_k Q_k(t)\]
\[\geq tF(x^*) - \|\lambda^*\|\|Q(t)\|,\]
where (a) follows from Lemma 6 and the fact that $\lambda^*_k \geq 0, \forall k \in \{1, 2, \ldots, m\}$; and (b) follows from the Cauchy-Schwarz inequality.

2.4. An Upper Bound of the Drift-Plus-Penalty Expression.

**Lemma 9.** Let $x^*$ be an optimal solution of problem (1)-(3). For all $t \geq 0$ in Algorithm 3, we have
\[\Delta(t) + F(x(t))\]
\[\leq F(x^*) + \alpha(t)[\|x^* - x(t-1)\|^2 - \|x^* - x(t)\|^2] + \frac{1}{2}[\|G(x(t))\|^2 - \|G(x(t-1))\|^2]\]
\[+ \left[\frac{\beta^2 + L_f + \|Q(t) + G(x(t-1))\|^2}{2}\right] \mathbf{L}_g - \alpha(t)\|x(t) - x(t-1)\|^2\]
\[\text{where} \ \beta, L_f \text{ and } \mathbf{L}_g \text{ are defined in Assumption 1.}\]
Proof. Fix $t \geq 0$. By part (2) in Lemma 5, $Q_k(t) + G_k(x(t-1)) \geq 0, \forall k \in \{1, 2, \ldots, m\}$. Thus, $\sum_{k=1}^{m} [Q_k(t) + G_k(x(t-1))] [\nabla g_k(x(t-1))] + \alpha(t) \| x(t-1) - x(t) \|^2$ is strongly convex with respect to $x \in X$. Since $\alpha(t) \| x(t-1) - x(t) \|^2$ is strongly convex with respect to $x$ with modulus $2\alpha(t)$, it follows that $\sum_{k=1}^{m} [Q_k(t) + G_k(x(t-1))] [\nabla g_k(x(t-1))] + \alpha(t) \| x(t-1) - x(t) \|^2$ is strongly convex with respect to $x$ with modulus $2\alpha(t)$.

Since $x(t)$ is chosen to minimize the above strongly convex function, by Lemma 4, we have

$$
\left[\nabla f(x(t-1)) + \tilde{f}(x(t))\right]^T x(t) + \tilde{f}(x(t)) + \alpha(t) \| x(t) - x(t-1) \|^2 \\
\leq \left[\nabla f(x(t-1))\right]^T x^* + \tilde{f}(x^*) \sum_{k=1}^{m} [Q_k(t) + G_k(x(t-1))] [\nabla g_k(x(t-1))] + \alpha(t) \| x^* - x(t-1) \|^2
$$

Adding constant $f(x(t-1)) - [\nabla f(x(t-1))]^T x(t-1) + \sum_{k=1}^{m} [Q_k(t) + G_k(t-1)] [g_k(x(t-1))] - [\nabla g_k(x(t-1))]^T x(t-1)$ on both sides and rearranging terms yields

$$
f(x(t-1)) - [\nabla f(x(t-1))]^T x(t-1) + \tilde{f}(x(t)) + \sum_{k=1}^{m} [Q_k(t) + G_k(x(t-1))] [g_k(x(t-1))] - [\nabla g_k(x(t-1))]^T x(t-1)
$$

$$
= \sum_{k=1}^{m} [Q_k(t) + G_k(x(t-1))] [g_k(x(t-1))] - [\nabla g_k(x(t-1))]^T x^* + \tilde{f}(x^*)
$$

$$
- \alpha(t) \| x(t) - x(t-1) \|^2 + \alpha(t) \| x^* - x(t-1) \|^2 - \| x^* - x(t) \|^2
$$

$$(a) \leq f(x^*) + \tilde{f}(x^*) + \sum_{k=1}^{m} [Q_k(t) + G_k(x(t-1))] [g_k(x^*) + \tilde{g}_k(x^*)]
$$

$$
- \alpha(t) \| x(t) - x(t-1) \|^2 + \alpha(t) \| x^* - x(t-1) \|^2 - \| x^* - x(t) \|^2
$$

$$(b) \leq F(x^*) - \alpha(t) \| x(t) - x(t-1) \|^2 + \alpha(t) \| x^* - x(t-1) \|^2 - \| x^* - x(t) \|^2$$

where (a) follows from the convexity of $f(\cdot)$ and each $g_k(\cdot)$, and the fact that $Q_k(t) + G_k(x(t-1)) \geq 0, \forall k \in \{1, 2, \ldots, m\}$ (i.e., part (2) in Lemma 5); and (b) follows because $F(x^*) = f(x^*) + \tilde{f}(x^*), G_k(x^*) = g_k(x^*) + \tilde{g}_k(x^*) \leq 0, \forall k \in \{1, 2, \ldots, m\}$, which further follows from the feasibility of $x^*$, and $Q_k(t) + G_k(x(t-1)) \geq 0, \forall k \in \{1, 2, \ldots, m\}$ (i.e., part (2) in Lemma 5).
Recall that \( f(x) \) is smooth on \( \mathcal{X} \) with modulus \( L_f \) by Assumption 1. By Lemma 3, we have

\[
f(x(t)) \leq f(x(t - 1)) + [\nabla f(x(t - 1))]^T [x(t) - x(t - 1)] + \frac{L_f}{2} \|x(t) - x(t - 1)\|^2.
\]

Adding \( \tilde{f}(x(t)) \) on both sides, recalling \( F(x) = f(x) + \tilde{f}(x) \) and rearranging terms yields

\[
F(x(t)) - \frac{L_f}{2} \|x(t) - x(t - 1)\|^2 \leq f(x(t - 1)) + [\nabla f(x(t - 1))]^T [x(t) - x(t - 1)] + \tilde{f}(x(t)).
\]

(9)

