NEW LAGRANGIAN METHODS FOR CONSTRAINED CONVEX PROGRAMS AND THEIR APPLICATIONS

by

Hao Yu

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

May 2018

Dedication

To my parents

Acknowledgements

First and foremost, I would like to thank my advisor, Prof. Michael J. Neely. I came across Mike's Ph.D. thesis when I was an M.Phil. student at HKUST. While I was not working on the same topic, I was amazed by the beautiful and elegant drift-plus-penalty technique developed in Mike's Ph.D. work. I told myself if I were going to pursue my Ph.D., I wish I can finish with such kind of good work. Serendipitously, I later became Mike's Ph.D. student at USC. While doing research with Mike, I was impressed by his strong ability to sort out insight from complicated-looking problems. It happened so many times that when my research problems were stuck by some critical issues and seemed to come to a dead end, a seemingly random suggestion from Mike then suddenly opened up a new door for me. In my eyes, Mike is an ideal scholar and advisor. He is modest, rigorous and persistent, and always strives for perfection for each research problem. I have learned so much from him and he will continue to be my role model for my future life.

I thank Prof. Meisam Razaviyayn and Prof. Ashutosh Nayyar for sitting on both my qualifying exam and defense committees; and Prof. Lieven Vandenberghe, Prof. Bhaskar Krishnamachari and Prof. Rahul Jain for siting on my qualifying exam committee. I appreciate them for their valuable comments and suggestions.

I also thank my groupmates Xiaohan Wei and Sucha Supittayapornpong. Many of my research ideas originate from casual yet stimulating discussions with them. I also want to say "thank you" to all my other friends and colleagues on the "5th" floor at EEB.

Finally, I would like to dedicate this thesis to my parents. They always support every bold decision that I make and encourage me to pursue a higher degree. Without their unconditional love and support, this thesis would not have been possible.

Table of Contents

Abstract

In this thesis, we develop new Lagrangian methods with fast convergence for constrained convex programs with complicated functional constraints. The dual subgradient method, also known as the dual ascent method, and the primal dual subgradient method, also known as the Arrow-Hurwicz-Uzawa subgradient method, are classical Lagrangian methods to solve constrained convex programs. Both methods are known to have a slow $O(1/\epsilon^2)$ convergence time. In contrast, the new Lagrangian methods proposed in this thesis have a faster $O(1/\epsilon)$ convergence time. Recall that the alternating direction method of multipliers (ADMM), which is another representative Lagrangian method for convex programs with linear equality constraints, is also known to have $O(1/\epsilon)$ convergence. However, our methods work for general convex programs with possibly non-linear constraints.

We first revisit the classical dual subgradient method and study its convergence time for constrained strongly convex programs in Chapter 2. By using a novel drift-plus-penalty type analysis, we show that the dual subgradient method enjoys a faster $O(1/\epsilon)$ convergence time for general (possibly non-differentiable) constrained strongly convex programs. After that, we seek to develop new Lagrangian methods with the fast $O(1/\epsilon)$ convergence time for general constrained convex programs without strong convexity in Chapter 3, which is the core chapter in this thesis. Based on the new Lagrangian methods developed in Chapter 3, new techniques that exceed the state-of-the-art are developed for joint rate control and routing in data networks in Chapter 4 and for online convex optimization with stochastic and long term constraints in Chapters 5-6.

The other focus of this thesis is to illustrate the practical relevance of mathematical optimization techniques in engineering systems. In Chapter 7, we adapt our new online convex optimization technique to the power control for energy harvesting devices with outdated state information such that we can achieve utility within $O(\epsilon)$ of the optimal by using a battery with an $O(1/\epsilon)$ capacity. In Chapter 8, we extend conventional drift-plus-penalty stochastic optimization and Zinkevich's online convex optimization to develop new dynamic transmit covariance design policies for MIMO fading systems with unknown channel distributions and inaccurate channel state information. In Chapter 9, we study the index coding problem and characterize the optimality of two representative scalar and fractional linear codes, i.e., cyclic codes and maximum distance separable (MDS) codes, by studying the integrality gap between the integer linear program from an information theoretical lower bound and its linear relaxations and the Lagrangian duality between various linear relaxations and their dual problems.

Chapter 1

Introduction

A *constrained convex program*, also called *constrained convex optimization problem*, has the form:

$$\min\ f(\mathbf{x}) \tag{1.1}$$

$$\text{s.t. } g_k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \ldots, m\}, \tag{1.2}$$

$$\mathbf{x} \in \mathcal{X}, \tag{1.3}$$

where $f(\mathbf{x})$ and $g_k(\mathbf{x})$ are convex functions and $\mathcal{X}$ is a closed convex set. In general, a convex program can have linear equality constraints given by $h_j(\mathbf{x}) = 0, \forall j$, where functions $h_j(\mathbf{x})$ are linear functions. Since each linear equality constraint $h_j(\mathbf{x}) = 0$ can be equivalently represented by two convex inequality constraints $h_j(\mathbf{x}) \leq 0$ and $-h_j(\mathbf{x}) \leq 0$, linear equality constraints are not included in problem formulation (1.1)-(1.3). In this thesis, we often denote the stacked vector of multiple functions $g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})$ as $\mathbf{g}(\mathbf{x}) = \big[g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})\big]^\mathsf{T}$.

To deal with the challenge of constraints (1.2) in the above convex program, the definition of Lagrangian is introduced as follows:

Definition 1.1 (Lagrangian and Lagrangian Dual Function)**.** *For each inequality constraint $g_k(\mathbf{x}) \leq 0$, we introduce a scalar $\lambda_k \geq 0$, called a Lagrangian multiplier, and define the Lagrangian L associated with the problem* (1.1)-(1.3) *as*

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \boldsymbol{\lambda}^\mathsf{T}\mathbf{g}(\mathbf{x}) = f(\mathbf{x}) + \sum_{k=1}^{m} \lambda_k g_k(\mathbf{x}),$$

where $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]^\mathsf{T}$ is the stacked vector of all Lagrangian multipliers. Define the Lagrangian

dual function (or just dual function) associated with the problem (1.1)-(1.3) *as*

$$q(\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}} \{f(\mathbf{x}) + \sum_{k=1}^{m} \lambda_k g_k(\mathbf{x})\}$$

The Lagrangian augments the objective function $f(\mathbf{x})$ with a weighted sum of the constraint functions $g_k(\mathbf{x})$. The vector $\boldsymbol{\lambda}$ is called the *Lagrange multiplier vector* or *dual variable vector* associated with the problem (1.1)-(1.3). Correspondingly, the vector $\mathbf{x}$ is called the *primal variable vector*.

In this thesis, we assume all considered convex programs have at lease one optimal solution and the following condition holds.

Condition 1.1 (Existence of Lagrange Multipliers Attaining Strong Duality)**.** *The convex program* (1.1)-(1.3) *has Lagrange multipliers attaining the strong duality. That is, there exists a Lagrange multiplier vector* $\boldsymbol{\lambda}^* = [\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*]^\mathsf{T} \geq \mathbf{0}$ *such that*

$$q(\boldsymbol{\lambda}^*) = f(\mathbf{x}^*),$$

where $\mathbf{x}^*$ *is an optimal solution to the problem* (1.1)-(1.3) *and* $q(\boldsymbol{\lambda})$ *defined in Definition 1.1 is the dual function of the problem* (1.1)-(1.3).

Condition 1.1 is a mild condition and is implied by various conditions called constraint qualifications [BNO03, BSS06, BV04]. For example, it is implied by the existence of a vector $\mathbf{s} \in \mathcal{X}$ such that $g_k(\mathbf{s}) < 0$ for all $k \in \{1, \ldots, m\}$, called the *Slater condition*.

1.1 Lagrangian Methods for Constrained Convex Programs

Theoretically, the constrained convex program (1.1)-(1.3) can be solved directly using a first-order method with the projection onto the set $\{\mathbf{x} \in \mathcal{X} : g_k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \ldots, m\}\}$. However, such a projection itself can be computationally challenging or even as difficult as the original problem since the projection requires to minimize a quadratic function subject to the constraints (1.2) and (1.3). Alternatively, interior point methods, which convert constrained convex programs into unconstrained problems by introducing a barrier function for each functional constraint (1.2), have been developed to solve the convex program (1.1)-(1.3) [NW06, BV04]. An interior point

method typically takes a relatively small number of iterations to converge to a good solution. However, the per-iteration complexity can be huge since each iteration involves a Newton step that essentially computes Hessians and matrix inverses. In addition, interior point methods are centralized and hence are not suitable for distributed implementations in engineering systems.

Lagrangian methods, which are based on the Lagrangian defined in Definition 1.1 and strong duality described in Condition 1.1, are effective methods for constrained, especially large-scale, convex programs in the form of (1.1)-(1.3). A Lagrangian method iteratively updates both the primal variable $\mathbf{x}$ and the dual variable $\boldsymbol{\lambda}$; and the per-iteration complexity is typically low. In fact, Lagrangian methods often yield distributive implementations and hence are widely exploited in engineering applications such as data networks [KMT98, LL99, Low03], decentralized multi-agent systems [TTM11], model predictive control (MPC) [NN14] and so on.

1.1.1 Dual Subgradient Method

As a representative example of Lagrangian methods, the dual subgradient method, also known as the dual ascent method, to solve the convex program (1.1)-(1.3) is described as follows:

Algorithm 1.1 Dual Subgradient Method

Let $c > 0$ be a constant step size. Let $\boldsymbol{\lambda}(0) \geq \mathbf{0}$ be an arbitrary constant vector. At each iteration $t \in \{0, 1, 2, \ldots\}$, update $\mathbf{x}(t)$ and $\boldsymbol{\lambda}(t+1)$ as follows:

- Update primal variables via

$$\mathbf{x}(t) = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmin}} \big\{ f(\mathbf{x}) + \sum_{k=1}^{m} \lambda_k(t) g_k(\mathbf{x}) \big\}.$$

- Update dual variables via

$$\lambda_k(t+1) = \max \big\{ \lambda_k(t) + c g_k(\mathbf{x}(t)), 0 \big\}, \forall k \in \{1, 2, \ldots, m\}.$$

- Output the running average $\overline{\mathbf{x}}(t+1)$ given by

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

  as the solution at iteration $t+1$.

Note that the $\operatorname{argmin}_{\mathbf{x} \in \mathcal{X}} \{\cdot\}$ involved in the primal variable updates of Algorithm 1.1 may not be well defined even though $\mathcal{X}$ is a closed set. In general, to ensure $\operatorname{argmin}_{\mathbf{x} \in \mathcal{X}} \{h(\mathbf{x})\}$ is well defined for a continuous function $h(\mathbf{x})$, we need to exclude the possibility that $h(\mathbf{x})$ becomes

smaller as $\|\mathbf{x}\| \to \infty$. One condition that ensures the existence of $\operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\{h(\mathbf{x})\}$ is to assume $h(\mathbf{x})$ is coercive, i.e., $h(\mathbf{x}) \to \infty$ whenever $\|\mathbf{x}\| \to \infty$. This condition holds whenever $h(\mathbf{x})$ is a strongly convex function, defined in Definition 1.7. Alternatively, if the closed set $\mathcal{X}$ is bounded, then $\operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\{h(\mathbf{x})\}$ exists for any continuous function $h(\mathbf{x})$.

Let $q(\boldsymbol{\lambda})$ be the Lagrangian dual function, defined in Definition 1.1, for the convex program (1.1)-(1.3). By Danskin's Theorem (see Proposition B.25 in [Ber99]), $g(\mathbf{x}(t))$ is a subgradient of $q(\boldsymbol{\lambda})$ at the point $\boldsymbol{\lambda} = \boldsymbol{\lambda}(t)$. Recall that $\boldsymbol{\lambda} \in \mathbb{R}^m_+$. It follows that the dynamic of the dual vector $\boldsymbol{\lambda}(t)$ can be interpreted as a projected subgradient method with the constant step size c to maximize the dual function $q(\boldsymbol{\lambda})$. Thus, Algorithm 1.1 is called a dual subgradient method.

It is worthwhile emphasizing that with a constant step size c, $\boldsymbol{\lambda}(t)$ does not necessarily converge to an optimal dual vector $\boldsymbol{\lambda}^*$ that maximizes $q(\boldsymbol{\lambda})$ since $q(\boldsymbol{\lambda})$ is in general non-differentiable and the subgradient method with a constant step size may not converge to a maximizer of a non-differentiable concave function. Even if we assume $\boldsymbol{\lambda}(t) = \boldsymbol{\lambda}^*$ at certain iteration t, the corresponding $\mathbf{x}(t) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} \{f(\mathbf{x}) + \sum_{k=1}^m \lambda_k(t) g_k(\mathbf{x})\}$ is not necessarily an optimal solution to the problem (1.1)-(1.3) when the minimizer of $f(\mathbf{x}) + \sum_{k=1}^m \lambda_k(t) g_k(\mathbf{x})$ is not unique. In fact, $\mathbf{x}(t) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} \{f(\mathbf{x}) + \sum_{k=1}^m \lambda_k(t) g_k(\mathbf{x})\}$ can even be infeasible in this case. This is because an optimal solution to the problem (1.1)-(1.3) is a nontrivial convex combination of the minimizers of $f(\mathbf{x}) + \sum_{k=1}^m \lambda_k^* g_k(\mathbf{x})$ when $q(\boldsymbol{\lambda})$ is not differentiable at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$.

As a result, there is no performance guarantee of $\mathbf{x}(t)$ in Algorithm 1.1 for general convex programs in the form of (1.1)-(1.3) and it is important to use the running average $\overline{\mathbf{x}}(t)$ as the solution. The running average sequence $\overline{\mathbf{x}}(t)$ is also called the ergodic sequence in [LPS99]. The idea of using the running averages $\overline{\mathbf{x}}(t)$ as the solutions, rather than the original primal variables $\mathbf{x}(t)$, dates back to Shor [Sho85] and is further developed in [SC96, LL97, LPS99, GPS15].

If the functions $f(\mathbf{x})$ and each $g_k(\mathbf{x})$ are separable with respect to components or blocks of $\mathbf{x}$, then the updates of primal variable $\mathbf{x}(t)$ can be decomposed into several smaller independent subproblems, each of which only involves a component or block of $\mathbf{x}(t)$. For example, if $f(\mathbf{x})$ and $g_k(\mathbf{x})$ are linear functions, then the updates $\mathbf{x}(t)$ can be decomposed into n scalar convex minimizations that often have closed-form solutions. Such a desirable property has made the dual subgradient method a pervasive decomposition technique for distributed resource allocation in network utility maximization problems [LL99, PC06].

1.1.2 Primal-Dual Subgradient Method

A close relative of the dual subgraident method is the primal-dual subgradient method, also known as the Arrow-Hurwicz-Uzawa subgradient method. The convex program (1.1)-(1.3) can be solved by the primal-dual subgradient method as described in Algorithm 1.2.

Algorithm 1.2 Primal-Dual Subgradient Method

Let $c > 0$ be a constant step size. Let $\mathbf{x}(0) \in \mathcal{X}$ be arbitrary and $\boldsymbol{\lambda}(\mathbf{0}) = \mathbf{0}$. At each iteration $t \in \{1, 2, \ldots\}$, update $\mathbf{x}(t)$ and $\boldsymbol{\lambda}(t)$ as follows:

- Update primal variables via

$$\mathbf{x}(t) = \mathcal{P}_{\mathcal{X}}\big[\mathbf{x}(t-1) - c\nabla f(\mathbf{x}(t-1)) - c\sum_{k=1}^{m} \lambda_k(t-1)\nabla g_k(\mathbf{x}(t-1))\big],$$

  where $\nabla f(\mathbf{x}(t-1))$ is a subgradient of $f(\mathbf{x})$ at point $\mathbf{x} = \mathbf{x}(t-1)$, $\nabla g_k(\mathbf{x}(t-1))$ is a subgradient of $g_k(\mathbf{x})$ at point $\mathbf{x} = \mathbf{x}(t-1)$, and $\mathcal{P}_{\mathcal{X}}[\cdot]$ is the projection onto convex set $\mathcal{X}$.