Recall that each \( g_k(x) \) is smooth on \( \mathcal{X} \) with modulus \( L_{g_k} \) by Assumption 1. Thus, each \( [Q_k(t) + G_k(x(t - 1))]g_k(x) \) is smooth with modulus \( [Q_k(t) + G_k(x(t - 1))]L_{g_k} \). By Lemma 3, we have

\[
\begin{align*}
&[Q_k(t) + G_k(x(t - 1))]g_k(x(t)) \\
\leq &\sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))]g_k(x(t - 1)) \\
&+ [Q_k(t) + G_k(x(t - 1))]^T [\nabla g_k(x(t - 1))] [x(t) - x(t - 1)] \\
&+ \frac{1}{2} \|Q(t) + G(x(t - 1))\|^2 L_{g_k} \|x(t) - x(t - 1)\|^2.
\end{align*}
\]

(10)

Summing (10) over \( k \in \{1, 2, \ldots, m\} \) yields

\[
\sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))]g_k(x(t)) \\
\leq \sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))]g_k(x(t - 1)) + [\nabla g_k(x(t - 1))]^T [x(t) - x(t - 1)] \\
+ \frac{1}{2} \|Q(t) + G(x(t - 1))\|^2 L_{g_k} \|x(t) - x(t - 1)\|^2.
\]

Adding \( \sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))]\tilde{g}_k(x(t)) \) on both sides, recalling \( G_k(x(t)) = g_k(x(t)) + \tilde{g}_k(x(t)) \) and rearranging terms yields

\[
\begin{align*}
&\sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))]g_k(x(t)) - \frac{1}{2} \|Q(t) + G(x(t - 1))\|^2 L_{g_k} \|x(t) - x(t - 1)\|^2 \\
\leq &\sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))]g_k(x(t - 1)) + [\nabla g_k(x(t - 1))]^T [x(t) - x(t - 1)] \\
+ &\tilde{g}_k(x(t)).
\end{align*}
\]

(11)
Substituting (14) into (13) and rearranging terms yields
\[
\| \alpha \|_2
\]
where (a) follows from the fact that
\[
combining (8) and (12); and rearranging terms, we have
\[
\beta
\]
which further follows from the assumption that
\[
G \leq F
\]
\[
\Delta(\beta)
\]
Summing up (9) and (11) together yields
\[
F(x(t)) + [Q(t) + G(x(t - 1))]^T G(x(t))
\]
\[
L_f + [Q(t) + G(x(t - 1))]^T L_g \| x(t) - x(t - 1) \|^2
\]
\[
\leq f(x(t)) + [\nabla f(x(t - 1))]^T [x(t) - x(t - 1)] + f(x(t))
\]
\[
\sum_{k=1}^m [Q_k(t) + G_k(x(t - 1))] [g_k(x(t - 1)) + [\nabla g_k(x(t - 1))]^T [x(t) - x(t - 1)]
\]
(12)
\[
+ \tilde{g}_k(x(t))
\]
Note that the right side of (12) is identical to the left side of (8). Thus, by combining (8) and (12); and rearranging terms, we have
\[
F(x(t)) + [Q(t) + G(x(t - 1))]^T G(x(t))
\]
\[
\leq F(x^*) + \alpha(t) \left[ \| x^* - x(t - 1) \|^2 - \| x^* - x(t) \|^2 \right]
\]
\[
+ \left[ \frac{L_f + [Q(t) + G(x(t - 1))]^T L_g}{2} - \alpha(t) \right] \| x(t) - x(t - 1) \|^2.
\]
Note that \( u^T v = \frac{1}{2} \| u \|^2 + \| v \|^2 - \| u - v \|^2 \) for any \( u, v \in \mathbb{R}^m \). Thus, we have
\[
\left[ G(x(t - 1)) \right]^T G(x(t)) \leq \frac{1}{2} \left[ \| G(x(t - 1)) \|^2 + \| G(x(t)) \|^2 - \| G(x(t - 1)) - G(x(t)) \|^2 \right].
\]
Substituting (14) into (13) and rearranging terms yields
\[
F(x(t)) + Q^T(t) G(x(t))
\]
\[
\leq F(x^*) + \alpha(t) \left[ \| x^* - x(t - 1) \|^2 - \| x^* - x(t) \|^2 \right]
\]
\[
+ \left[ \frac{L_f + [Q(t) + G(x(t - 1))]^T L_g}{2} - \alpha(t) \right] \| x(t) - x(t - 1) \|^2
\]
\[
+ \frac{1}{2} \| G(x(t - 1)) - G(x(t)) \|^2 - \frac{1}{2} \| G(x(t - 1)) \|^2 - \frac{1}{2} \| G(x(t)) \|^2
\]
(13)
\[
\leq F(x^*) + \alpha(t) \left[ \| x^* - x(t - 1) \|^2 - \| x^* - x(t) \|^2 \right]
\]
\[
+ \left[ \frac{\beta^2 + L_f + [Q(t) + G(x(t - 1))]^T L_g}{2} - \alpha(t) \right] \| x(t) - x(t - 1) \|^2
\]
\[
- \frac{1}{2} \| G(x(t - 1)) \|^2 - \frac{1}{2} \| G(x(t)) \|^2,
\]
where (a) follows from the fact that \( \| G(x(t - 1)) - G(x(t)) \| \leq \beta \| x(t) - x(t - 1) \| \), which further follows from the assumption that \( G(x) \) is Lipschitz continuous with modulus \( \beta \).

Summing (7) to the above inequality yields
\[
\Delta(t) + F(x(t))
\]
\[
\leq F(x^*) + \alpha(t) \left[ \| x^* - x(t - 1) \|^2 - \| x^* - x(t) \|^2 \right] + \frac{1}{2} \left[ \| G(x(t)) \|^2 - \| G(x(t - 1)) \|^2 \right]
\]
\[
+ \left[ \frac{\beta^2 + L_f + [Q(t) + G(x(t - 1))]^T L_g}{2} - \alpha(t) \right] \| x(t) - x(t - 1) \|^2
\]
The next corollary follows directly by noting that $L_g = 0$ when each $g_k(x)$ is a linear function.

**Corollary 1.** Let $x^*$ be an optimal solution of problem (1)-(3) where each $g_k(x)$ is a linear function. If $\alpha(t) = \alpha > \frac{1}{2}[\beta^2 + L_f], \forall t \geq 0$ in Algorithm 3, then for all $t \geq 0$, we have

$$
\Delta(t) + F(x(t)) \\
\leq F(x^*) + \alpha \left[ \| x^* - x(t-1) \|^2 - \| x^* - x(t) \|^2 \right] + \frac{1}{2} \left[ \| G(x(t)) \|^2 - \| G(x(t-1)) \|^2 \right]
$$

where $\beta$ and $L_f$ are defined in Assumption 1.

**Proof.** Note that if each $g_k(x)$ is a linear function, then we have $L_g = 0$. Fix $t \geq 0$. By Lemma 9 with $\alpha(t) = \alpha$ and $L_g = 0$, we have

$$
\Delta(t) + F(x(t)) \\
\leq F(x^*) + \alpha \left[ \| x^* - x(t-1) \|^2 - \| x^* - x(t) \|^2 \right] + \frac{1}{2} \left[ \| G(x(t)) \|^2 - \| G(x(t-1)) \|^2 \right] + \frac{\beta^2 + L_f}{2} \| x(t) - x(t-1) \|^2
$$

where $(a)$ follows from $\alpha > \frac{1}{2}[\beta^2 + L_f]$. $\square$

3. **Convergence Time Analysis of Algorithm 3.** This section analyzes the convergence time of Algorithm 3 for convex program (1)-(3). In particular, the following two rules for choosing $\alpha(t)$ in Algorithm 3 are considered.

- **Constant $\alpha(t)$:** Choose algorithm parameters $\alpha(t)$ via

  $$(15) \quad \alpha(t) = \alpha > \frac{1}{2}[\beta^2 + L_f], \forall t \geq 0$$

- **Non-decreasing $\alpha(t)$:** Choose algorithm parameters $\alpha(t)$ via

  $$(16) \quad \alpha(t) = \begin{cases} \frac{1}{2} \left[ \beta^2 + L_f + [Q(0)^T + G(x(-1))]^T L_g \right], & t = 0 \\ \max \left\{ \alpha(t-1), \frac{1}{2} \left[ \beta^2 + L_f + [Q(t) + G(x(t-1))]^T L_g \right] \right\}, & t \geq 1 \end{cases}$$

Note that part (2) of Lemma 5 implies $\alpha(0) > 0$, and hence $\alpha(t) > 0, \forall t \geq 0$ since $\alpha(t)$ is a nondecreasing sequence.

3.1. **Convex Programs with Linear $g(x)$**. This subsection proves that if each $g_k(x)$ is a linear function, then it suffices to choose constant parameters $\alpha(t) = \alpha > \frac{1}{2}[\beta^2 + L_f]$ in Algorithm 3 to solve convex program (1)-(3) with an $O(1/\epsilon)$ convergence time.

**Theorem 1.** Consider convex program (1)-(3) under Assumption 1 and Assumption 2 where each $g_k(x)$ is a linear function. Let $x^*$ be an optimal solution. Let $\lambda^*$ be a Lagrange multiplier vector satisfying Assumption 2. If we choose constant $\alpha(t)$ in Algorithm 3 according to (15), then for all $t \geq 1$, we have

1. $F(x(t)) \leq F(x^*) + \frac{\alpha}{\epsilon} \| x^* - x(-1) \|^2$.
2. $G_k(x(t)) \leq \frac{1}{t} \left[ \| \lambda^* \|^2 + \sqrt{2\alpha \| x^* - x(t-1) \|^2} + \sqrt{\frac{\alpha}{\frac{1}{2} \beta^2 + \frac{1}{4} L_f}} \| G(x^*) \| \right]$. 


where \( \beta \) and \( L_f \) are defined in Assumption 1. That is, Algorithm 3 ensures error decays like \( \tilde{O}(1/t) \) and provides an \( \epsilon \)-approximate solution with convergence time \( O(1/\epsilon) \).

**Proof.**

1. By Corollary 1, we have \( F(x(\tau)) \leq F(x^*) + \alpha \left[ \|x^* - x(t-1)\|^2 - \|x^* - x(\tau)\|^2 \right] + \frac{1}{2} \left[ \|G(x(\tau))\|^2 - \|G(x(t-1))\|^2 \right] - \Delta(\tau) \) for all \( \tau \in \{0,1,2,\ldots\} \).

   Fix \( t \geq 1 \). Summing over \( \tau \in \{0,1,2,\ldots,t-1\} \) yields

   \[
   \sum_{\tau=0}^{t-1} F(x(\tau)) \leq tF(x^*) + \alpha \sum_{\tau=0}^{t-1} \left[ \|x^* - x(\tau-1)\|^2 - \|x^* - x(\tau)\|^2 \right] + \frac{1}{2} \sum_{\tau=0}^{t-1} \left[ \|G(x(\tau))\|^2 - \|G(x(\tau-1))\|^2 \right] - \sum_{\tau=0}^{t-1} \Delta(\tau)
   \]

   \[
   = tF(x^*) + \alpha \|x^* - x(-1)\|^2 - \alpha \|x^* - x(t-1)\|^2 + \frac{1}{2} \|G(x(t-1))\|^2 - \|G(x(-1))\|^2 + \frac{1}{2} \|Q(0)\|^2 - \frac{1}{2} \|Q(t)\|^2
   \]

   \[
   \leq tF(x^*) + \alpha \|x^* - x(-1)\|^2 - \alpha \|x^* - x(t-1)\|^2 + \frac{1}{2} \|G(x(t-1))\|^2 - \|G(x(-1))\|^2 + \frac{1}{2} \|Q(0)\|^2 - \frac{1}{2} \|Q(t)\|^2
   \]

   where (a) follows by recalling that \( \Delta(\tau) = \frac{1}{2} \|Q(\tau+1)\|^2 - \frac{1}{2} \|Q(\tau)\|^2 \) and (b) follows from \( \|Q(0)\|^2 \leq \|G(x(-1))\|^2 \) by part (3) in Lemma 5.