- Update dual variables via

$$\lambda_k(t) = [\lambda_k(t-1) + cg_k(\mathbf{x}(t-1))]_0^{\lambda_k^{\max}}, \forall k \in \{1, 2, \ldots, m\},$$

  where $\lambda_k^{\max} > \lambda_k^*, \forall k \in \{1, 2, \ldots, m\}$ with λ_k^* defined in Condition 1.1 and $[\cdot]_0^{\lambda_k^{\max}}$ is the projection onto interval $[0, \lambda_k^{\max}]$.

- Output the running average $\overline{\mathbf{x}}(t+1)$ given by

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

  as the solution at iteration $t+1$.

Recall that if the strong duality for the problem (1.1)-(1.3) is attained by certain Lagrange multipliers, i.e., Condition 1.1 holds, then by the saddle point theorem for convex programs (Proposition 5.1.6 in [Ber99]), $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$ is an optimal solution-multiplier pair if and only if it is a saddle point of the Lagrangian. Let $L(\mathbf{x}, \boldsymbol{\lambda})$ be the Lagrangian, defined in Definition 1.1, for the convex program (1.1)-(1.3). It follows that $\nabla f(\mathbf{x}(t-1)) + \sum_{k=1}^{m} \lambda_k(t-1)\nabla g_k(\mathbf{x}(t-1)) \in \frac{\partial}{\partial \mathbf{x}} L(\mathbf{x}(t-1), \boldsymbol{\lambda}(t-1))$ and $\mathbf{g}(\mathbf{x}(t-1)) \in \frac{\partial}{\partial \boldsymbol{\lambda}} L(\mathbf{x}(t-1), \boldsymbol{\lambda}(t-1))$, and hence Algorithm 1.2 can be interpreted as an Arrow-Hurwicz-Uzawa algorithm for solving the saddle points of the Lagragian $L(\mathbf{x}, \boldsymbol{\lambda})$.

Note that if $f(\mathbf{x})$ or $g_k(\mathbf{x})$ are separable, then the primal updates in Algorithm 1.1 requires to solve unconstrained convex minimization problems, which can incur huge complexity when the

number of constraints or the primal variable dimension is large. In contrast, Algorithm 1.2 iterates the primal variables using projected gradient updates, which can be performed independently for each component of $\mathbf{x}$ as long as the set $\mathcal{X}$ is a Cartesian product. Note that if $\mathcal{X}$ is a complicated set where coordinates are coupled together, we can introduce more inequality constraints and leave $\mathcal{X}$ a simple set. This property makes Algorithm 1.2 suitable for large scale convex programs with non-separable objective or constraint functions.

On the other hand, Algorithm 1.2 requires the knowledge of upper bounds for the optimal Lagrange multipliers to determine algorithm parameters $\lambda_k^{\max}, \forall k \in \{1, 2, \ldots, m\}$ used in the dual updates. These upper bounds can be difficult to estimate for some convex programs.

1.1.3 Drift-Plus-Penalty Technique for Deterministic Optimization

The drift-plus-penalty technique was originally developed to solve more general stochastic optimization [Nee03, GNT06, Nee10] and was shown applicable to deterministic convex programs [Nee05, Nee14]. The drift-plus-penalty technique originates from the *backpressure algorithm* considered in the seminal work [TE92] by Tassiulas and Ephremides. The backpressure algorithm developed in [TE92] is to perform routing and scheduling by minimizing a Lyapunov drift expression such that all data queues in the stochastic data network are stabilized whenever possible. However, such a backpressure algorithm has no utility performance guarantee when network utilities exist in the considered network. The drift-plus-penalty technique extends the method in [TE92] by introducing an additional penalty term, corresponding to the network utility, to the drift minimization such that the problem of joint network stability and utility maximization can be solved. Later, this technique is further extended to solve general stochastic optimization, not only stochastic optimization in queueing networks, by introducing virtual queues for general stochastic constraints, not only queue stability constraints.

A deterministic convex program in the form of (1.1)-(1.3) can be solved by the drift-plus-penalty technique as described in Algorithm 1.3. Note that the drift-plus-penalty technique introduces a virtual queue $Q_k(t)$ for each functional constraint (1.2). These virtual queues can be interpreted as the queue backlogs of constraint violations and indeed correspond to physical queue backlogs when each constraint (1.2) is a node flow balance constraint in data network applications. It was noted in [NMR05, HN11, SHN14] that the drift-plus-penalty technique applied to deterministic convex programs is closely related to the dual subgradient method. In

fact, if we let $V = \frac{1}{c}$ and $\frac{Q_k(0)}{V} = \lambda_k(0), \forall k \in \{1, 2, \ldots, m\}$, then Algorithm 1.3 and Algorithm 1.1 yield the same sequence of $\mathbf{x}(t)$ and $\frac{Q_k(t)}{V} = \lambda_k(t), \forall k \in \{1, 2, \ldots, m\}, \forall t \geq 0$.

Algorithm 1.3 Drift-Plus-Penalty (DPP) Algorithm

Let $V > 0$ be a constant parameter. Let $\mathbf{Q}(0) \geq 0$ be arbitrary. At each iteration $t \in \{0, 1, 2, \ldots\}$, update $\mathbf{x}(t)$ and $\mathbf{Q}(t+1)$ as follows:

- Update primal variables via

$$\mathbf{x}(t) = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmin}} \big\{ V f(\mathbf{x}) + \sum_{k=1}^{m} Q_k(t) g_k(\mathbf{x}) \big\}.$$

- Update virtual queues via

$$Q_k(t+1) = \max \big\{ Q_k(t) + g_k(\mathbf{x}(t)), 0 \big\}, \forall k \in \{1, 2, \ldots, m\}.$$

- Output the running average $\overline{\mathbf{x}}(t+1)$ given by

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

  as the solution at iteration $t+1$.

However, we emphasize that the drift-plus-penalty technique can solve stochastic optimization that is more general than deterministic convex programs. In addition, the performance analysis of the drift-plus-penalty is based on Lyapunov type analysis of a drift-plus-penalty expression and is fundamentally different from the conventional analysis of the dual subgradient method. In fact, the new Lagrangian methods developed in this thesis inherits heavily from the drift-plus-penalty technique even though many of them are intended to be developed to solve deterministic convex programs.

1.1.4 Alternating Direction Method of Multipliers (ADMM)

Now consider a special case of the convex program (1.1)-(1.3) with separable objective functions and linear equality constraints, given as follows:

$$\min \ f_1(\mathbf{x}) + f_2(\mathbf{y}) \tag{1.4}$$

$$\text{s.t. } \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{c}, \tag{1.5}$$

$$\mathbf{x} \in \mathcal{X} \subseteq \mathcal{R}^{n_1}, \mathbf{y} \in \mathcal{Y} \subseteq \mathcal{R}^{n_2}. \tag{1.6}$$

Note that a linear inequality constraint can be converted to an equality constraint by introducing a non-negative dummy variable. By augmenting the Lagrangian with a quadratic term of the equality constraint function, we define the augmented Lagrangian as

$$L_\rho(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}) = f_1(\mathbf{x}) + f_2(\mathbf{y}) + \boldsymbol{\lambda}^\mathsf{T}\big(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{c}\big) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{c}\|^2$$

where ρ is a constant algorithm parameter. In the ADMM applied to solve the problem (1.4)-(1.6), each iteration consists of the following steps:

- Update $\mathbf{x}(t) = \text{argmin}_{\mathbf{x}\in\mathcal{X}}\, L_\rho(\mathbf{x}, \mathbf{y}(t-1), \boldsymbol{\lambda}(t-1))$.
- Update $\mathbf{y}(t) = \text{argmin}_{\mathbf{y}\in\mathcal{Y}}\, L_\rho(\mathbf{x}(t), \mathbf{y}, \boldsymbol{\lambda}(t-1))$.
- Update $\boldsymbol{\lambda}(t) = \boldsymbol{\lambda}(t-1) + \rho\big[\mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{y}(t) - \mathbf{c}\big]$.

At each iteration, the ADMM updates primal variables $\mathbf{x}$ and $\mathbf{y}$ in an alternating manner, also known as Gauss-Seidel manner, which accounts for the name "alternating direction method". The isolated update of variables $\mathbf{x}$ and $\mathbf{y}$ can reduce per-iteration complexity in comparison to jointly choosing $(\mathbf{x}, \mathbf{y})$ to minimize $L_\rho(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda})$. The ADMM recently has attracted a lot interest in machine learning, network scheduling, computational biology and finance. See [BPC+11] for a recent survey on the development and applications of ADMM.

However, a significant limitation of the ADMM algorithm is that it can only solve problems with linear constraints. This is mainly because the quadratic term introduced in the augmented Lagrangian can be non-convex when there are non-linear constraints.

1.2 Convergence Time of Existing Lagrangian Methods

For the convex program (1.1)-(1.3), we define an ϵ-approximate solution as follows.

Definition 1.2 (ϵ-approximate Solution)**.** *Let $\mathbf{x}^*$ be an optimal solution to the problem* (1.1)-(1.3)*. For any $\epsilon > 0$, a point $\mathbf{x}^\epsilon \in \mathcal{X}$ is said to be an ϵ-approximate solution if $f(\mathbf{x}^\epsilon) \leq f(\mathbf{x}^*) + \epsilon$ and $g_k(\mathbf{x}^\epsilon) \leq \epsilon, \forall k \in \{1, \ldots, m\}$.*

Note that if $\mathbf{x}^\epsilon$ is an ϵ-approximate solution and there exists $\mathbf{z} \in \mathcal{X}$ such that $g_k(\mathbf{z}) \leq -\delta, \forall k \in \{1, \ldots, m\}$ for some $\delta > 0$, one can convert an ϵ-approximate point $\mathbf{x}^\epsilon$ to another point

$\mathbf{x} = \theta\mathbf{x}^{\epsilon} + (1-\theta)\mathbf{z}$, for $\theta = \frac{\delta}{\epsilon+\delta}$, which satisfies all constraints $g_k(\mathbf{x}) \leq 0$ and has an objective value within $O(\epsilon)$ of optimality.

The convergence time of an iterative algorithm measures the number of iterations required to obtain an ϵ-approximate solution and is formally defined as follows:

Definition 1.3 (Convergence Time)**.** *Let $\mathbf{x}(t), t \in \{1, 2, \ldots\}$ be the solution sequence yielded by an iterative algorithm. The convergence time (to an ϵ approximate solution) of this algorithm is the number of iterations required to achieve an ϵ-approximate solution. That is, this algorithm is said to have an $O(h(\epsilon))$ convergence time if $\{\mathbf{x}(t), t \geq O(h(\epsilon))\}$ is a sequence of ϵ-approximate solutions.*

Or alternatively, we have the definition of convergence rates given as follows:

Definition 1.4 (Convergence Rate)**.** *Let $\tilde{h}(t)$ be a decreasing function converging to 0 as $t \to \infty$; and $\mathbf{x}(t), t \in \{1, 2, \ldots\}$ be the solution sequence yielded by an iterative algorithm. This algorithm is said to have an $O(\tilde{h}(t))$ convergence rate if $f(\mathbf{x}(t)) \leq f(\mathbf{x}^*) + \tilde{h}(t)$ and $g_k(\mathbf{x}(t)) \leq \tilde{h}(t), \forall k \in \{1, \ldots, m\}$ for all $t \geq 1$.*

The definition of convergence rate is independent of ϵ and requires the error to eventually converge to zero. In contrast, the definition of convergence time depends on ϵ and only requires that the solution error is eventually smaller than ϵ. In this sense, the definition of convergence rate is slightly stronger than the definition of convergence time; and a convergence rate result can imply the convergence time result. For example, if a solution sequence $\mathbf{x}(t)$ satisfies $f(\mathbf{x}(t)) \leq f(\mathbf{x}^*) + \frac{1}{\sqrt{t}}$ and $g_k(\mathbf{x}(t)) \leq \frac{1}{\sqrt{t}}, \forall k \in \{1, \ldots, m\}$ for all $t \geq 1$, then this algorithm has an $O(\frac{1}{\sqrt{t}})$ convergence rate (since error decays with time like $O(\frac{1}{\sqrt{t}})$) and this further implies that the convergence time of this algorithm is $O(\frac{1}{\epsilon^2})$. However, the definition of convergence time is still quite useful in analyzing the convergence performance of an iterative algorithm since some algorithms fundamentally do not have vanishing errors. In this thesis, we use terminologies "convergence rate" and "convergence time" interchangeably when proper.

The convergence time results of existing Lagrangian methods are summarized as follows:

- **Dual subgradient methods and drift-plus-penalty technique:** For general convex programs in the form of (1.1)-(1.3), where the objective function $f(\mathbf{x})$ is convex but not necessarily strongly convex, the convergence time of Algorithm 1.3 is shown to be $O(\frac{1}{\epsilon^2})$ in [Nee05, Nee14]. A similar $O(\frac{1}{\epsilon^2})$ convergence time of Algorithm 1.1 is shown in [NO09a].

A recent work [SHN14] shows that Algorithm 1.3 has an $O(\frac{1}{\epsilon})$ convergence time if the dual function is locally polyhedral and Algorithm 1.3 with a different average scheme has an $O(\frac{1}{\epsilon^{1.5}})$ convergence time if the dual function is locally quadratic. For a special class of strongly convex programs in the form of (1.1)-(1.3), where $f(\mathbf{x})$ is second-order differentiable and strongly convex and $g_k(\mathbf{x}), \forall k \in \{1, 2, \ldots, m\}$ are second-order differentiable and have bounded Jacobians, the convergence time of Algorithm 1.1 is shown to be $O(\frac{1}{\epsilon})$ in [NN14].

- **Primal-dual subgradient methods:** The convergence time of Algorithm 1.2 is proven to be $O(\frac{1}{\epsilon^2})$ in [NO09b].

- **ADMM:** The best known convergence time of ADMM algorithm for the convex program (1.4)-(1.6) with general convex $f_1(\cdot)$ and $f_2(\cdot)$ is recently shown to be $O(\frac{1}{\epsilon})$ [HY12, LMZ15]. An asynchronous ADMM algorithm with the same $O(\frac{1}{\epsilon})$ convergence time is studied in [WO13]. Geometric convergence rate of ADMM is possible under restrictive assumptions such as the strong convexity of the objective functions and rank conditions of the linear equality constraints [HL16, DY16].

1.3 Facts From Convex Analysis

In this section, we introduce basic facts from convex analysis that will be frequently used throughout this thesis. If not specified, we always use $\|\cdot\|$ to denote the Euclidean norm, also known as l_2 norm, of a vector.

Definition 1.5 (Lipschitz Continuity)**.** *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. Function $h : \mathcal{X} \mapsto \mathbb{R}^m$ is said to be Lipschitz continuous on $\mathcal{X}$ with modulus L if there exists $L > 0$ such that*

$$\|h(\mathbf{y}) - h(\mathbf{x})\| \leq L\|\mathbf{y} - \mathbf{x}\|, \forall \mathbf{x}, \mathbf{y} \in \mathcal{X}.$$

Definition 1.6 (Smooth Functions)**.** *Let $\mathcal{X} \subseteq \mathbb{R}^n$ and function $h(\mathbf{x})$ be continuously differentiable on $\mathcal{X}$. Function $h(\mathbf{x})$ is said to be smooth on $\mathcal{X}$ with modulus L if $\nabla h(\mathbf{x})$ is Lipschitz continuous on $\mathcal{X}$ with modulus L.*

Note that linear function $h(\mathbf{x}) = \mathbf{a}^\mathsf{T}\mathbf{x}$ is smooth with modulus 0. If a function $h(\mathbf{x})$ is smooth with modulus L, then $ch(\mathbf{x})$ is smooth with modulus cL for any $c > 0$.

Lemma 1.1 (Descent Lemma, Proposition A.24 in [Ber99]). *Let h be a continuously differentiable function. If h is smooth on $\mathcal{X}$ with modulus L, then for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ we have*

$$h(\mathbf{y}) \geq h(\mathbf{x}) + [\nabla h(\mathbf{x})]^{\mathsf{T}}(\mathbf{y} - \mathbf{x}) - \frac{L}{2}||\mathbf{y} - \mathbf{x}||^2,$$
$$h(\mathbf{y}) \leq h(\mathbf{x}) + [\nabla h(\mathbf{x})]^{\mathsf{T}}(\mathbf{y} - \mathbf{x}) + \frac{L}{2}||\mathbf{y} - \mathbf{x}||^2.$$

Definition 1.7 (Strongly Convex Functions). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. Function $h(\mathbf{x})$ is said to be strongly convex on $\mathcal{X}$ with modulus α if there exists a constant $\alpha > 0$ such that $h(\mathbf{x}) - \frac{1}{2}\alpha\|\mathbf{x}\|^2$ is convex on $\mathcal{X}$.*

Recall that function $h(\mathbf{x})$ is concave if and only if $-h(\mathbf{x})$ is convex. Similarly, function $h(\mathbf{x})$ is strongly concave with modulus α if and only if $-h(\mathbf{x})$ is strongly convex with modulus α. Or alternatively, function $h(\mathbf{x})$ is strongly concave with if there exists a constant $\alpha > 0$ such that $h(\mathbf{x}) + \frac{1}{2}\alpha\|\mathbf{x}\|^2$ is concave.

The next corollary follows directly from the definition of strongly convex functions.

Corollary 1.1. *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. If function $h(\mathbf{x})$ is convex on $\mathcal{X}$ and $\alpha > 0$, then $h(\mathbf{x}) + \alpha\|\mathbf{x} - \mathbf{x}_0\|^2$ is strongly convex with modulus 2α for any constant $\mathbf{x}_0 \in \mathbb{R}^n$.*

Lemma 1.2 (Theorem D.6.1.2 in [HUL01]). *Let function $h(\mathbf{x})$ be strongly convex on $\mathcal{X}$ with modulus α. Let $\partial h(\mathbf{x})$ be the set of all subgradients of h at point $\mathbf{x}$. Then*

$$h(\mathbf{y}) \geq h(\mathbf{x}) + \mathbf{d}^{\mathsf{T}}(\mathbf{y} - \mathbf{x}) + \frac{\alpha}{2}\|\mathbf{y} - \mathbf{x}\|^2, \forall \mathbf{x}, \mathbf{y} \in \mathcal{X}, \forall \mathbf{d} \in \partial h(\mathbf{x}).$$

Lemma 1.3 (Proposition B.24 (f) in [Ber99]). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. Let function $h(\mathbf{x})$ be convex on $\mathcal{X}$ and $\mathbf{x}^{opt}$ be a global minimum of h on $\mathcal{X}$. Let $\partial h(\mathbf{x})$ be the set of all subgradients of h at point $\mathbf{x}$. Then, there exists $\mathbf{d} \in \partial h(\mathbf{x}^{opt})$ such that*

$$\mathbf{d}^{\mathsf{T}}(\mathbf{x} - \mathbf{x}^{opt}) \geq 0, \forall \mathbf{x} \in \mathcal{X}.$$

Corollary 1.2. *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. Let function $h(\mathbf{x})$ be strongly convex on $\mathcal{X}$ with modulus α and $\mathbf{x}^{opt}$ be a global minimum of h on $\mathcal{X}$. Then,*

$$h(\mathbf{x}^{opt}) \leq h(\mathbf{x}) - \frac{\alpha}{2}\|\mathbf{x}^{opt} - \mathbf{x}\|^2, \forall \mathbf{x} \in \mathcal{X}.$$

Proof. A special case when h is differentiable and $\mathcal{X} = \mathbb{R}^n$ is Theorem 2.1.8 in [Nes04]. The proof for general h and $\mathcal{X}$ is as follows.

Fix $\mathbf{x} \in \mathcal{X}$. By Lemma 1.3, there exists $\mathbf{d} \in \partial h(\mathbf{x}^{opt})$ such that $\mathbf{d}^{\mathsf{T}}(\mathbf{x} - \mathbf{x}^{opt}) \geq 0$. By Lemma 1.2, we also have

$$\begin{aligned} h(\mathbf{x}) &\geq h(\mathbf{x}^{opt}) + \mathbf{d}^{\mathsf{T}}(\mathbf{x} - \mathbf{x}^{opt}) + \frac{\alpha}{2}\|\mathbf{x} - \mathbf{x}^{opt}\|^2 \\ &\overset{(a)}{\geq} h(\mathbf{x}^{opt}) + \frac{\alpha}{2}\|\mathbf{x} - \mathbf{x}^{opt}\|^2, \end{aligned}$$

where (a) follows from the fact that $\mathbf{d}^{\mathsf{T}}(\mathbf{x} - \mathbf{x}^{opt}) \geq 0$. □

Similarly, we have the next corollary for strongly concave functions.

Corollary 1.3. *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. Let function $h(\mathbf{x})$ be strongly concave on $\mathcal{X}$ with modulus α and $\mathbf{x}^{opt}$ be a global maximum of h on $\mathcal{X}$. Then,*

$$h(\mathbf{x}^{opt}) \geq h(\mathbf{x}) + \frac{\alpha}{2}\|\mathbf{x}^{opt} - \mathbf{x}\|^2, \forall \mathbf{x} \in \mathcal{X}.$$

1.4 Thesis Outline and Our Contributions

This thesis is organized as follows:

- **Chapter 2 – Convergence time of dual subgradient methods for strongly convex programs:** This chapter considers general strongly convex programs (possibly non-differentiable) and shows that the classical dual subgraident method with simple running averages has $O(\frac{1}{\epsilon})$ convergence. This chapter also shows that if the strongly convex program satisfies additional assumptions, then the dual subgradient method with a new running average scheme, called the sliding running average, can achieve $O(\log(\frac{1}{\epsilon}))$ convergence.

- **Chapter 3 – New Lagrangian methods for convex programs:**

  This chapter considers constrained convex programs (possibly without strong convexity) and develops two new Lagrangian methods with fast $O(\frac{1}{\epsilon})$ convergence. The new methods improve the conventional dual subgradient method or primal-dual subgradient method, both of which are known to have slow $O(\frac{1}{\epsilon^2})$ convergence. The new methods can deal with nonlinear convex inequality constraints that can not be handled by the alternating direction

method of multipliers (ADMM). The first new Lagrangian method is a dual type method and works for general constrained convex programs (with possibly non-differentiable objective and constraint functions). The second one is a primal-dual type method but only works for smooth constrained convex programs. Both methods have the same $O(1/\epsilon)$ convergence time, yet the second one enjoys smaller per-iteration complexity when the objective function or the constraint functions are not separable.

- **Chapter 4 – New backpressure algorithms for joint rate control and routing:**

  This chapter considers backpressure algorithms for joint rate control and routing in multi-hop data networks. To achieve an arbitrary small utility optimality gap, all existing backpressure algorithms necessarily yield arbitrarily large queue lengths. Inspired by the new Lagrangian dual optimization methods developed in Chapter 3, this chapter proposes new backpressure algorithms that can converge to the exact optimal utility while ensuring all queue lengths are bounded by a finite constant.

- **Chapter 5 – Online convex optimization with stochastic constraints:** This chapter considers online convex optimization (OCO) with stochastic constraints, which generalizes Zinkevich's OCO over a known simple fixed set by introducing multiple stochastic functional constraints that are i.i.d. generated at each round and are disclosed to the decision maker only after the decision is made. This formulation arises naturally when decisions are restricted by stochastic environments or deterministic environments with noisy observations. To solve this problem, this chapter proposes a new algorithm that achieves $O(\sqrt{T})$ expected regret and constraint violations and $O(\sqrt{T}\log(T))$ high probability regret and constraint violations.

- **Chapter 6 – Online convex optimization with long term constraints:** This chapter considers online convex optimization with long term constraints, which is a special case problem of online convex optimization with stochastic constraints considered in Chapter 5. In online convex optimization with long term constraints, we relax the functional constraints by allowing them to be violated at each round but still requiring them to be satisfied in the long term. Inspired by the Lagrangian dual optimization technique developed in Chapter 3, this chapter develops a new online learning algorithm that can achieve $O(\sqrt{T})$ regret and $O(1)$, i.e., finite, constraint violations.

- **Chapter 7 – Power control for energy harvesting devices with outdated state information:** This chapter considers utility optimal power control for energy harvesting wireless devices with a finite capacity battery. The distribution information of the underlying wireless environment and harvestable energy is unknown and only outdated system state information is known at the device controller. This chapter proposes a new online power control algorithm that achieves utility within $O(\epsilon)$ of the optimal, for any desired $\epsilon > 0$, by using a battery with an $O(1/\epsilon)$ capacity.

- **Chapter 8 – Dynamic transmit covariance design in MIMO fading systems with unknown channel distributions and inaccurate channel state information:**

  This chapter considers dynamic transmit covariance design in point-to-point MIMO fading systems with unknown channel state distributions and inaccurate channel state information subject to both long term and short term power constraints. We develop different dynamic transmit covariance policies for the case of instantaneous channel state information at the transmitter (CSIT) and the case of delayed CSIT, respectively. In either case, the corresponding policy can approach optimality with an $O(\delta)$ gap, where δ is the inaccuracy measure of CSIT.

- **Chapter 9 – Duality codes and the integrality gap bound for index coding:**

  This chapter considers the index coding problem that captures the essence of the classical network coding problem. By studying the integrality gap of an integer linear program, which arises from an information theoretical bound for the index coding problem, and the Lagrangian duality of its linear programing relaxations, we analyze the performance of cyclic codes and partial clique codes for index coding. We also show these codes are optimal when the bipartite digraph representation of the index coding problem is planar.

Chapter 2

Convergence Time of Dual Subgradient Methods for Strongly Convex Programs

Consider the following strongly convex program:

$$\min\ f(\mathbf{x}) \tag{2.1}$$

$$\text{s.t. } g_k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \ldots, m\} \tag{2.2}$$

$$\mathbf{x} \in \mathcal{X} \tag{2.3}$$

where the set $\mathcal{X} \subseteq R^n$ is closed and convex; function $f(\mathbf{x})$ is continuous and strongly convex on $\mathcal{X}$; functions $g_k(\mathbf{x}), \forall k \in \{1, 2, \ldots, m\}$ are Lipschitz continuous and convex on $\mathcal{X}$. Note that the functions $f(\mathbf{x}), g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})$ are not necessarily differentiable. Denote the stacked vector of multiple functions $g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})$ as $\mathbf{g}(\mathbf{x}) = \big[g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})\big]^{\mathsf{T}}$. Throughout this chapter, the strongly convex program (2.1)-(2.3) is required to satisfy the following assumptions:

Assumption 2.1 (Basic Assumptions)**.**

- *There exists an optimal solution* $\mathbf{x}^* \in \mathcal{X}$ *that solves the strongly convex program* (2.1)-(2.3).
- *The function* $f(\mathbf{x})$ *is strongly convex on* $\mathcal{X}$ *with modulus* σ.
- *There exists a constant* β *such that* $\|\mathbf{g}(\mathbf{x}_1) - \mathbf{g}(\mathbf{x}_2)\| \leq \beta\|\mathbf{x}_1 - \mathbf{x}_2\|$ *for all* $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$. *That is, the function* $\mathbf{g}(\mathbf{x})$ *is Lipschitz continuous on* $\mathcal{X}$ *with modulus* β.

Note that the strong convexity of the function $f(\mathbf{x})$ implies that the optimum is unique.

Assumption 2.2 (Existence of Lagrange Multipliers)**.** *Condition 1.1 holds for the strongly convex program* (2.1)-(2.3)*. That is, there exists a Lagrange multiplier vector* $\boldsymbol{\lambda}^* = [\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*]^\mathsf{T} \geq \mathbf{0}$ *such that*

$$q(\boldsymbol{\lambda}^*) = f(\mathbf{x}^*),$$

where $\mathbf{x}^*$ *is the optimal solution to the problem*(2.1)-(2.3) *and* $q(\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}} \{f(\mathbf{x}) + \sum_{k=1}^m \lambda_k g_k(\mathbf{x})\}$ *is the Lagrangian dual function of the problem* (2.1)-(2.3)*.*

The strongly convex program (2.1)-(2.3) arises often in applications such as model predictive control (MPC) [NN14], network flow control [LL99] and decentralized multi-agent control [TTM11]. Algorithm 1.1, the dual subgradient method, is a conventional method to solve (2.1)-(2.3) [BSS06]. It is an iterative algorithm that, every iteration, removes the inequality constraints (2.2) and chooses primal variables to minimize a function over the set $\mathcal{X}$. Algorithm 1.1 can be interpreted as a subgradient/gradient method applied to the Lagrangian dual function of convex program (2.1)-(2.3) and allows for many different step size rules [Ber99]. Note that by Danskin's theorem (Proposition B.25(a) in [BSS06]), the Lagrangian dual function of a strongly convex program is differentiable, thus Algorithm 1.1 for strongly convex program (2.1)-(2.3) is in fact a dual gradient method.

The classical dual subgradient method with a constant step size, Algorithm 1.1, uses the **simple running averages**, also called the ergodic sequence in [LPS99], $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbf{x}(\tau)$ as the solutions at each iteration. In this chapter, we also proposes a new running average scheme, called the sliding running averages as follows:

- **Sliding Running Averages:** Use $\widetilde{\mathbf{x}}(t) = \mathbf{x}(0)$ and

$$\widetilde{\mathbf{x}}(t) = \begin{cases} \frac{1}{\frac{t}{2}} \sum_{\tau=\frac{t}{2}}^{t-1} \mathbf{x}(\tau) & \text{if } t \text{ is even} \\ \widetilde{\mathbf{x}}(t-1) & \text{if } t \text{ is odd} \end{cases}$$

  as the solution at each iteration $t \in \{1, 2, \ldots\}$.

This chapter shows that the sliding running averages can have better convergence time when the dual function of the convex program satisfies additional assumptions. The results in this chapter are originally developed in our papers [YN15, YN18b].

2.1 Related Work

As reviewed in Section 1.2, the convergence time of Algorithm 1.1 for general (possibly without strong convexity) convex programs is $O(\frac{1}{\epsilon^2})$. For a special class of strongly convex programs in the form of (2.1)-(2.3), where $f(\mathbf{x})$ is second-order differentiable and strongly convex and $g_k(\mathbf{x}), \forall k \in \{1, 2, \ldots, m\}$ are second-order differentiable and have bounded Jacobians, the convergence time of the dual subgradient algorithm is shown to be $O(\frac{1}{\epsilon})$ in [NN14]. Note that convex program (2.1)-(2.3) with second order differentiable $f(\mathbf{x})$ and $g_k(\mathbf{x}), k \in \{1, 2, \ldots, m\}$ in general can be solved via interior point methods with geometric convergence. However, to achieve fast convergence in practice, the barrier parameters must be scaled carefully and the computation complexity associated with each iteration can be high. In contrast, the dual subgradient method can yield distributive implementations with low per-iteration computation complexity when the objective and constraint functions are separable.

This chapter considers a class of strongly convex programs that is more general than those treated in [NN14].[1] Besides the strong convexity of $f(\mathbf{x})$, we only require the constraint functions $g_k(\mathbf{x})$ to be Lipschitz continuous. The functions $f(\mathbf{x})$ and $g_k(\mathbf{x})$ can even be non-differentiable. Thus, this paper can deal with non-smooth optimization. For example, the l_1 norm $\|\mathbf{x}\|_1$ is non-differentiable and often appears as part of the objective or constraint functions in machine learning, compressed sensing and image processing applications. This chapter shows that the convergence time of the dual subgradient method with simple running averages for general strongly convex programs is $O(\frac{1}{\epsilon})$ and the convergence time can be improved to $O(\log(\frac{1}{\epsilon}))$ by using sliding running averages when the dual function is locally quadratic.