   Recalling that \( \|Q(t)\|^2 \geq \|G(x(t-1))\|^2 \) by part (3) in Lemma 5 and ignoring a negative term \( -\alpha \|x^* - x(t-1)\|^2 \) on the right side of (17) yields

   \[
   \sum_{\tau=0}^{t-1} F(x(\tau)) \leq tF(x^*) + \alpha \|x^* - x(-1)\|^2
   \]

   Dividing both sides by \( t \) and using Jensen’s inequality for convex function \( F(x) \) yields

   \[
   F(x(t)) \leq F(x^*) + \frac{\alpha}{t} \|x^* - x(-1)\|^2.
   \]

2. Fix \( t \geq 1 \). Note that (17) can be written as

   \[
   \sum_{\tau=0}^{t-1} F(x(\tau)) \leq tF(x^*) + \alpha \|x^* - x(-1)\|^2 - \alpha \|x^* - x(t-1)\|^2 + \frac{1}{2} \|G(x(t-1)) - G(x^*)\|^2 - \frac{1}{2} \|Q(t)\|^2
   \]

   \[
   = tF(x^*) + \alpha \|x^* - x(-1)\|^2 - \|x^* - x(t-1)\|^2 + \frac{1}{2} \|G(x(t-1)) - G(x^*)\|^2
   \]

   \[
   + \|G(x^*)\|^2 \left( G(x(t-1)) - G(x^*) \right) + \frac{1}{2} \|G(x^*)\|^2 - \frac{1}{2} \|Q(t)\|^2
   \]

13
Combining (18) and (19), cancelling common terms and rearranging terms yields

\[ tF(x^*) + \alpha \|x^* - x(-1)\|^2 - \alpha \|x^* - x(t-1)\|^2 + \frac{1}{2} \|G(x(t-1)) - G(x^*)\|^2 \\
+ \|G(x^*)\| \|G(x(t-1)) - G(x^*)\| + \frac{1}{2} \|G(x^*)\|^2 - \frac{1}{2} \|Q(t)\|^2 \]

\[ \leq tF(x^*) + \alpha \|x^* - x(-1)\|^2 - \alpha \|x^* - x(t-1)\|^2 + \frac{1}{2} \beta^2 \|x^* - x(t-1)\|^2 \\
+ \beta \|G(x^*)\| \|x^* - x(t-1)\| + \frac{1}{2} \|G(x^*)\|^2 - \frac{1}{2} \|Q(t)\|^2 \]

\[ = tF(x^*) + \alpha \|x^* - x(-1)\|^2 - (\alpha - \frac{1}{2} \beta^2) \left[ \|x^* - x(t-1)\| - \frac{\beta}{2\alpha - \beta^2} \|G(x^*)\|^2 \right] \\
+ \frac{\alpha}{2\alpha - \beta^2} \|G(x^*)\|^2 - \frac{1}{2} \|Q(t)\|^2 \]

(18)

where (a) follows from the Cauchy-Schwarz inequality; (b) follows from Lipschitz continuity of $G(x)$ in Assumption 1; and (c) follows from $\alpha > \frac{1}{2}[\beta^2 + L_f] \geq \frac{1}{2} \beta^2$.

By Lemma 8, we have

\[ \sum_{\tau=0}^{t-1} F(x(\tau)) \geq tF(x^*) - \|\lambda^*\| \|Q(t)\| \]

Combining (18) and (19), cancelling common terms and rearranging terms yields

\[ \frac{1}{2} \|Q(t)\|^2 - \|\lambda^*\| \|Q(t)\| - \alpha \|x^* - x(-1)\|^2 - \frac{\alpha}{2\alpha - \beta^2} \|G(x^*)\|^2 \leq 0 \]

\[ \Rightarrow \left[ \|Q(t)\| - \|\lambda^*\| \right]^2 \leq \|\lambda^*\|^2 + 2\alpha \|x^* - x(-1)\|^2 + \frac{\alpha}{\alpha - \frac{1}{2} \beta^2} \|G(x^*)\|^2 \]

\[ \Rightarrow \|Q(t)\| \leq \|\lambda^*\| + \sqrt{\|\lambda^*\|^2 + 2\alpha \|x^* - x(-1)\|^2 + \frac{\alpha}{\alpha - \frac{1}{2} \beta^2} \|G(x^*)\|^2} \]

(20)

\[ \Rightarrow \|Q(t)\| \leq 2 \|\lambda^*\| + \sqrt{2\alpha \|x^* - x(-1)\|} + \sqrt{\frac{\alpha}{\alpha - \frac{1}{2} \beta^2} \|G(x^*)\|} \]

where (a) follows from the basic inequality $\sqrt{z_1 + z_2 + z_3} \leq \sqrt{z_1} + \sqrt{z_2} + \sqrt{z_3}$ for any $z_1, z_2, z_3 \geq 0$.

Fix $k \in \{1, 2, \ldots, m\}$. By Jensen’s inequality for convex function $G_k(x)$, we have

\[ G_k(x(t)) \leq \frac{1}{t} \sum_{\tau=0}^{t-1} G_k(x(\tau)) \]

\[ \leq \frac{1}{t} Q_k(t) \]
\[ \leq \frac{1}{t} \|Q(t)\| \]

\[ \leq \frac{1}{t} \left[ 2\|x^*\| + \sqrt{2\alpha\|x^* - x(-1)\|} + \sqrt{\frac{\alpha + 1}{2\beta^2}} \|G(x^*)\| \right] \]

where (a) follows from Lemma 6 and (b) follows from (20).