A closely related recent work is [NP16] that considers strongly convex programs with strongly convex and second order differentiable objective functions $f(\mathbf{x})$ and conic constraints in the form of $\mathbf{G}\mathbf{x} + h \in \mathcal{K}$, where $\mathcal{K}$ is a proper cone. The authors in [NP16] show that a hybrid algorithm using both dual subgradient and dual fast gradient methods can have convergence time $O(\frac{1}{\epsilon^{2/3}})$; and the dual subgradient method can have convergence time $O(\log(\frac{1}{\epsilon}))$ if the strongly convex program satisfies an error bound property. Results in this chapter are developed independently and consider general nonlinear convex constraint functions; and show that the dual subgradi-

[1] Note that bounded Jacobians imply Lipschitz continuity. Work [NN14] also considers the effect of inaccurate solutions for the primal updates. The analysis in this chapter can also deal with inaccurate updates. In this case, there will be an error term δ on the right of (2.6).

ent/gradient method with a different averaging scheme has an $O(\log(\frac{1}{\epsilon}))$ convergence time when the dual function is locally quadratic. Another independent parallel work is [NPN15] that considers strongly convex programs with strongly convex and smooth objective functions $f(\mathbf{x})$ and general constraint functions $\mathbf{g}(\mathbf{x})$ with bounded Jacobians. The authors in [NPN15] shows that the dual subgradient/gradient method with simple running averages has $O(\frac{1}{\epsilon})$ convergence.

This chapter and independent parallel works [NP16, NPN15] obtain similar convergence times of the dual subgradient/gradient method with different averaging schemes for strongly convex programs under slightly different assumptions. However, the proof technique in this chapter is fundamentally different from that used in [NP16] and [NPN15]. Works [NP16, NPN15] and other previous works, e.g., [NN14], follow the classical optimization analysis approach based on the descent lemma, while this chapter is based on the drift-plus-penalty analysis that was originally developed for stochastic optimization in dynamic queuing systems [Nee03, Nee10]. Using the drift-plus-penalty technique, we further propose a new Lagrangian dual type algorithm with $O(\frac{1}{\epsilon})$ convergence for general convex programs (possibly without strong convexity) in Chapter 3.

2.2 Convergence Time Analysis

This section analyzes the convergence time of Algorithm 1.1 for the strongly convex program (2.1)-(2.3) under Assumptions 2.1-2.2.

2.2.1 An Upper Bound of the Drift-Plus-Penalty Expression

Denote $\boldsymbol{\lambda}(t) = \big[\lambda_1(t), \ldots, \lambda_m(t)\big]^{\mathsf{T}}$. Define Lyapunov function $L(t) = \frac{1}{2}\|\boldsymbol{\lambda}(t)\|^2$ and Lyapunov drift $\Delta(t) = L(t+1) - L(t)$.

Lemma 2.1. *At each iteration t in Algorithm 1.1, we have*

$$\frac{1}{c}\Delta(t) = [\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \tag{2.4}$$

Proof. The update equations $\lambda_k(t+1) = \max\{\lambda_k(t) + cg_k(\mathbf{x}(t)), 0\}, \forall k \in \{1, 2, \ldots, m\}$ can be rewritten as

$$\lambda_k(t+1) = \lambda_k(t) + c\tilde{g}_k(\mathbf{x}(t)), \forall k \in \{1, 2, \ldots, m\}, \tag{2.5}$$

where $\tilde{g}_k(\mathbf{x}(t)) = \begin{cases} g_k(\mathbf{x}(t)), & \text{if } \lambda_k(t) + cg_k(\mathbf{x}(t)) \geq 0 \\ -\frac{1}{c}\lambda_k(t), & \text{else} \end{cases}, \forall k \in \{1, 2, \ldots, m\}$.

Fix $k \in \{1, 2, \ldots, m\}$. Squaring both sides of (2.5) and dividing by factor 2 yields:

$$\begin{aligned}
&\frac{1}{2}[\lambda_k(t+1)]^2 \\
=&\frac{1}{2}[\lambda_k(t)]^2 + \frac{c^2}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + c\lambda_k(t)\tilde{g}_k(\mathbf{x}(t)) \\
=&\frac{1}{2}[\lambda_k(t)]^2 + \frac{c^2}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + c\lambda_k(t)g_k(\mathbf{x}(t)) + c\lambda_k(t)[\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))] \\
\overset{(a)}{=}&\frac{1}{2}[\lambda_k(t)]^2 + \frac{c^2}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + c\lambda_k(t)g_k(\mathbf{x}(t)) - c^2\tilde{g}_k(\mathbf{x}(t))[\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))] \\
=&\frac{1}{2}[\lambda_k(t)]^2 - \frac{c^2}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + c[\lambda_k(t) + c\tilde{g}_k(\mathbf{x}(t))]g_k(\mathbf{x}(t)) \\
\overset{(b)}{=}&\frac{1}{2}[\lambda_k(t)]^2 - \frac{1}{2}[\lambda_k(t+1) - \lambda_k(t)]^2 + c\lambda_k(t+1)g_k(\mathbf{x}(t))
\end{aligned}$$

where (a) follows from $\lambda_k(t)[\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))] = -c\tilde{g}_k(\mathbf{x}(t))[\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))]$, which can be shown by considering $\tilde{g}_k(\mathbf{x}(t)) = g_k(\mathbf{x}(t))$ and $\tilde{g}_k(\mathbf{x}(t)) \neq g_k(\mathbf{x}(t))$, separately; and (b) follows from the fact that $\lambda_k(t+1) = \lambda_k(t) + c\tilde{g}_k(\mathbf{x}(t))$. Summing over $k \in \{1, 2, \ldots, m\}$ yields $\frac{1}{2}\|\boldsymbol{\lambda}(t+1)\|^2 = \frac{1}{2}\|\boldsymbol{\lambda}(t)\|^2 - \frac{1}{2}c^2\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 + c[\boldsymbol{\lambda}(t+1)]^\mathsf{T}\mathbf{g}(\mathbf{x}(t))$. Rearranging the terms and dividing both sides by factor c yields the result. □

Lemma 2.2. *Let $\mathbf{x}^*, \sigma$ and β be constants defined in Assumption 2.1. If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1.1, then we have*

$$\frac{1}{c}\Delta(t) + f(\mathbf{x}(t)) \leq f(\mathbf{x}^*), \forall t \geq 0 \tag{2.6}$$

Proof. Fix $t \geq 0$. Since $f(\mathbf{x})$ is strongly convex with modulus σ; $g_k(\mathbf{x}), \forall k \in \{1, 2, \ldots, m\}$ are convex; and $\lambda_k(t), \forall k \in \{1, 2, \ldots, m\}$ are non-negative at each iteration t, the function $f(\mathbf{x}) + \sum_{k=1}^m \lambda_k(t)g_k(\mathbf{x})$ is also strongly convex with modulus σ at each iteration t. Note that $\mathbf{x}(t) = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \left\{ f(\mathbf{x}) + \sum_{k=1}^m \lambda_k(t)g_k(\mathbf{x}) \right\}$. By Corollary 1.2 with $\mathbf{x}^{opt} = \mathbf{x}(t)$ and $\mathbf{y} = \mathbf{x}^*$, we have

$$f(\mathbf{x}(t)) + \sum_{k=1}^m \lambda_k(t)g_k(\mathbf{x}(t)) \leq f(\mathbf{x}^*) + \sum_{k=1}^m \lambda_k(t)g_k(\mathbf{x}^*) - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2.$$

Hence, $f(\mathbf{x}(t)) \leq f(\mathbf{x}^*) + [\boldsymbol{\lambda}(t)]^\mathsf{T}\left[\mathbf{g}(\mathbf{x}^*) - \mathbf{g}(\mathbf{x}(t))\right] - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2$. Adding this inequality to

equation (2.4) yields

$$\begin{aligned}
&\frac{1}{c}\Delta(t) + f(\mathbf{x}(t)) \\
\leq & f(\mathbf{x}^*) - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + [\boldsymbol{\lambda}(t)]^{\mathsf{T}}[\mathbf{g}(\mathbf{x}^*) - \mathbf{g}(\mathbf{x}(t))] + [\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)).
\end{aligned}$$

Define

$$B(t) = -\frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + [\boldsymbol{\lambda}(t)]^{\mathsf{T}}[\mathbf{g}(\mathbf{x}^*) - \mathbf{g}(\mathbf{x}(t))] + [\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)).$$

Next, we need to show that $B(t) \leq 0$.

Since $\mathbf{x}^*$ is the optimal solution to the problem (2.1)-(2.3), we have $g_k(\mathbf{x}^*) \leq 0, \forall k \in \{1, 2, \ldots, m\}$. Note that $\lambda_k(t+1) \geq 0, \forall k \in \{1, 2, \ldots, m\}, \forall t \geq 0$. Thus,

$$[\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}^*) \leq 0, \quad \forall t \geq 0 \tag{2.7}$$

Now we have,

$$\begin{aligned}
B(t) =& -\frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + [\boldsymbol{\lambda}(t)]^{\mathsf{T}}[\mathbf{g}(\mathbf{x}^*) - \mathbf{g}(\mathbf{x}(t))] + [\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) \\
\overset{(a)}{\leq}& -\frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + [\boldsymbol{\lambda}(t)]^{\mathsf{T}}[\mathbf{g}(\mathbf{x}^*) - \mathbf{g}(\mathbf{x}(t))] + [\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) \\
& - [\boldsymbol{\lambda}(t+1)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}^*) \\
=& -\frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + [\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(t+1)]^{\mathsf{T}}[\mathbf{g}(\mathbf{x}^*) - \mathbf{g}(\mathbf{x}(t))] \\
\overset{(b)}{\leq}& -\frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + \|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(t+1)\|\|\mathbf{g}(\mathbf{x}(t)) - \mathbf{g}(\mathbf{x}^*)\| \\
\overset{(c)}{\leq}& -\frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 - \frac{\sigma}{2}\|\mathbf{x}(t) - \mathbf{x}^*\|^2 + \beta\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(t+1)\|\|\mathbf{x}(t) - \mathbf{x}^*\| \\
=& -\frac{1}{2c}\left[\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\| - c\beta\|\mathbf{x}(t) - \mathbf{x}^*\|\right]^2 - \frac{1}{2}(\sigma - c\beta^2)\|\mathbf{x}(t) - \mathbf{x}^*\|^2 \\
\overset{(d)}{\leq}& 0
\end{aligned}$$

where (a) follows from (2.7); (b) follows from the Cauchy-Schwarz inequality; (c) follows from Assumption 2.1; and (d) follows from $c \leq \frac{\sigma}{\beta^2}$. □

2.2.2 Objective Value Violation

Theorem 2.1 (Objective Value Violation)**.** *Let $\mathbf{x}^*, \sigma$ and β be constants defined in Assumption 2.1. If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1.1, then*

$$f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(0)\|^2}{2ct}, \forall t \geq 1.$$

Proof. Fix $t \geq 1$. By Lemma 2.2, we have $\frac{1}{c}\Delta(\tau) + f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*)$ for all $\tau \in \{0, 1, \ldots, t-1\}$. Summing over $\tau \in \{0, 1, \ldots, t-1\}$ we have:

$$\begin{aligned}
& \frac{1}{c}\sum_{\tau=0}^{t-1}\Delta(\tau) + \sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq t f(\mathbf{x}^*) \\
\Rightarrow \quad & \frac{1}{c}[L(t) - L(0)] + \sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq t f(\mathbf{x}^*) \\
\Rightarrow \quad & \frac{1}{t}\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*) + \frac{L(0) - L(t)}{ct} \leq f(\mathbf{x}^*) + \frac{L(0)}{ct}
\end{aligned}$$

Note that $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbf{x}(\tau)$ and by the convexity of $f(\mathbf{x})$, we have

$$f(\overline{\mathbf{x}}(t)) \leq \frac{1}{t}\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*) + \frac{L(0)}{ct} = f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(0)\|^2}{2ct}$$

□

Remark 2.1. *Similarly, we can prove that $f(\widetilde{\mathbf{x}}(2t)) \leq f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t)\|^2}{2ct}$ since $\widetilde{\mathbf{x}}(2t) = \frac{1}{t}\sum_{\tau=t}^{2t-1}\mathbf{x}(\tau)$. A later lemma (Lemma 2.4) guarantees that $\|\boldsymbol{\lambda}(t)\| \leq \sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|$, where $\boldsymbol{\lambda}^*$ is defined in Assumption 2.2. Thus, $f(\widetilde{\mathbf{x}}(2t)) \leq f(\mathbf{x}^*) + \frac{\left(\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|\right)^2}{2ct}, \forall t \geq 1$.*

2.2.3 Constraint Violation

The analysis of constraint violations is similar to that in [Nee14] for general convex programs. However, using the improved upper bound in Lemma 2.2, the convergence time of constraint violations in strongly convex programs is order-wise better than that in general convex programs.

Lemma 2.3. *For any $t_2 > t_1 \geq 0$,*

$$\lambda_k(t_2) \geq \lambda_k(t_1) + c\sum_{\tau=t_1}^{t_2-1} g_k(\mathbf{x}(\tau)), \forall k \in \{1, 2, \ldots, m\}.$$

In particular, for any $t > 0$,

$$\lambda_k(t) \geq \lambda_k(0) + c\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)), \forall k \in \{1, 2, \ldots, m\}.$$

Proof. Fix $k \in \{1, 2, \ldots, m\}$. Note that $\lambda_k(t_1 + 1) = \max\{\lambda_k(t_1) + cg_k(\mathbf{x}(t_1)), 0\} \geq \lambda_k(t_1) + cg_k(\mathbf{x}(t_1))$. By induction, this lemma follows. □

Lemma 2.4. *Let σ and β be constants defined in Assumption 2.1 and $\boldsymbol{\lambda}^*$ be the Lagrange multiplier vector defined in Assumption 2.2. If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1.1, then $\boldsymbol{\lambda}(t)$ satisfies*

$$\|\boldsymbol{\lambda}(t)\| \leq \sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|, \forall t \geq 1. \tag{2.8}$$

Proof. Fix $t \geq 1$. Let $\mathbf{x}^*$ be the optimal solution to the problem (2.1)-(2.3). Assumption 2.2 implies that

$$f(\mathbf{x}^*) = q(\boldsymbol{\lambda}^*) \leq f(\mathbf{x}(\tau)) + \sum_{k=1}^{m} \lambda_k^* g_k(\mathbf{x}(\tau)), \forall \tau \in \{0, 1, \ldots, t-1\},$$

where the inequality follows from the definition of $q(\boldsymbol{\lambda})$.

Thus, we have $f(\mathbf{x}^*) - f(\mathbf{x}(\tau)) \leq \sum_{k=1}^{m} \lambda_k^* g_k(\mathbf{x}(\tau)), \forall \tau \in \{0, 1, \ldots, t-1\}$. Summing over $\tau \in \{0, 1, \ldots, t-1\}$ yields

$$\begin{aligned}
tf(\mathbf{x}^*) - \sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) &\leq \sum_{\tau=0}^{t-1}\sum_{k=1}^{m} \lambda_k^* g_k(\mathbf{x}(\tau)) \\
&= \sum_{k=1}^{m} \lambda_k^* \Big[\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau))\Big] \\
&\overset{(a)}{\leq} \frac{1}{c}\sum_{k=1}^{m} \lambda_k^* [\lambda_k(t) - \lambda_k(0)] \\
&\leq \frac{1}{c}\sum_{k=1}^{m} \lambda_k^* \lambda_k(t) \\
&\overset{(b)}{\leq} \frac{1}{c}\|\boldsymbol{\lambda}^*\|\|\boldsymbol{\lambda}(t)\|
\end{aligned} \tag{2.9}$$

where (a) follows from Lemma 2.3 and (b) follows from the Cauchy-Schwarz inequality. On the

other hand, summing (2.6) in Lemma 2.2 over $\tau \in \{0, 1, \ldots, t-1\}$ yields

$$\begin{aligned} tf(\mathbf{x}^*) - \sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) &\geq \frac{L(t) - L(0)}{c} \\ &= \frac{\|\boldsymbol{\lambda}(t)\|^2 - \|\boldsymbol{\lambda}(0)\|^2}{2c} \end{aligned} \tag{2.10}$$

Combining (2.9) and (2.10) yields

$$\begin{aligned} & \frac{\|\boldsymbol{\lambda}(t)\|^2 - \|\boldsymbol{\lambda}(0)\|^2}{2c} \leq \frac{1}{c}\|\boldsymbol{\lambda}^*\|\|\boldsymbol{\lambda}(t)\| \\ \Rightarrow \quad & \left[\|\boldsymbol{\lambda}(t)\| - \|\boldsymbol{\lambda}^*\|\right]^2 \leq \|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2 \\ \Rightarrow \quad & \|\boldsymbol{\lambda}(t)\| \leq \sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\| \end{aligned}$$

□

Theorem 2.2 (Constraint Violation)**.** *Let σ and β be constants defined in Assumption 2.1 and $\boldsymbol{\lambda}^*$ be the Lagrange multiplier vector defined in Assumption 2.2. If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1.1, then*

$$g_k(\overline{\mathbf{x}}(t)) \leq \frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{ct}, \forall k \in \{1, 2, \ldots, m\}, \forall t \geq 1.$$

Proof. Fix $t \geq 1$ and $k \in \{1, 2, \ldots, m\}$. Recall that $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbf{x}(\tau)$. Thus,

$$\begin{aligned} g_k(\overline{\mathbf{x}}(t)) &\overset{(a)}{\leq} \frac{1}{t}\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)) \\ &\overset{(b)}{\leq} \frac{\lambda_k(t) - \lambda_k(0)}{ct} \\ &\leq \frac{\lambda_k(t)}{ct} \\ &\leq \frac{\|\boldsymbol{\lambda}(t)\|}{ct} \\ &\overset{(c)}{\leq} \frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{ct} \end{aligned}$$

where (a) follows from the convexity of $g_k(\mathbf{x}), k \in \{1, 2, \ldots, m\}$; (b) follows from Lemma 2.3; and (c) follows from Lemma 2.4. □

Remark 2.2. *Similarly, we can prove that*

$$g_k(\widetilde{\mathbf{x}}(2t)) \leq \frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{ct}, \forall k \in \{1, 2, \ldots, m\}, \forall t \geq 1.$$

The next corollary provides a lower bound of $f(\overline{\mathbf{x}}(t))$ and follows directly from Assumption 2.2 and Theorem 2.2.

Corollary 2.1. *Let σ and β be constants defined in Assumption 2.1 and let $\boldsymbol{\lambda}^*$ be the Lagrange multiplier vector defined in Assumption 2.2. If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1, then*

$$f(\overline{\mathbf{x}}(t)) \geq f(\mathbf{x}^*) - \frac{1}{t}\frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{c}\sum_{k=1}^{m}\lambda_k^*, \forall t \geq 1.$$

Proof. Fix $t \geq 1$. By Assumption 2.2, we have

$$f(\overline{\mathbf{x}}(t)) + \sum_{k=1}^{m}\lambda_k^* g_k(\overline{\mathbf{x}}(t)) \geq q(\boldsymbol{\lambda}^*) = f(\mathbf{x}^*) + \sum_{k=1}^{m}\lambda_k^* g_k(\mathbf{x}^*) = f(\mathbf{x}^*).$$

Thus, we have

$$\begin{aligned} f(\overline{\mathbf{x}}(t)) \geq & f(\mathbf{x}^*) - \sum_{k=1}^{m}\lambda_k^* g_k(\overline{\mathbf{x}}(t)) \\ \overset{(a)}{\geq} & f(\mathbf{x}^*) - \sum_{k=1}^{m}\lambda_k^* \frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{ct} \\ = & f(\mathbf{x}^*) - \frac{1}{t}\frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{c}\sum_{k=1}^{m}\lambda_k^*, \end{aligned}$$

where (a) follows from the constraint violation bound, i.e., Theorem 2.2, and the fact that $\lambda_k^* \geq 0, \forall k \in \{1, 2, \ldots, m\}$. □

2.2.4 Convergence Time of Algorithm 1.1

The next theorem summarizes Theorem 2.1 and Theorem 2.2.

Theorem 2.3. *Let $\mathbf{x}^*, \sigma$ and β be constants defined in Assumption 2.1 and let $\boldsymbol{\lambda}^*$ be the Lagrange*

multiplier vector defined in Assumption 2.2. If $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then for all* $t \geq 1$*,*

$$f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(0)\|^2}{2ct}.$$
$$g_k(\overline{\mathbf{x}}(t)) \leq \frac{\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|}{t}, \forall k \in \{1, 2, \ldots, m\}.$$

Specifically, if $\boldsymbol{\lambda}(0) = \mathbf{0}$*, then*

$$f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*).$$
$$g_k(\overline{\mathbf{x}}(t)) \leq \frac{2\|\boldsymbol{\lambda}^*\|}{ct}, \forall k \in \{1, 2, \ldots, m\}.$$

In summary, if $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then* $\overline{\mathbf{x}}(t)$ *ensures that error decays like* $O(\frac{1}{t})$ *and provides an* ϵ*-approximate solution with convergence time* $O(\frac{1}{\epsilon})$*.*

Remark 2.3. *If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then* $\widetilde{\mathbf{x}}(t)$ *also ensures that error decays like* $O(\frac{1}{t})$ *and provides an* ϵ*-approximate solution with convergence time* $O(\frac{1}{\epsilon})$*.*

2.3 Geometric Convergence of Algorithm 1.1 with Sliding Running Averages

This section shows that the convergence time of Algorithm 1.1 with sliding running averages $\widetilde{\mathbf{x}}(t)$ is $O(\log(\frac{1}{\epsilon}))$ when the dual function of the problem (2.1)-(2.3) satisfies additional assumptions.

2.3.1 Smooth Dual Function

Recall the definition of smooth functions in Definition 1.6. Define $q(\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) + \sum_{k=1}^m \lambda_k g_k(\mathbf{x})\} \overset{(a)}{=} \min_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) + \sum_{k=1}^m \lambda_k g_k(\mathbf{x})\}$ as the *Lagrangian dual function* of the problem (2.1)-(2.3) where (a) follows because $f(\mathbf{x})$ is strongly convex, $f(\mathbf{x})$ and $g_k(\mathbf{x})$ are continuous and $\mathcal{X}$ is a closed set. For fixed $\boldsymbol{\lambda} \in \mathbb{R}_+^m$, $f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathsf{T}}\mathbf{g}(\mathbf{x})$ is strongly convex with respect to $\mathbf{x} \in \mathcal{X}$ with modulus α. Define $\mathbf{x}(\boldsymbol{\lambda}) = \text{argmin}_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathsf{T}}\mathbf{g}(\mathbf{x})\}$. By Danskin's theorem (Proposition B.25 in [Ber99]), $q(\boldsymbol{\lambda})$ is differentiable with gradient $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}) = \mathbf{g}(\mathbf{x}(\boldsymbol{\lambda}))$.

Lemma 2.5 (Smooth Dual Function)**.** *The Lagrangian dual function of the problem* (2.1)-(2.3)*, $q(\boldsymbol{\lambda})$, is smooth on $\mathbb{R}^m_+$ with modulus $\gamma = \frac{\beta^2}{\sigma}$.*

Proof. Fix $\boldsymbol{\lambda}, \boldsymbol{\mu} \in \mathbb{R}^m_+$. Let $\mathbf{x}(\boldsymbol{\lambda}) = \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) + \boldsymbol{\lambda}^\mathsf{T}\mathbf{g}(\mathbf{x})\}$ and $\mathbf{x}(\boldsymbol{\mu}) = \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) + \boldsymbol{\mu}^\mathsf{T}\mathbf{g}(\mathbf{x})\}$. Recall that for fixed $\boldsymbol{\lambda} \in \mathbb{R}^m_+$, $f(\mathbf{x}) + \boldsymbol{\lambda}^\mathsf{T}\mathbf{g}(\mathbf{x})$ is strongly convex with respect to $\mathbf{x} \in \mathcal{X}$ with modulus α. By Corollary 1.2, we have $f(\mathbf{x}(\boldsymbol{\lambda})) + \boldsymbol{\lambda}^\mathsf{T}\mathbf{g}(\mathbf{x}(\boldsymbol{\lambda})) \leq f(\mathbf{x}(\boldsymbol{\mu})) + \boldsymbol{\lambda}^\mathsf{T}\mathbf{g}(\mathbf{x}(\boldsymbol{\mu})) - \frac{\alpha}{2}\|\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{x}(\boldsymbol{\mu})\|^2$ and $f(\mathbf{x}(\boldsymbol{\mu})) + \boldsymbol{\mu}^\mathsf{T}\mathbf{g}(\mathbf{x}(\boldsymbol{\mu})) \leq f(\mathbf{x}(\boldsymbol{\lambda})) + \boldsymbol{\mu}^\mathsf{T}\mathbf{g}(\mathbf{x}(\boldsymbol{\lambda})) - \frac{\alpha}{2}\|\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{x}(\boldsymbol{\mu})\|^2$. Summing the above two inequalities and simplifying gives

$$\begin{aligned}\alpha\|\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{x}(\boldsymbol{\mu})\|^2 &\leq [\boldsymbol{\mu} - \boldsymbol{\lambda}]^\mathsf{T}[\mathbf{g}(\mathbf{x}(\boldsymbol{\lambda})) - \mathbf{g}(\mathbf{x}(\boldsymbol{\mu}))] \\ &\overset{(a)}{\leq} \|\boldsymbol{\lambda} - \boldsymbol{\mu}\|\|\mathbf{g}(\mathbf{x}(\boldsymbol{\lambda})) - \mathbf{g}(\mathbf{x}(\boldsymbol{\mu}))\| \\ &\overset{(b)}{\leq} \beta\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|\|\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{x}(\boldsymbol{\mu})\|\end{aligned}$$

where (a) follows from the Cauchy-Schwarz inequality and (b) follows because $\mathbf{g}(\mathbf{x})$ is Lipschitz continuous by Assumption 2.1. This implies

$$\|\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{x}(\boldsymbol{\mu})\| \leq \frac{\beta}{\alpha}\|\boldsymbol{\lambda} - \boldsymbol{\mu}\| \tag{2.11}$$

Thus, we have

$$\begin{aligned}\|\nabla q(\boldsymbol{\lambda}) - \nabla q(\boldsymbol{\mu})\| &\overset{(a)}{=} \|\mathbf{g}(\mathbf{x}(\boldsymbol{\lambda})) - \mathbf{g}(\mathbf{x}(\boldsymbol{\mu}))\| \\ &\overset{(b)}{\leq} \beta\|\mathbf{x}(\boldsymbol{\lambda}) - \mathbf{x}(\boldsymbol{\mu})\| \\ &\overset{(c)}{\leq} \frac{\beta^2}{\alpha}\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|\end{aligned}$$

where (a) follows from $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}) = \mathbf{g}(\mathbf{x}(\boldsymbol{\lambda}))$; (b) follows from the Lipschitz continuity of $\mathbf{g}(\mathbf{x})$; and (c) follows from (2.11).

Thus, $q(\boldsymbol{\lambda})$ is smooth on $\mathbb{R}^m_+$ with modulus $\gamma = \frac{\beta^2}{\alpha}$. □

Since $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}(t)) = \mathbf{g}(\mathbf{x}(t))$, the dynamic of $\boldsymbol{\lambda}(t)$ can be interpreted as the projected gradient method with step size c to solve $\max_{\boldsymbol{\lambda}\in\mathbb{R}^m_+}\{q(\boldsymbol{\lambda})\}$ where $\mathbf{q}(\cdot)$ is a smooth function by Lemma 2.5. Thus, we have the next lemma.

Lemma 2.6. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.2.*

If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1.1, then

$$q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t)) \leq \frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2, \quad \forall t \geq 1.$$

Proof. Recall that a projected gradient descent algorithm with step size $c < \frac{1}{\gamma}$ converges to the maximum of a concave function with smooth modulus γ with the error decaying like $O(\frac{1}{t})$. Thus, this lemma follows. The proof is essentially the same as the convergence time proof of the projected gradient method for set constrained smooth optimization in [Nes04]. See Section 2.7.1 for the detailed proof. □

2.3.2 Problems with Locally Quadratic Dual Functions

In addition to Assumptions 2.1-2.2, we further require the next assumption in this subsection.

Assumption 2.3 (Locally Quadratic Dual Function)**.** *Let $\boldsymbol{\lambda}^*$ be a Lagrange multiplier vector of the problem* (2.1)-(2.3) *defined in Assumption 2.2. There exists $D_q > 0$ and $L_q > 0$, where the subscript q denotes locally "quadratic", such that for all $\boldsymbol{\lambda} \in \{\boldsymbol{\lambda} \in \mathbb{R}^m_+ : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_q\}$, the dual function $q(\boldsymbol{\lambda}) = \min_{\mathbf{x}\in\mathcal{X}} \left\{ f(\mathbf{x}) + \sum_{k=1}^m \lambda_k g_k(\mathbf{x}) \right\}$ satisfies*

$$q(\boldsymbol{\lambda}^*) \geq q(\boldsymbol{\lambda}) + L_q\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2.$$

Lemma 2.7. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. Let $q(\boldsymbol{\lambda}), \boldsymbol{\lambda}^*, D_q$ and L_q be defined in Assumption 2.3. We have the following properties:*

1. *If $\boldsymbol{\lambda} \in \mathbb{R}^m_+$ and $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}) \leq L_q D_q^2$, then $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_q$.*

2. *The Lagrange multiplier defined in Assumption 2.2 is unique.*

Proof.

1. Assume not and there exists $\boldsymbol{\lambda}' \in \mathbb{R}^m_+$ such that $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}') \leq L_q D_q^2$ and $\|\boldsymbol{\lambda}' - \boldsymbol{\lambda}^*\| > D_q$. Define $\boldsymbol{\lambda} = (1-\eta)\boldsymbol{\lambda}^* + \eta\boldsymbol{\lambda}'$ for some $\eta \in (0,1)$. Note that $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| = \|\eta(\boldsymbol{\lambda}' - \boldsymbol{\lambda}^*)\| = \eta\|(\boldsymbol{\lambda}' - \boldsymbol{\lambda}^*)\|$. Thus, we can choose $\eta \in (0,1)$ such that $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| = D_q$, i.e., $\eta = \frac{D_q}{\|\boldsymbol{\lambda}' - \boldsymbol{\lambda}^*\|}$. Note that $\boldsymbol{\lambda} \in \mathbb{R}^m_+$ because $\boldsymbol{\lambda}' \in \mathbb{R}^m_+$ and $\boldsymbol{\lambda}^* \in \mathbb{R}^m_+$. Since the dual function $q(\cdot)$ is a concave function, we have $q(\boldsymbol{\lambda}) \geq (1-\eta)q(\boldsymbol{\lambda}^*) + \eta q(\boldsymbol{\lambda}')$. Thus, $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}) \leq q(\boldsymbol{\lambda}^*) -$

$\big((1-\eta)q(\boldsymbol{\lambda}^*)+\eta q(\boldsymbol{\lambda}')\big) = \eta(q(\boldsymbol{\lambda}^*)-q(\boldsymbol{\lambda}')) \leq \eta L_q D_q^2$. This contradicts Assumption 2.3 that $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}) \geq L_q\|\boldsymbol{\lambda}-\boldsymbol{\lambda}^*\|^2 = L_q D_q^2$.

2. Assume not and there exists $\boldsymbol{\mu}^* \neq \boldsymbol{\lambda}^*$ such that $\boldsymbol{\mu}^* \in \mathbb{R}_+^m$ and $q(\boldsymbol{\mu}^*) = q(\boldsymbol{\lambda}^*)$. By part 1 of this lemma, $\|\boldsymbol{\mu}^* - \boldsymbol{\lambda}^*\| \leq D_q$. Thus, we have

$$\begin{aligned} q(\boldsymbol{\mu}^*) &\overset{(a)}{\leq} q(\boldsymbol{\lambda}^*) - L_q\|\boldsymbol{\mu}^* - \boldsymbol{\lambda}^*\|^2 \\ &\overset{(b)}{<} q(\boldsymbol{\lambda}^*) \end{aligned}$$

where (a) follows from Assumption 2.3 and (b) follows from the assumption that $\boldsymbol{\mu}^* \neq \boldsymbol{\lambda}^*$. This contradicts the assumption that $q(\boldsymbol{\mu}^*) = q(\boldsymbol{\lambda}^*)$.