### 3.2. Convex Programs with Possibly Non-linear \(g(x)\)

For convex program (1)-(3) with possibly nonlinear \(g(x)\), the following assumption is further assumed:

**Assumption 3.**

- There exists \(C > 0\) such that \(\|G(x)\| \leq C\) for all \(x \in \mathcal{X}\).
- There exists \(R > 0\) such that \(\|x - y\| \leq R\) for all \(x, y \in \mathcal{X}\).

Note that Assumption 3 holds when \(\mathcal{X}\) is a compact set.

This subsection proves that if convex program (1)-(3) with possibly nonlinear \(g(x)\) satisfies Assumptions 1-3, then it suffices to choose non-decreasing parameters \(\alpha(t)\) according to (16) in Algorithm 3 to solve convex program (1)-(3) with an \(O(1/\epsilon)\) convergence time.

**Lemma 10.** Consider convex program (1)-(3) under Assumptions 1-3. Let \(x^*\) be an optimal solution and \(\lambda^*\) be a Lagrange multiplier vector satisfying Assumption 2. If we choose non-decreasing \(\alpha(t)\) in Algorithm 3 according to (16), then we have

1. \(\sum_{\tau=0}^{t} \alpha(\tau) \left[ ||x^* - x(\tau - 1)||^2 - ||x^* - x(\tau)||^2 \right] \leq \alpha(t)R^2, \forall t \geq 0;\)
2. \(\sum_{\tau=0}^{t-1} F(x(\tau)) \leq tF(x^*) + \alpha(t-1)R^2 + \frac{1}{2}\|G(x(t-1))\|^2 - \frac{1}{2}\|Q(t)\|^2, \forall t \geq 1;\)
3. \(\|Q(t+1)\| \leq 2\|x^*\| + R\sqrt{2\alpha(t)} + C, \forall t \geq 0;\)

where \(R\) and \(C\) are defined in Assumption 3.

**Proof.**

1. This is obviously true when \(t = 0\). Fix \(t \geq 1\). Note that

\[
\sum_{\tau=0}^{t} \alpha(\tau) \left[ ||x^* - x(\tau - 1)||^2 - ||x^* - x(\tau)||^2 \right]
= \alpha(0) ||x^* - x(-1)||^2 + \sum_{\tau=0}^{t-1} \left[ \alpha(\tau + 1) - \alpha(\tau) \right] ||x^* - x(\tau)||^2 - \alpha(t) ||x^* - x(t)||^2
\]

\[\leq \alpha(0)R^2 + \sum_{\tau=0}^{t-1} \left[ \alpha(\tau + 1) - \alpha(\tau) \right] R^2\]

\[= \alpha(t)R^2\]

where (a) follows because \(||x^* - x(\tau)|| \leq R, \forall \tau \geq 0\) by Assumption 3 and \(\alpha(t+1) \geq \alpha(\tau), \forall \tau \geq 0\) by (16).

2. Fix \(t \geq 1\). By Lemma 9, for all \(\tau \in \{0, 1, 2, \ldots\}\), we have

\[
\Delta(\tau) + F(x(\tau))
\leq F(x^*) + \alpha(\tau) \left[ ||x^* - x(\tau - 1)||^2 - ||x^* - x(\tau)||^2 \right]
+ \frac{1}{2} \left[ ||G(x(\tau))||^2 - ||G(x(\tau - 1))||^2 \right]
+ \frac{\beta^2}{2} + L_f + \frac{[Q(\tau) + G(x(\tau - 1))]^T L_g - \alpha(\tau)}{2} \|x(\tau) - x(\tau - 1)\|^2
\]
∑_{τ=0}^{t-1} F(x(τ)) &≤ (a) t F(x^*) + (t - 1) R^2 + \frac{1}{2} \|G(x(t - 1))\|^2 - \frac{1}{2} \|Q(0)\|^2 \\
&+ \frac{1}{2} ||G(x(τ))||^2 - ||G(x(τ - 1))||^2 - \sum_{τ=0}^{t-1} \Delta(τ)

where (a) follows from part (1) of this lemma and by recalling that \(\Delta(τ) = \frac{1}{2} ||Q(τ + 1)||^2 - \frac{1}{2} ||Q(τ)||^2\); and (b) follows because \(||Q(0)||^2 \leq ||G(x(-1))||^2\) by part (3) in Lemma 5.

3. By part (2) of this lemma, we have

\(\sum_{τ=0}^{t} F(x(τ)) \leq (t + 1) F(x^*) + (t) R^2 + \frac{1}{2} ||G(x(τ))||^2 - \frac{1}{2} ||Q(t + 1)||^2\)

(21)

where (a) follows from \(||G(x(τ))|| \leq C\) by Assumption 3. By Lemma 8, we have

\(\sum_{τ=0}^{t} F(x(τ)) \geq (t + 1) F(x^*) - ||λ^*|| ||Q(t + 1)||\)

(22)

Combining (21) and (22), cancelling common terms and rearranging terms yields

\[\frac{1}{2} ||Q(t + 1)||^2 - ||λ^*|| ||Q(t + 1)|| - α(t) R^2 - \frac{1}{2} C^2 \leq 0\]

⇒ \[\||Q(t + 1)|| - ||λ^*||\]^2 \leq ||λ^*||^2 + 2α(t) R^2 + C^2\]

⇒ \[||Q(t + 1)|| \leq ||λ^*|| + \sqrt{||λ^*||^2 + 2α(t) R^2 + C^2}\]

(23)

\(\frac{(a)}{\leq} ||Q(t + 1)|| \leq 2 ||λ^*|| + \sqrt{2α(t) R + C}\)

where (a) follows from the basic inequality \(\sqrt{z_1 + z_2 + z_3} \leq \sqrt{z_1} + \sqrt{z_2} + \sqrt{z_3}\) for any \(z_1, z_2, z_3 \geq 0\).
Lemma 11. Consider convex program \((1) - (3)\) under Assumptions 1-3. If we choose non-decreasing \(\alpha(t)\) in Algorithm 3 according to \((16)\), then

\[
\alpha(t) \leq \alpha^{\text{max}}, \forall t \geq 0
\]

with constant

\[
\alpha^{\text{max}} = \left[ \sqrt{\frac{1}{2} \beta^2 + \frac{1}{2} L_f} + \|\lambda^*\| L_g + C \|L_g\| + \frac{\sqrt{2}}{2} R \|L_g\| \right]^2
\]

where \(\beta, L_f\) and \(L_g\) are defined in Assumption 1; and \(R\) and \(C\) are defined in Assumption 3.