□

Define

$$T_q = \frac{\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2}{2cL_q D_q^2}, \tag{2.12}$$

where the subscript q denotes locally "quadratic".

Lemma 2.8. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then* $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq D_q$ *for all* $t \geq T_q$*, where* T_q *is defined in* (2.12).

Proof. By Lemma 2.6 and Lemma 2.7, if $\frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2 \leq L_q D_q^2$, then $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq D_q$. It can be checked that $t \geq \frac{\|\boldsymbol{\lambda}(0)-\boldsymbol{\lambda}^*\|^2}{2cL_q D_q^2}$ implies that $\frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2 \leq L_q D_q^2$. □

Lemma 2.9. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then*

1. $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq \frac{1}{\sqrt{t}}\frac{1}{\sqrt{2cL_q}}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|, \forall t \geq T_q$*, where* T_q *is defined in* (2.12).
2. $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq \big(\sqrt{\frac{1}{1+2cL_q}}\big)^{t-T_q}\|\boldsymbol{\lambda}(T_q) - \boldsymbol{\lambda}^*\| \leq \big(\frac{1}{\sqrt{1+2cL_q}}\big)^t D_q(1+2cL_q)^{\frac{T_q}{2}}, \forall t \geq T_q$ *, where* T_q *is defined in* (2.12).

Proof.

1. By Lemma 2.6, $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t)) \leq \frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2, \forall t \geq 1$. By Lemma 2.8 and Assumption 2.3, $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t)) \geq L_q\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2, \forall t \geq T_q$. Thus, we have $L_q\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 \leq \frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2, \forall t \geq T_q$, which implies that $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq \frac{1}{\sqrt{t}}\frac{1}{\sqrt{2cL_q}}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|, \forall t \geq T_q$.

2. By part 1, we know $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq D_q, \forall t \geq T_q$. The second part is essentially a local version of Theorem 12 in [NNG15], which shows that the projected gradient method for set constrained smooth convex optimization converge geometrically if the objective function satisfies a quadratic growth condition. See Section 2.7.2 for the detailed proof.

□

Corollary 2.2. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then* $\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \leq 2\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t D_q(1+2cL_q)^{\frac{T_q}{2}}, \forall t \geq T_q$, *where* T_q *be defined in* (2.12).

Proof.

$$\begin{aligned}\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \leq & \|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}^*\| + \|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \\ \overset{(a)}{\leq} & \big(\frac{1}{\sqrt{1+2cL_q}}\big)^{2t} D_q(1+2cL_q)^{\frac{T_q}{2}} + \big(\frac{1}{\sqrt{1+2cL_q}}\big)^{t} D_q(1+2cL_q)^{\frac{T_q}{2}} \\ \overset{(b)}{\leq} & 2\big(\frac{1}{\sqrt{1+2cL_q}}\big)^{t} D_q(1+2cL_q)^{\frac{T_q}{2}},\end{aligned}$$

where (a) follows from part 2 in Lemma 2.9; and (b) follows from $\frac{1}{\sqrt{1+cLq}} < 1$. □

Theorem 2.4. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then*

$$f(\widetilde{\mathbf{x}}(2t)) \leq f(\mathbf{x}^*) + \frac{1}{t}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t \eta, \forall t \geq T_q,$$

where $\eta_q = \frac{2D_q^2(1+2cL_q)^{T_q} + 2D_q(1+2cL_q)^{\frac{T_q}{2}}\left(\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|\right)}{c}$ *and* T_q *is defined in* (2.12).

Proof. Fix $t \geq T_q$. By Lemma 2.2, we have $\frac{1}{c}\Delta(\tau) + f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*)$ for all $\tau \in \{0, 1, \ldots\}$. Summing over $\tau \in \{t, t+1, \ldots, 2t-1\}$ yields $\frac{1}{c}\sum_{\tau=t}^{2t-1} \Delta(\tau) + \sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \leq tf(\mathbf{x}^*)$. Dividing by factor t yields

$$\frac{1}{t}\sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*) + \frac{L(t) - L(2t)}{ct} \tag{2.13}$$

Thus, we have

$$
\begin{aligned}
f(\widetilde{\mathbf{x}}(2t)) &\overset{(a)}{\leq} \frac{1}{t}\sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \overset{(b)}{\leq} f(\mathbf{x}^*) + \frac{L(t)-L(2t)}{ct} \\
&= f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t)\|^2 - \|\boldsymbol{\lambda}(2t)\|^2}{2ct} \\
&= f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t)-\boldsymbol{\lambda}(2t)+\boldsymbol{\lambda}(2t)\|^2 - \|\boldsymbol{\lambda}(2t)\|^2}{2ct} \\
&\overset{(c)}{\leq} f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t)-\boldsymbol{\lambda}(2t)\|^2 + 2\|\boldsymbol{\lambda}(2t)\|\|\boldsymbol{\lambda}(t)-\boldsymbol{\lambda}(2t)\|}{2ct} \\
&\overset{(d)}{\leq} f(\mathbf{x}^*) + \frac{\Big(2\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t D_q(1+2cL_q)^{\frac{T_q}{2}}\Big)^2}{2ct} + \frac{4\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t D_q(1+2cL_q)^{\frac{T_q}{2}}\|\boldsymbol{\lambda}(2t)\|}{2ct} \\
&\overset{(e)}{\leq} f(\mathbf{x}^*) + \frac{1}{t}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t\Big(\frac{2D_q^2(1+2cL_q)^{T_q}}{c} + \frac{2D_q(1+2cL_q)^{\frac{T_q}{2}}\|\boldsymbol{\lambda}(2t)\|}{c}\Big) \\
&\overset{(f)}{=} f(\mathbf{x}^*) + \frac{1}{t}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t \eta_q
\end{aligned}
$$

where (a) follows from $\widetilde{\mathbf{x}}(2t) = \frac{1}{t}\sum_{\tau=t}^{2t-1}\mathbf{x}(\tau)$ and the convexity of $f(\mathbf{x})$; (b) follows from (2.13); (c) follows from the Cauchy-Schwarz inequality; (d) follows from Corollary 2.2; (e) follows from $\frac{1}{\sqrt{1+2cL_q}} < 1$; and (f) follows from $\|\boldsymbol{\lambda}(2t)\| \leq \sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|$ and the definition of η_q. □

Theorem 2.5. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then*

$$
g_k(\widetilde{\mathbf{x}}(2t)) \leq \frac{2D_q(1+2cL_q)^{\frac{T_q}{2}}}{ct}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t, \forall k \in \{1,2,\ldots,m\}, \forall t \geq T_q,
$$

where T_q *is defined in* (2.12).

Proof. Fix $t \geq T_q$ and $k \in \{1,2,\ldots,m\}$. Thus, we have

$$
\begin{aligned}
g_k(\widetilde{\mathbf{x}}(2t)) &\overset{(a)}{\leq} \frac{1}{t}\sum_{\tau=t}^{2t-1} g_k(\mathbf{x}(\tau)) \\
&\overset{(b)}{\leq} \frac{1}{ct}\big(\lambda_k(2t) - \lambda_k(t)\big) \\
&\leq \frac{1}{ct}\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \\
&\overset{(c)}{\leq} \frac{2D_q(1+2cL_q)^{\frac{T_q}{2}}}{ct}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t
\end{aligned}
$$

where (a) follows from the convexity of $g_k(\mathbf{x})$; (b) follows from Lemma 2.3; and (c) follows from Corollary 2.2. □

The next theorem summarizes both Theorem 2.4 and Theorem 2.5.

Theorem 2.6. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.3. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then for all* $t \geq T_q$,

$$f(\widetilde{\mathbf{x}}(2t)) \leq f(\mathbf{x}^*) + \frac{1}{t}\Big(\frac{1}{\sqrt{1+2cL_q}}\Big)^t \eta_q, \forall t \geq T_q,$$
$$g_k(\widetilde{\mathbf{x}}(2t)) \leq \frac{2D_q(1+2cL_q)^{\frac{T_q}{2}}}{ct}\Big(\frac{1}{\sqrt{1+2cL_q}}\Big)^t, \forall k \in \{1,2,\ldots,m\}, \forall t \geq T_q,$$

where $\eta_q = \frac{2D_q^2(1+2cL_q)^{T_q}+2D_q(1+2cL_q)^{\frac{T_q}{2}}\left(\sqrt{\|\boldsymbol{\lambda}(0)\|^2+\|\boldsymbol{\lambda}^*\|^2}+\|\boldsymbol{\lambda}^*\|\right)}{c}$ *and* T_q *is defined in* (2.12). *In summary, if* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then* $\widetilde{\mathbf{x}}(t)$ *ensures error decays like* $O\big(\big(\frac{1}{1+2cL_q}\big)^{t/2}\big)$ *and provides an* ϵ*-approximate solution with convergence time* $O(\log(\frac{1}{\epsilon}))$.

2.3.3 Problems with Locally Strongly Concave Dual Functions

The following assumption is stronger than Assumptions 2.3 but can be easier to verify in certain cases. For example, if the dual function of the convex program is available, Assumption 2.4 is easier to verify, e.g., by studying the Hessian of the dual function.

Assumption 2.4 (Locally Strongly Concave Dual Function)**.** *Let* $\boldsymbol{\lambda}^*$ *be a Lagrange multiplier vector defined in Assumption 2.2. There exists* $D_c > 0$ *and* $L_c > 0$*, where the subscript c denotes locally strongly "concave", such that the dual function* $q(\boldsymbol{\lambda})$ *is strongly concave with modulus* L_c *over* $\{\boldsymbol{\lambda} \in \mathbb{R}^m_+ : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}$.

The next lemma summarizes that Assumption 2.4 implies Assumption 2.3.

Lemma 2.10. *If the strongly convex program* (2.1)-(2.3) *satisfies Assumption 2.4, then it also satisfies Assumption 2.3 with* $D_q = D_c$ *and* $L_q = \frac{L_c}{2}$.

Proof. Since $q(\cdot)$ is strongly concave and is maximized at $\boldsymbol{\lambda}^*$, by Corollary 1.3, $q(\boldsymbol{\lambda}^*) \geq q(\boldsymbol{\lambda}) + \frac{L_c}{2}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2$ for all $\boldsymbol{\lambda} \in \{\boldsymbol{\lambda} \in \mathbb{R}^m_+ : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}$. □

Since Assumption 2.4 implies Assumption 2.3, by the results from the previous subsection, $\widetilde{\mathbf{x}}(t)$ from Algorithm 1.1 provides an ϵ-approximate solution with convergence time $O(\log(\frac{1}{\epsilon}))$. In this subsection, we show that if the problem (2.1)-(2.3) satisfies Assumption 2.4, then the geometric error decay has a smaller contraction modulus.

The next lemma relates the smoothness of the dual function and Assumption 2.4.

Lemma 2.11. *If function h is both smooth with modulus γ and strongly concave with modulus L_c over set $\mathcal{X}$, which is not a singleton, then $L_c \leq \gamma$.*

Proof. This is a basic fact in convex analysis. See Section 2.7.3 for the detailed proof. □

For any problem (2.1)-(2.3) satisfying Assumptions 2.1-2.2 and 2.4, we define

$$T_c = \frac{\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2}{cL_c D_c^2}, \tag{2.14}$$

where the subscript c denotes locally strongly "concave".

Lemma 2.12. *Cosider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.2 and 2.4. Let D_c and L_c be defined in Assumption 2.4. If $c \leq \frac{\sigma}{\beta^2}$ in Algorithm 1.1, then*

1. *$\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq D_c$ for all $t \geq T_c$, where T_c is defined in* (2.14).
2. *$\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq \left(\sqrt{1-cL_c}\right)^{t-T_c} \|\boldsymbol{\lambda}(T_c) - \boldsymbol{\lambda}^*\| \leq \left(\sqrt{1-cL_c}\right)^{t} \frac{D_c}{(\sqrt{1-cL_c})^{T_c}}$, $\forall t \geq T_c$, where T_c is defined in* (2.14).

Proof.

1. By Lemma 2.10, $q(\cdot)$ is locally quadratic with $D_q = D_c$ and $L_q = \frac{L_c}{2}$. The remaining part of the proof is identical to the proof of Lemma 2.8.

2. By part 1 of this lemma, $\boldsymbol{\lambda}(t) \in \{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}, \forall t \geq T_c$. That is, the dynamic of $\boldsymbol{\lambda}(t), t \geq T_c$ is the same as that in the projected gradient method with step size c to solve[2] $\max_{\boldsymbol{\lambda} \in \{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}} \{q(\boldsymbol{\lambda})\}$. Thus, the part is essentially a local version of the convergence time result of the projected gradient method for set constrained smooth and strongly convex optimization [Nes04]. See Section 2.7.4 for the detailed proof.

[2]Recall that the projected gradient method with constant step size when applied to set constrained smooth and strongly convex optimization converges to the optimal solution at the rate $O(\kappa^t)$ where κ is a parameter depending on the step size, the smoothness modulus and the strong convexity modulus [Nes04].

□

The next corollary follows from Part 2 of Lemma 2.12.

Corollary 2.3. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.2 and 2.4. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then*

$$\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \leq \left(\sqrt{1-cL_c}\right)^t \frac{2D_c}{(\sqrt{1-cL_c})^{T_c}}, \forall t \geq T_c,$$

where T_c *is defined in* (2.14).

Proof.

$$\begin{aligned}
&\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \\
\leq &\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}^*\| + \|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \\
\overset{(a)}{\leq} &\left(\sqrt{1-cL_c}\right)^{2t} \frac{D_c}{(\sqrt{1-cL_c})^{T_c}} + \left(\sqrt{1-cL_c}\right)^t \frac{D_c}{(\sqrt{1-cL_c})^{T_c}} \\
\overset{(b)}{\leq} &\left(\sqrt{1-cL_c}\right)^t \frac{2D_c}{(\sqrt{1-cL_c})^{T_c}}
\end{aligned}$$

where (a) follows from part 2 of Lemma 2.12 and (b) follows from the fact that $\sqrt{1-cL_c} < 1$, which is implied by $c \leq \frac{1}{\gamma}$ and $L_c \leq \gamma$. □

Theorem 2.7. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.2 and 2.4. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then*

$$f(\widetilde{\mathbf{x}}(2t)) \leq f(\mathbf{x}^*) + \frac{1}{t}\left(\sqrt{1-cL_c}\right)^t \eta_c, \quad \forall t \geq T_c,$$

where $\eta_c = \frac{2D_c^2}{(\sqrt{1-cL_c})^{2T_c}} + \frac{2D_c\left(\sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|\right)}{(\sqrt{1-cL_c})^{T_c}}$ *is a fixed constant and* T_c *is defined in* (2.14).

Proof. Fix $t \geq T_c$. By Lemma 2.2, we have $\frac{1}{c}\Delta(\tau) + f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*)$ for all $\tau \in \{0, 1, \ldots\}$.

Summing over $\tau \in \{t, t+1, \ldots, 2t-1\}$ we have:

$$\begin{aligned} & \frac{1}{c}\sum_{\tau=t}^{2t-1}\Delta(\tau) + \sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \leq t f(\mathbf{x}^*) \\ \Rightarrow \quad & \frac{1}{c}[L(2t) - L(t)] + \sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \leq t f(\mathbf{x}^*) \\ \Rightarrow \quad & \frac{1}{t}\sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*) + \frac{L(t) - L(2t)}{ct} \end{aligned} \tag{2.15}$$

Thus, we have

$$\begin{aligned} f(\widetilde{\mathbf{x}}(2t)) &\overset{(a)}{\leq} \frac{1}{t}\sum_{\tau=t}^{2t-1} f(\mathbf{x}(\tau)) \overset{(b)}{\leq} f(\mathbf{x}^*) + \frac{L(t) - L(2t)}{ct} \\ &= f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t)\|^2 - \|\boldsymbol{\lambda}(2t)\|^2}{2ct} \\ &= f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(2t) + \boldsymbol{\lambda}(2t)\|^2 - \|\boldsymbol{\lambda}(2t)\|^2}{2ct} \\ &= f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(2t)\|^2 + 2[\boldsymbol{\lambda}(2t)]^{\mathsf{T}}[\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(2t)]}{2ct} \\ &\overset{(c)}{\leq} f(\mathbf{x}^*) + \frac{\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(2t)\|^2 + 2\|\boldsymbol{\lambda}(2t)\|\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(2t)\|}{2ct} \\ &\overset{(d)}{\leq} f(\mathbf{x}^*) + \frac{\Big(\big(\sqrt{1-cL_c}\big)^t \frac{2D_c}{(\sqrt{1-cL_c})^{T_c}}\Big)^2}{2ct} + \frac{2\Big(\big(\sqrt{1-cL_c}\big)^t \frac{2D_c}{(\sqrt{1-cL_c})^{T_c}}\Big)\|\boldsymbol{\lambda}(2t)\|}{2ct} \\ &\overset{(e)}{\leq} f(\mathbf{x}^*) + \frac{\Big(\sqrt{1-cL_c}\Big)^t\Big(\frac{2D_c^2}{(\sqrt{1-cL_c})^{2T_c}} + \frac{2D_c\|\boldsymbol{\lambda}(2t)\|}{(\sqrt{1-cL_c})^{T_c}}\Big)}{t} \\ &\overset{(f)}{=} f(\mathbf{x}^*) + \frac{1}{t}\big(\sqrt{1-cL_c}\big)^t \eta_c \end{aligned}$$

where (a) follows from the fact that $\widetilde{\mathbf{x}}(2t) = \frac{1}{t}\sum_{\tau=t}^{2t-1}\mathbf{x}(\tau)$ and the convexity of $f(\mathbf{x})$; (b) follows from (2.15); (c) follows from the Cauchy-Schwarz inequality; (d) is true because

$$\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \leq \big(\sqrt{1-cL_c}\big)^t \frac{2D_c}{(\sqrt{1-cL_c})^{T_c}}, \forall t \geq T_c$$

by Corollary 2.3; (e) follows from the fact that $\sqrt{1-cL_c} < 1$; and (f) follows from the fact that $\|\boldsymbol{\lambda}(2t)\| \leq \sqrt{\|\boldsymbol{\lambda}(0)\|^2 + \|\boldsymbol{\lambda}^*\|^2} + \|\boldsymbol{\lambda}^*\|$ by Lemma 2.4 and the definition of η_c. □

Theorem 2.8. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.2*

and 2.4. If $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then*

$$g_k(\widetilde{\mathbf{x}}(2t)) \leq \frac{1}{t}\big(\sqrt{1-cL_c}\big)^t \frac{2D_c}{c(\sqrt{1-cL_c})^{T_c}}, \forall k \in \{1,2,\ldots,m\}, \forall t \geq T_c$$

where T_c *is defined in* (2.14).

Proof. Fix $t \geq T_c$ and $k \in \{1,2,\ldots,m\}$. Thus, we have

$$\begin{aligned} g_k(\widetilde{\mathbf{x}}(2t)) &\overset{(a)}{\leq} \frac{1}{t}\sum_{\tau=t}^{2t-1} g_k(\mathbf{x}(\tau)) \\ &\overset{(b)}{\leq} \frac{1}{ct}[\lambda_k(2t) - \lambda_k(t)] \\ &\leq \frac{1}{ct}\|\boldsymbol{\lambda}(2t) - \boldsymbol{\lambda}(t)\| \\ &\overset{(c)}{\leq} \frac{1}{t}\big(\sqrt{1-cL_c}\big)^t \frac{2D_c}{c(\sqrt{1-cL_c})^{T_c}} \end{aligned}$$

where (a) follows from the convexity of $g_k(\cdot)$; (b) follows from Lemma 2.3; and (c) follows from Corollary 2.3. □

The next theorem summarizes both Theorem 2.7 and Theorem 2.8.

Theorem 2.9. *Consider the strongly convex program* (2.1)-(2.3) *satisfying Assumptions 2.1-2.2 and 2.4. If* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then for all* $t \geq T_c$,

$$\begin{aligned} f(\tilde{\mathbf{x}}(2t)) &\leq f(\mathbf{x}^*) + \frac{1}{t}\big(\sqrt{1-cL_c}\big)^t \eta_c, \\ g_k(\widetilde{\mathbf{x}}(2t)) &\leq \frac{1}{t}\big(\sqrt{1-cL_c}\big)^t \frac{2D_c}{c(\sqrt{1-cL_c})^{T_c}}, \forall k \in \{1,2,\ldots,m\}, \end{aligned}$$

where $\eta_c = \frac{2D_c^2}{(\sqrt{1-cL_c})^{2T_c}} + \frac{2D_c\left(\sqrt{\|\boldsymbol{\lambda}(0)\|^2+\|\boldsymbol{\lambda}^*\|^2}+\|\boldsymbol{\lambda}^*\|\right)}{(\sqrt{1-cL_c})^{T_c}}$ *is a fixed constant and* T_c *is defined in* (2.14). *In summary, if* $c \leq \frac{\sigma}{\beta^2}$ *in Algorithm 1.1, then* $\widetilde{\mathbf{x}}(t)$ *ensures error decays like* $O\big(\frac{1}{t}(1-cL_c)^{t/2}\big)$ *and provides an* ϵ*-approximate solution with convergence time* $O(\log(\frac{1}{\epsilon}))$.

Under Assumptions 2.1-2.2 and 2.4, Theorem 2.9 shows that if $c \leq \frac{\sigma}{\beta^2}$, then $\widetilde{\mathbf{x}}(t)$ provides an ϵ-approximate solution with convergence time $O(\log(\frac{1}{\epsilon}))$. Since $L_q = \frac{L_c}{2}$ by Lemma 2.10 and note that $\sqrt{1-cL_c} \leq \frac{1}{\sqrt{1+2cL_q}}$, the geometric contraction modulus shown in Theorem 2.9 under Assumption 2.4 is smaller than the geometric contraction modulus shown in Theorem 2.6 under Assumption 2.3.

2.3.4 Discussion

Practical Implementation

Assumptions 2.3 and 2.4 in general are difficult to verify. However, we note that to ensure $\widetilde{\mathbf{x}}(t)$ provides the better $O(\log(\frac{1}{\epsilon}))$ convergence time, we only require $c \leq \frac{\sigma}{\beta^2}$, which is independent of the parameters in Assumptions 2.3 or 2.4. Namely, in practice, we can blindly apply Algorithm 1.1 to the problem (2.1)-(2.3) with no need to verify Assumption 2.3 or 2.4. If the problem (2.1)-(2.3) happens to satisfy Assumptions 2.3 or 2.4, then $\widetilde{\mathbf{x}}(t)$ enjoys the faster convergence time $O(\log(\frac{1}{\epsilon}))$. If not, then $\widetilde{\mathbf{x}}(t)$ or $\overline{\mathbf{x}}(t)$ at least have convergence time $O(\frac{1}{\epsilon})$.

Local Assumption and Local Geometric Convergence

Since Assumption 2.3 only requires the "quadratic" property to be satisfied in a local radius D_q around $\boldsymbol{\lambda}^*$, the error of Algorithm 1.1 starts to decay like $O\Big(\frac{1}{t}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t\Big)$ only after $\lambda(t)$ arrives at the D_q local radius after T_q iterations, where T_q is independent of the approximation requirement ϵ and hence is order $O(1)$. Thus, Algorithm 1.1 provides an ϵ-approximate solution with convergence time $O(\log(\frac{1}{\epsilon})$. However, it is possible that T_q is relatively large if D_q is small.

In fact, $T_q > 0$ because Assumption 2.3 only requires the dual function to have the "quadratic" property in a local radius. Thus, the theory developed in this section can deal with a large class of problems. On the other hand, if the dual function has the "quadratic" property globally, i.e., for all $\boldsymbol{\lambda} \geq \mathbf{0}$, then $T_q = 0$ and the error of Algorithm 1.1 decays like $O\Big(\frac{1}{t}\big(\frac{1}{\sqrt{1+2cL_q}}\big)^t\Big), \forall t \geq 1$.

A similar tradeoff holds with respect to Assumption 2.4.

2.4 Applications

2.4.1 Strongly Convex Programs Satisfying Non-Degenerate Constraint Qualifications

Theorem 2.10. *Consider the following strongly convex program:*

$$\min \quad f(\mathbf{x}) \tag{2.16}$$

$$s.t. \quad g_k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \ldots, m\} \tag{2.17}$$

$$\mathbf{x} \in \mathbb{R}^n \tag{2.18}$$

where $f(\mathbf{x})$ is a second-order continuously differentiable and strongly convex function; $g_k(\mathbf{x}), \forall k \in \{1, 2, \ldots, m\}$ are Lipschitz continuous, second-order continuously differentiable and convex functions. Let $\mathbf{x}^*$ be the unique solution to this strongly convex program.

1. *Let $\mathcal{K} \subseteq \{1, 2, \ldots, m\}$ be the set of active constraints, i.e., $g_k(\mathbf{x}^*) = 0, \forall k \in \mathcal{K}$, and denote the vector composed by $g_k(\mathbf{x}), k \in \mathcal{K}$ as $\mathbf{g}_\mathcal{K}$. If $\mathbf{g}(\mathbf{x})$ has a bounded Jacobian and rank$(\nabla_\mathbf{x} \mathbf{g}_\mathcal{K}(\mathbf{x}^*)^\mathsf{T}) = |\mathcal{K}|$, then Assumptions 2.1-2.3 hold for this problem.*

2. *If $\mathbf{g}(\mathbf{x})$ has a bounded Jacobian and rank$(\nabla_\mathbf{x} \mathbf{g}(\mathbf{x}^*)^\mathsf{T}) = m$, then Assumptions 2.1-2.4 hold for this problem.*

Proof. See Section 2.7.5. □

Corollary 2.4. *Consider the following strongly convex program with linear inequality constraints:*

$$\min \quad f(\mathbf{x}) \tag{2.19}$$

$$s.t. \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{2.20}$$

where $f(\mathbf{x})$ is second-order continuously differentiable and strongly convex function; and $\mathbf{A}$ is an $m \times n$ matrix.

1. *Let $\mathbf{x}^*$ be the optimal solution. Assume $\mathbf{A}\mathbf{x}^* \leq \mathbf{b}$ has l rows that hold with equality, and let $\mathbf{A}'$ be the $l \times n$ submatrix of $\mathbf{A}$ corresponding to these "active" rows. If rank$(\mathbf{A}') = l$, then Assumptions 2.1-2.3 hold for this problem.*

2. If rank$(\mathbf{A}) = m$*, then Assumptions 2.1-2.4 hold for this problem with* $D_c = \infty$*.*

2.4.2 Network Utility Maximization with Independent Link Capacity Constraints

Consider a network with l links and n flow streams. Let $\{b_1, b_2, \ldots, b_l\}$ be the capacities of each link and $\{x_1, x_2, \ldots, x_n\}$ be the rates of each flow stream. Let $\mathcal{N}(k) \subseteq \{1, 2, \ldots, n\}, 1 \leq k \leq l$ be the set of flow streams that use link k. This problem is to maximize the utility function $\sum_{i=1}^{n} w_i \log(x_i)$ with $w_i > 0, \forall 1 \leq i \leq n$, which represents a measure of network fairness [Kel97], subject to the capacity constraint of each link. This problem is known as the network utility maximization (NUM) problem and can be formulated as follows[3]:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} -w_i \log(x_i) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{N}(k)} x_i \leq b_k, \forall k \in \{1, 2, \ldots, l\} \\ & x_i \geq 0, \forall i \in \{1, 2, \ldots, n\} \end{aligned}$$

Typically, many link capacity constraints in the above formulation are redundant, e.g., if $\mathcal{N}(k_1) = \mathcal{N}(k_2)$ and $b_{k_1} \leq b_{k_2}$, then the capacity constraint of the k_2-th link is redundant. Assume that redundant link capacity constraints are eliminated and the remaining links are reindexed. The above formulation can be rewritten as follows:

$$\min \quad \sum_{i=1}^{n} -w_i \log(x_i) \tag{2.21}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{2.22}$$

$$\mathbf{x} \geq \mathbf{0} \tag{2.23}$$

where $w_i > 0, \forall 1 \leq i \leq n$; $\mathbf{A} = [\mathbf{a}_1, \cdots, \mathbf{a}_n]$ is a 0-1 matrix of size $m \times n$ such that $a_{ij} = 1$ if and only if flow x_j uses link i; and $\mathbf{b} > \mathbf{0}$.

Note that problem (2.21)-(2.23) satisfies Assumptions 2.1 and 2.2. By the results from Section

[3]In this paper, the NUM problem is always formulated as a minimization problem. Without loss of optimality, we define $\log(0) = -\infty$ and hence $\log(\cdot)$ is defined over $\mathbb{R}_+$. Or alternatively, we can replace the non-negative rate constraints with $x_i \geq x_i^{\min}, \forall i \in \{1, 2, \ldots, n\}$ where $x_i^{\min}, \forall i \in \{1, 2, \ldots, n\}$ are sufficiently small positive numbers.

2.2, $\overline{\mathbf{x}}(t)$ has an $O(\frac{1}{\epsilon})$ convergence time for this problem. The next theorem provides sufficient conditions such that $\widetilde{\mathbf{x}}(t)$ can have better convergence time $O(\log(\frac{1}{\epsilon}))$.

Theorem 2.11. *The network utility maximization problem* (2.21)-(2.23) *has the following properties:*

1. *Let* $b^{\max} = \max_{1\leq i\leq n} b_i$ *and* $\mathbf{x}^{\max} > \mathbf{0}$ *such that* $x_i^{\max} > b^{\max}, \forall i \in \{1,\ldots,n\}$. *The network utility maximization problem* (2.21)-(2.23) *is equivalent to the following problem*

$$\min \quad \sum_{i=1}^{n} -w_i \log(x_i) \tag{2.24}$$

$$s.t. \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{2.25}$$

$$\mathbf{0} \leq \mathbf{x} \leq \mathbf{x}^{\max} \tag{2.26}$$

2. *Let* $\mathbf{x}^*$ *be the optimal solution. Assume* $\mathbf{A}\mathbf{x}^* \leq \mathbf{b}$ *has* m' *rows that hold with equality, and let* $\mathbf{A}'$ *be the* $m' \times n$ *submatrix of* $\mathbf{A}$ *corresponding to these "active" rows. If rank*$(\mathbf{A}') = m'$, *then Assumptions 2.1-2.3 hold for this problem.*

3. *If rank*$(\mathbf{A}) = m$, *then Assumptions 2.1-2.4 hold for this problem.*

Proof. See Section 2.7.6. □

Remark 2.4. *Theorem 2.11 and Corollary 2.4 complement each other. If rank*$(\mathbf{A}) = m$, *we can apply Theorem 2.11 to problem* (2.21)-(2.23). *However, to apply Corollary 2.4, we require rank*$(\mathbf{B}) = m + n$, *where* $\mathbf{B} = \begin{bmatrix} \mathbf{A} \\ \mathbf{I}_n \end{bmatrix}$. *This is always false since the size of* $\mathbf{A}'$ *is* $(m+n) \times n$. *Thus, Corollary 2.4 can not be applied to problem* (2.21)-(2.23) *even if rank*$(\mathbf{A}) = m$. *On the other hand, Corollary 2.4 considers general utilities while Theorem 2.11 is restricted to the utility* $\sum_{i=1}^{n} -w_i \log(x_i)$.