Proof. This lemma can be proven by induction as follows. Note that by \((16)\), we have

\[
\alpha(0) = \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \frac{1}{2} [Q(0) + G(x(-1))]^T L_g
\]

\[
\leq \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \frac{1}{2} \|Q(0) + G(x(-1))\| L_g
\]

\[
\leq \frac{1}{2} \beta^2 + \frac{1}{2} L_f + C \|L_g\|
\]

\[
\leq \alpha^{\text{max}}
\]

where (a) follows from the Cauchy-Schwarz inequality; and (b) follows from \(\|Q(0) + G(x(-1))\| \leq \|Q(0)\| + \|G(x(-1))\| \leq 2 \|G(x(-1))\| \leq 2C\) where the second inequality follows from part (3) of Lemma 5 and the third inequality follows from Assumption 3. Thus, we have \(\alpha(0) \leq \alpha^{\text{max}}\).

Now assume \(\alpha(t) \leq \alpha^{\text{max}}\) holds for \(t = t_0\) and consider \(t = t_0 + 1\). By \((16)\), \(\alpha(t_0 + 1)\) is given by

\[
\alpha(t_0 + 1) = \max \{ \alpha(t_0), \frac{1}{2} [\beta^2 + L_f + Q(t_0 + 1) + G(x(t_0))]^T L_g \}
\]

Since \(\alpha(t_0) \leq \alpha^{\text{max}}\) by induction hypothesis, to prove \(\alpha(t_0 + 1) \leq \alpha^{\text{max}}\), it remains to prove

\[
\frac{1}{2} [\beta^2 + L_f + Q(t_0 + 1) + G(x(t_0))]^T L_g \leq \alpha^{\text{max}}
\]

By part (3) of Lemma 10, we have

\[
\|Q(t_0 + 1)\| \leq 2 \|\lambda^*\| + R \sqrt{2 \alpha(t_0)} + C
\]

\[
\leq 2 \|\lambda^*\| + R \sqrt{2 \alpha^{\text{max}}} + C
\]

(25)
where (a) follows the hypothesis in the induction. Thus, we have
\[
\frac{1}{2} \left[ \beta^2 + L_f + \|Q(t_0 + 1) + G(x(t_0))\|\|L_g\| \right]
\]
\[
\leq \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \frac{1}{2} \|Q(t_0 + 1) + G(x(t_0))\|\|L_g\|
\]
\[
\leq \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \frac{1}{2} \|Q(t_0 + 1)\|\|L_g\| + \|G(x(t_0))\|\|L_g\|
\]
\[
\leq \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \frac{1}{2} \|2\|\lambda^*\| + R\sqrt{2}\alpha_{\text{max}} + C\|L_g\| + \frac{1}{2} C\|L_g\|
\]
\[
= \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \|\lambda^*\|\|L_g\| + C\|L_g\| + \frac{\sqrt{2}}{2} R\|L_g\|\sqrt{\alpha_{\text{max}}}
\]
\[
\leq \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \|\lambda^*\|\|L_g\| + C\|L_g\| + \frac{\sqrt{2}}{2} R\|L_g\|\sqrt{\alpha_{\text{max}}}
\]
\[
= \frac{1}{2} \beta^2 + \frac{1}{2} L_f + \|\lambda^*\|\|L_g\| + C\|L_g\| + \frac{\sqrt{2}}{2} R\|L_g\|\sqrt{\alpha_{\text{max}}}
\]
where (a) follows from the Cauchy-Schwarz inequality; (b) follows from triangle inequality; (c) follows from (25) and \(\|G(x(t_0))\| \leq C\) by Assumption 3; (d) follows by substituting \(\alpha_{\text{max}} = \sqrt{\frac{1}{2} \beta^2 + \frac{1}{2} L_f + \|\lambda^*\|\|L_g\| + C\|L_g\| + \frac{\sqrt{2}}{2} R\|L_g\|\sqrt{\alpha_{\text{max}}}}\); and (e) follow from the basic inequality \(z_1^2 + z_2^2 + z_1 z_2 \leq (z_1 + z_2)^2\) for any \(z_1, z_2 \geq 0\).

Thus, we have \(\alpha(t_0 + 1) \leq \alpha_{\text{max}}\). This lemma follows by induction. \(\Box\)

The next theorem summarizes the \(O(1/\epsilon)\) convergence time of Algorithm 3 for convex program (1)-(3) with possibly nonlinear \(g_k(x)\).

**Theorem 2.** Consider convex program (1)-(3) under Assumptions 1-3 with possibly nonlinear \(g_k(x)\). Let \(x^*\) be an optimal solution and \(\lambda^*\) be a Lagrange multiplier vector satisfying Assumption 2. If we choose non-decreasing \(\alpha(t)\) in Algorithm 3 according to (16), then for all \(t \geq 1\), we have

1. \(F(x(t)) \leq F(x^*) + \alpha_{\text{max}} R^2\).

2. \(G_k(x(t)) \leq \frac{1}{4} \left[ \|\lambda^*\| + R\sqrt{2}\alpha_{\text{max}} + C \right] \),

where \(\alpha_{\text{max}}\) is defined in Lemma 11; and \(R\) and \(C\) are defined in Assumption 3. That is, Algorithm 3 ensures error decays like \(O(1/t)\) and provides an \(\epsilon\)-approximate solution with convergence time \(O(1/\epsilon)\).

Proof.

1. Fix \(t \geq 1\). By part (2) of Lemma 10, we have
\[
\sum_{\tau=0}^{t-1} F(x(\tau)) \leq t F(x^*) + \alpha(t - 1) R^2 + \frac{1}{2} \|G(x(t - 1))\|^2 - \frac{1}{2} \|Q(t)\|^2
\]
\[
\leq t F(x^*) + \alpha_{\text{max}} R^2
\]
where (a) follows from \(\alpha(t - 1) \leq \alpha_{\text{max}}\) by Lemma 11 and \(\|Q(t)\| \geq \|G(x(t - 1))\|\) by Lemma 5.

Dividing both sides by \(t\) and using Jensen’s inequality for convex function \(F(x)\) yields \(F(x(t)) \leq F(x^*) + \frac{\alpha_{\text{max}}}{t} R^2\).
2. Fix $t \geq 1$ and $k \in \{1, 2, \ldots, m\}$. Recall that $x(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} x(\tau)$. Thus,

$$G_k(x(t)) \leq \frac{1}{t} \sum_{\tau=0}^{t-1} G_k(x(\tau))$$

$$\leq \frac{Q_k(t)}{t}$$

$$\leq \frac{\|Q(t)\|}{t}$$

$$\leq \frac{1}{t} \left( 2\|\lambda^*\| + R\sqrt{2\alpha_{\max}} + C \right),$$

where (a) follows from the convexity of $g_k(x), k \in \{1, 2, \ldots, m\}$ and Jensen’s inequality; (b) follows from Lemma 6; and (c) follows because $\|Q(t)\| \leq 2\|\lambda^*\| + R\sqrt{2\alpha_{\max}} + C$ by part (3) of Lemma 10 and $\alpha(t-1) \leq \alpha_{\max}$ by Lemma 11.

4. Numerical Experiment: Minimum Variance Portfolio with Norm Constraints.

4.1. Minimum Variance Portfolio with the $l_2$ Norm Constraint. Consider the following constrained smooth optimization

$$\min \ x^T M x$$

s.t. $\sum_{i=1}^{n} x_i = 1$

$$\|x\|^2 \leq b$$

$$0 \leq x_i \leq 1, \forall i = \{1, 2, \ldots, n\}$$

where $x$ is the weight vector of $n$ assets and $M$ is the correlation matrix of all assets. This problem is known as global minimum variance portfolio under flexible norm constraints (GMV-N) and the $l_2$-norm constraint $\|x\|^2 \leq b$ is imposed to avoid a solution $x$ that concentrates in low volatility assets. For example, in the special case maximum decorrelation portfolio, we choose $b = 3/n$ in the $l_2$-norm constraint [10].

Without loss of optimality, we can replace the equality constraint $\sum_{i=1}^{n} x_i = 1$ with an inequality constraint $\sum_{i=1}^{n} x_i \geq 1$ in the above formulation to obtain an equivalent reformulation. This equivalent reformulation is a special case of problem (1)-(3) with $\bar{f}(x) \equiv 0$ and $\bar{g}(x) \equiv 0$. In general, for any convex programs with a linear equality constraint $h(x) = 0$, we can always replace the equality constraint with two convex inequality constraints $h(x) \leq 0$ and $h(x) \geq 0$; and reformulate the convex programs into the general form (1)-(3). In fact, if the convex program has a linear equality constraint $h(x) = 0$, we can modify the corresponding virtual queue in Algorithm 3 as $Q_k(t+1) = Q_k + h(x(t))$ at each iteration to solve it directly. (This is also a property owned by Algorithm 2 to solve convex programs with linear equality constraints, see e.g., footnote 2 in [18].)

Since $M$ is not diagonal, the objective function is not separable and hence at each iteration the update of $x(t)$ in Algorithm 2 requires to solve an $n$-dimensional

This is because if we relax $\sum_{i=1}^{n} x_i = 1$ by $\sum_{i=1}^{n} x_i \geq 1$, the optimal solution $x^*$ to the relaxed problem must satisfy $\sum_{i=1}^{n} x^*_i = 1$. 

19
set constrained quadratic program, which can have huge complexity when \( n \) is large. In contrast, the update of \( x(t) \) in Algorithm 3 has a closed form update for each coordinate by Lemma 2.

In the numerical experiment, we take \( n = 500 \), \( b = 3/n \) and generate correlation matrix \( M = [\text{Diag}(N^T N)]^{-1/2} N^T N [\text{Diag}(N^T N)]^{-1/2} \) where \( N \) is an \( n \times n \) matrix follows the standard Gaussian distribution. We run both Algorithm 2 and Algorithm 3 with the same initial point \( x(0) = 0 \). Figure 1 and Figure 2 show that both algorithms have quite similar convergence performance as observed in the zoom-in subfigures. However, when implementing both algorithms using MATLAB in a PC with a 4 core 2.7GHz Intel i7 CPU and 16GB Memory, each iteration of Algorithm 3 only takes around 1.5 milliseconds while each iteration of Algorithm 2 takes around 270 milliseconds. (Note that our implementation uses quadprog in MATLAB to solve the box constrained quadratic program involved in each iteration of Algorithm 2.) Thus, Algorithm 3 is 180 times faster than Algorithm 2 in this example.

4.2. Minimum Variance Portfolio with the \( l_1 \) Norm Constraint. Consider the following constrained non-smooth optimization

\[
\min \quad x^T M x \\
\text{s.t.} \quad \sum_{i=1}^{n} x_i = 1 \\
\quad \|x\|_1 \leq b
\]

where \( x \) is the weight vector of \( n \) assets and \( M \) is the correlation matrix of all assets. Note that each component \( x_i \in \mathbb{R} \) can be possibly negative by assuming that we can sell short the considered assets. The \( l_1 \) norm constraint \( \|x\|_1 = \sum_{i=1}^{n} |x_i| \leq b \) is
imposed to promote sparsity and other desired properties. For example, the minimum variance portfolio with the shortsale constraint considered in [8] is corresponding to the special case $\delta = 1$ in the $l_1$ norm constraint [5].

Similarly to the minimum variance portfolio with the $l_2$ norm constraint, we can replace the equality constraint $\sum_{i=1}^n x_i = 1$ with an inequality constraint $\sum_{i=1}^n x_i \geq 1$ in the above formulation to obtain an equivalent reformulation that is special case of problem (1)-(3) with $\tilde{f}(x) \equiv 0$, $g(x) = [1 - \sum_{i=1}^n x_i, 0]^T$ and $\tilde{g}(x) = [0, \sum_{i=1}^n |x_i| - b]^T$.

Since $M$ is not diagonal, the objective function is not separable and hence at each iteration the update of $x(t)$ in Algorithm 2 requires to solve an $n$-dimensional unconstrained composite minimization, which can have huge complexity when $n$ is large. In contrast, each iteration of Algorithm 3 has a closed form update for each coordinate by Lemma 1.

In the numerical experiment, we take $n = 500$, $b = 3/n$ and generate correlation matrix $M = [\text{Diag}(N^T N)]^{-1/2} N^T N [\text{Diag}(N^T N)]^{-1/2}$ where $N$ is an $n \times n$ matrix follows the standard Gaussian distribution. We run both Algorithm 2 and Algorithm 3 with the same initial point $x(0) = 0$. Figure 3 and Figure 4 show that both algorithms have quite similar convergence performance as observed in the zoom-in subfigures. However, when implementing both algorithms using MATLAB in a PC with a 4 core 2.7GHz Intel i7 CPU and 16GB memory, each iteration of Algorithm 3 only takes around 1.5 milliseconds while each iteration of Algorithm 2 takes around 2.7 seconds. (Note that our implementation uses CVX [7] to solve the unconstrained composite minimization involved in each iteration of Algorithm 2.) Thus, Algorithm 3 is 1800 times faster than Algorithm 2 in this example.

5. Conclusion. This paper proposes a new primal-dual type algorithm with $O(1/\epsilon)$ convergence for constrained composite convex programs. The new algorithm is faster than the classical primal-dual subgradient algorithm and the dual subgradient algorithm, both of which have an $O(1/\epsilon^2)$ convergence time. The new algorithm has
Fig. 3. Minimum variance portfolio with the $l_1$ norm constraint: objective value performance.

the same convergence time as that of a parallel algorithm recently proposed in [18] for convex programs with separable objective and constraint functions. However, if the objective or constraint function is not separable, the algorithm in [18] is no longer parallel and each iteration requires to solve a set constrained convex program. In contrast, the algorithm proposed in this paper is still parallel when the convex program is smooth or the non-smooth part is separable. In these cases, the new algorithm has much smaller per-iteration complexity than the algorithm in [18].

REFERENCES

[1] A. Beck, A. Ben-Tal, N. Guttmann-Beck, and L. Tetrashvili, The CoMirror algorithm for solving nonsmooth constrained convex problems, Operations Research Letters, 38 (2010), pp. 493–498.

[2] A. Beck and M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM journal on imaging sciences, 2 (2009), pp. 183–202.

[3] D. P. Bertsekas, Nonlinear Programming, Athena Scientific, second ed., 1999.

[4] J. Brodie, I. Daubechies, C. De Mol, D. Giannone, and I. Loris, Sparse and stable markowitz portfolios, Proceedings of the National Academy of Sciences, (2009).

[5] V. DeMiguel, L. Garlappi, F. J. Nogales, and R. Uppal, A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms, Management Science, 55 (2009), pp. 798–812.

[6] X. Gao, Y. Xu, and S. Zhang, Randomized primal-dual proximal block coordinate updates, arXiv:1605.05969, (2016).

[7] M. Grant and S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.0 beta, http://cvxr.com/cvx, (2013).

[8] R. Jagannathan and T. Ma, Risk reduction in large portfolios: Why imposing the wrong constraints helps, The Journal of Finance, 58 (2003), pp. 1651–1684.

[9] G. M. James, C. Paulson, and P. Rusmevichientong, The constrained LASSO, Technical report, University of Southern California, (2012).

[10] P. J. Kremer, A. Talmaciu, and S. Paterlini, Risk minimization in multi-factor portfolios:
Fig. 4. Minimum variance portfolio with the $l_1$ norm constraint: constraint value performance.

What is the best strategy?, Annals of Operations Research, 1–37 (2017).

[11] A. Nedić and A. Ozdaglar, Approximate primal solutions and rate analysis for dual subgradient methods, SIAM Journal on Optimization, 19 (2009), pp. 1757–1780.

[12] A. Nedić and A. Ozdaglar, Subgradient methods for saddle-point problems, Journal of Optimization Theory and Applications, 142 (2009), pp. 205–228.

[13] M. J. Neely, Distributed and secure computation of convex programs over a network of connected processors, in DCDIS International Conference on Engineering Applications and Computational Algorithms, Guelph, Canada, 2005.

[14] M. J. Neely, A simple convergence time analysis of drift-plus-penalty for stochastic optimization and convex programs, arXiv:1412.0791, (2014).

[15] Y. Nesterov, Introductory Lectures on Convex Optimization: A Basic Course, Springer Science & Business Media, 2004.

[16] R. Shefi and M. Teboulle, A dual method for minimizing a nonsmooth objective over one smooth inequality constraint, Mathematical Programming, (2016), pp. 137–164.

[17] H. Yu and M. J. Neely, A primal-dual type algorithm with the $O(1/t)$ convergence rate for large scale constrained convex programs, in Proceedings of IEEE Conference on Decision and Control (CDC), 2016.

[18] H. Yu and M. J. Neely, A simple parallel algorithm with an $O(1/t)$ convergence rate for general convex programs, SIAM Journal on Optimization, 27 (2017), pp. 759–783.