Now we give an example of network utility maximization such that Assumption 2.3 is not satisfied. Consider the problem (2.21)-(2.23) with $\mathbf{w} = [1,1,1,1]^\mathsf{T}$,

$$\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4] = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

and $\mathbf{b} = [3, 7, 2, 8]^\mathsf{T}$. Note that $\text{rank}(\mathbf{A}) = 3 < m$; and $\boldsymbol{\mu}^\mathsf{T}\mathbf{A} = [0, 0, 0, 0]$ and $\boldsymbol{\mu}^\mathsf{T}\mathbf{b} = 0$ if $\boldsymbol{\mu} = [1, 1, -1, -1]^\mathsf{T}$.

It can be checked that the optimal solution to this NUM problem is $[x_1^*, x_2^*, x_3^*, x_4^*]^\mathsf{T} = [0.8553, 2.1447, 1.1447, 5.8553]^\mathsf{T}$. Note that all capacity constraints are tight and $[\lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*]^\mathsf{T} = [0.3858, 0.0903, 0.7833, 0.0805]^\mathsf{T}$ is the optimal dual variable that attains strong duality.

Next, we show that $q(\boldsymbol{\lambda})$ is not locally quadratic at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$ by contradiction. Assume that there exist $D_q > 0$ and $L_q > 0$ such that $q(\boldsymbol{\lambda}) \leq q(\boldsymbol{\lambda}^*) - L_q\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2$ for any $\boldsymbol{\lambda} \in \mathbb{R}_+^m$ and $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_q$. Put $\boldsymbol{\lambda} = \boldsymbol{\lambda}^* + t\boldsymbol{\mu}$ with $|t|$ sufficiently small such that $\boldsymbol{\lambda}^* + t\boldsymbol{\mu} \in \mathbb{R}_+^m$ and $\|\boldsymbol{\lambda}^* + t\boldsymbol{\mu} - \boldsymbol{\lambda}^*\| < D_q$. Note that by (2.32) and (2.33), we have

$$\boldsymbol{\mu}^\mathsf{T}\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = \sum_{i=1}^{n} \frac{\boldsymbol{\mu}^\mathsf{T}\mathbf{a}_i}{[\boldsymbol{\lambda}^*]^\mathsf{T}\mathbf{a}_i} + \boldsymbol{\mu}^\mathsf{T}\mathbf{b} = 0, \tag{2.27}$$

$$\boldsymbol{\mu}^\mathsf{T}\nabla_{\boldsymbol{\lambda}}^2 q(\boldsymbol{\lambda}^*)\boldsymbol{\mu} = 0. \tag{2.28}$$

Thus, we have

$$\begin{aligned} & q(\boldsymbol{\lambda}^* + t\boldsymbol{\mu}) \\ \overset{(a)}{=} & q(\boldsymbol{\lambda}^*) + t\boldsymbol{\mu}^\mathsf{T}\nabla_{\boldsymbol{\lambda}}\tilde{q}(\boldsymbol{\lambda}^*) + t^2\boldsymbol{\mu}^\mathsf{T}\nabla_{\boldsymbol{\lambda}}^2\tilde{q}(\boldsymbol{\lambda}^*)\boldsymbol{\mu} + o(t^2\|\boldsymbol{\mu}\|^2) \\ \overset{(b)}{=} & q(\boldsymbol{\lambda}^*) + o(t^2\|\boldsymbol{\mu}\|^2) \end{aligned}$$

where (a) follows from the second-order Taylor's expansion and (b) follows from equations (2.27) and (2.28). By definition of $o(t^2\|\boldsymbol{\mu}\|^2)$, there exists $\delta > 0$ such that $\frac{|o(t^2\|\boldsymbol{\mu}\|^2)|}{\|t\boldsymbol{\mu}\|^2} < L_q, \forall t \in (-\delta, \delta)$, i.e., $o(t^2\|\boldsymbol{\mu}\|^2) > -L_q\|t\boldsymbol{\mu}\|^2, \forall t \in (-\delta, \delta)$. This implies $q(\boldsymbol{\lambda}^* + t\boldsymbol{\mu}) = q(\boldsymbol{\lambda}^*) + o(t^2\|\boldsymbol{\mu}\|^2) > q(\boldsymbol{\lambda}^*) - L_q\|t\boldsymbol{\mu}\|^2$. A contradiction! Thus, $q(\boldsymbol{\lambda})$ is not locally quadratic at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$.

In view of the above example, the sufficient condition in Part 2 of Theorem 2.11 for Assumption 2.3 is sharp.

2.5 Numerical Experiment

2.5.1 Network Utility Maximization

Consider the simple NUM problem described in Figure 2.1. Let x_1, x_2 and x_3 be the data rates of stream 1, 2 and 3 and let the network utility be minimizing $-\log(x_1) - 2\log(x_2) - 3\log(x_3)$.

It can be checked that capacity constraints other than $x_1 + x_2 + x_3 \leq 10, x_1 + x_2 \leq 8$, and $x_2 + x_3 \leq 8$ are redundant. By Theorem 2.11, the NUM problem can be formulated as follows:

$$\begin{aligned} \min \quad & -\log(x_1) - 2\log(x_2) - 3\log(x_3) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{0} \leq \mathbf{x} \leq \mathbf{x}^{\max} \end{aligned}$$

where $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 10 \\ 8 \\ 8 \end{bmatrix}$ and $\mathbf{x}^{\max} = \begin{bmatrix} 11 \\ 11 \\ 11 \end{bmatrix}$. The optimal solution to this NUM problem is $x_1^* = 2, x_2^* = 3.2, x_3^* = 4.8$ and the optimal value is -7.7253. Note that the second capacity constraint $x_1 + x_2 \leq 8$ is loose and the other two capacity constraints are tight.

Since the objective function is decomposable, the dual subgradient method can yield a distributed solution. This is why the dual subgradient method is widely used to solve NUM problems [LL99]. It can be checked that the objective function is strongly convex with modulus $\sigma = \frac{2}{121}$ on $\mathcal{X} = \{\mathbf{0} \leq \mathbf{x} \leq \mathbf{x}^{\max}\}$ and $\mathbf{g}$ is Lipschitz continuous with modulus $\beta \leq \sqrt{6}$ on $\mathcal{X}$. Figure 2.2 verifies the convergence of $\overline{\mathbf{x}}(t)$ with $c = \frac{\sigma}{\beta^2} = 1/363$ and $\lambda_1(0) = \lambda_2(0) = \lambda_3(0) = 0$. Since $\lambda_1(0) = \lambda_2(0) = \lambda_3(0) = 0$, by Theorem 2.1, we know $f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*), \forall t > 0$, which is also verified in Figure 2.2. To verify the convergence time of constraint violations, Figure 2.3 plots $g_1(\overline{\mathbf{x}}(t))$, $g_2(\overline{\mathbf{x}}(t))$, $g_3(\overline{\mathbf{x}}(t))$ and $1/t$ with both x-axis and y-axis in $\log_{10}$ scales. As observed in Figure 2.3, the curves of $g_1(\overline{\mathbf{x}}(t))$ and $g_3(\overline{\mathbf{x}}(t))$ are parallel to the curve of $1/t$ for large t. Note that $g_1(\overline{\mathbf{x}}(t)) \leq 0$ is satisfied early because this constraint is loose. Figure 2.3 verifies the convergence time of $\overline{\mathbf{x}}(t)$ in Theorem 2.3 by showing that error decays like $O(\frac{1}{t})$ and suggests that the convergence time is actually $\Theta(\frac{1}{\epsilon})$ for this NUM problem.

Note that $\text{rank}(\mathbf{A}) = 3$. By Theorem 2.11, this NUM problem satisfies Assumptions 2.1-2.4. Apply Algorithm 1.1 with $c = \frac{\sigma}{\beta^2} = 1/363$ and $\lambda_1(0) = \lambda_2(0) = \lambda_3(0) = 0$ to this NUM problem. Figure 2.4 verifies the convergence of the objective and constraint functions for $\widetilde{\mathbf{x}}(t)$. Figure 2.5 verifies the results in Theorem 2.11 that the convergence time of $\widetilde{\mathbf{x}}(t)$ is $O(\log(\frac{1}{\epsilon}))$ by showing that error decays like $O(\frac{1}{t}0.998^t)$. If we compare Figure 2.5 and Figure 2.3, we can observe that $\widetilde{\mathbf{x}}(t)$ converges much faster than $\overline{\mathbf{x}}(t)$.

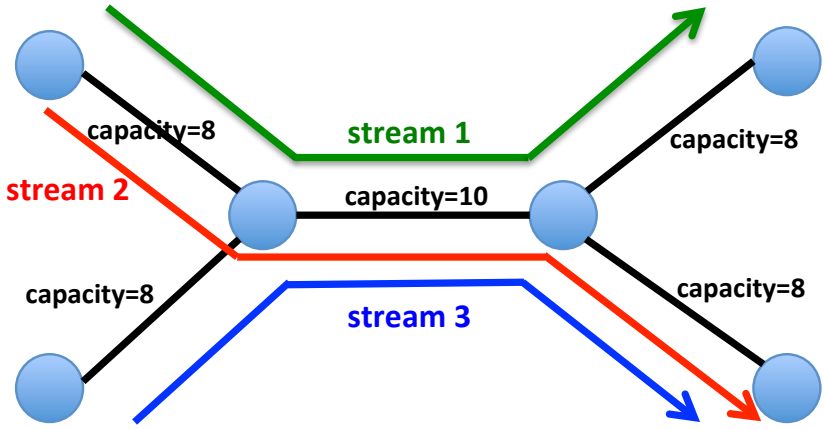

Figure 2.1: A simple NUM problem with 3 flow streams

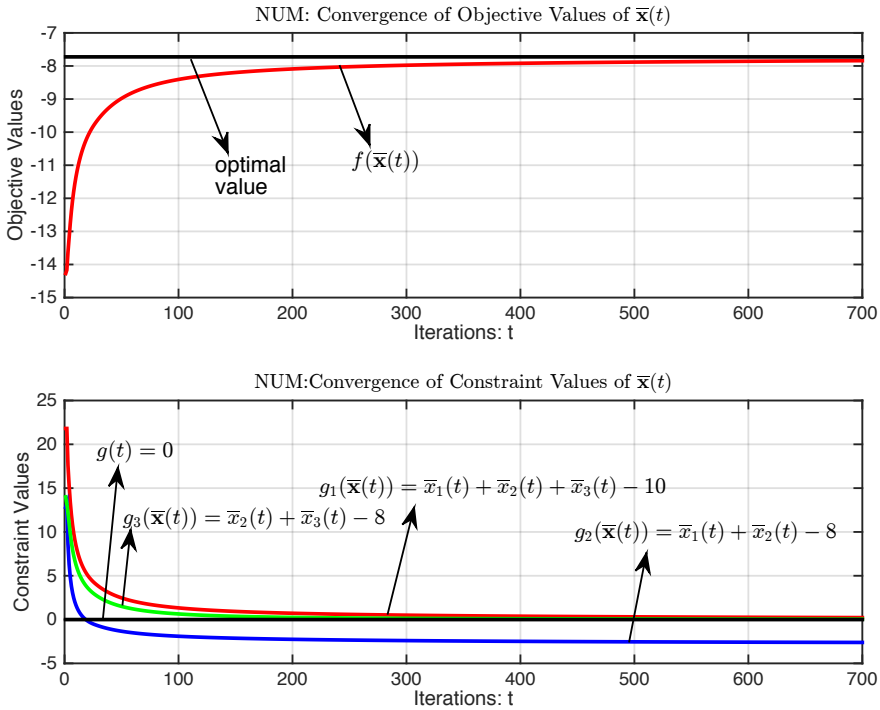

Figure 2.2: Convergence of $\overline{\mathbf{x}}(t)$ from Algorithm 1.1 for a NUM problem.

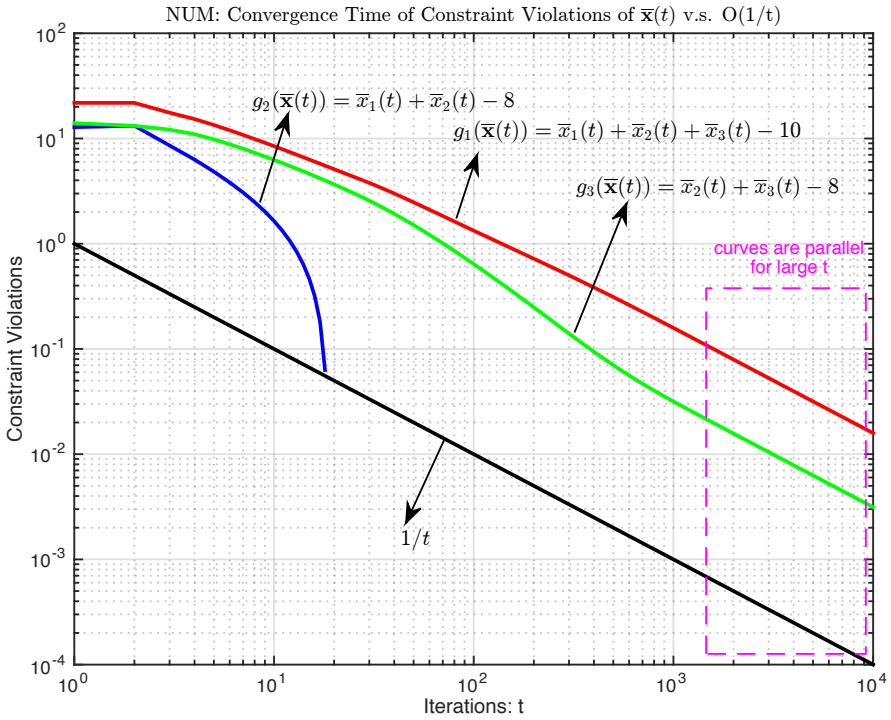

Figure 2.3: Convergence time of $\overline{\mathbf{x}}(t)$ from Algorithm 1.1 for a NUM problem.

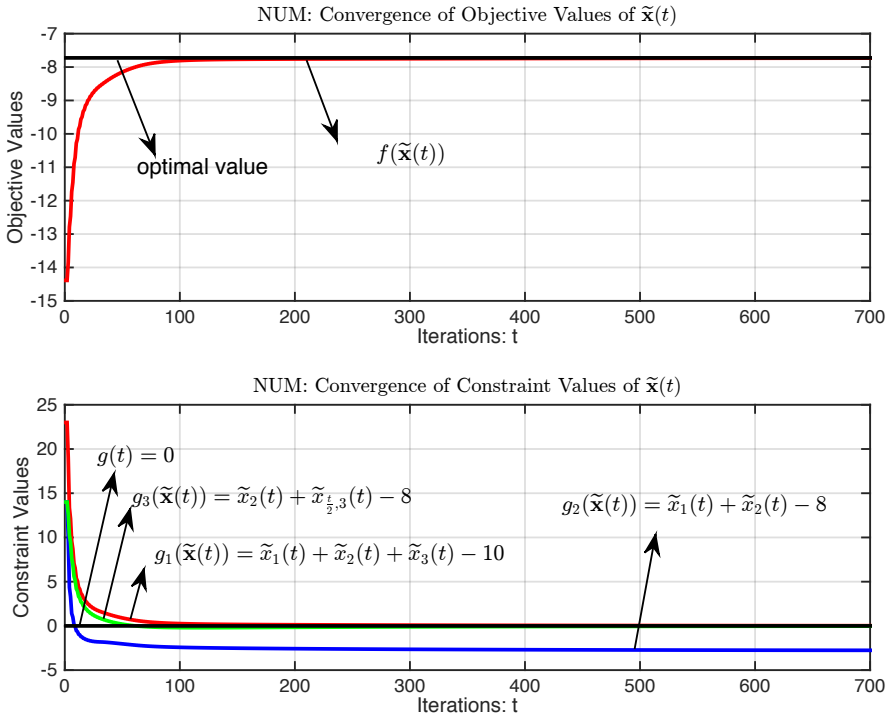

Figure 2.4: Convergence of $\widetilde{\mathbf{x}}(t)$ from Algorithm 1.1 for a NUM problem.

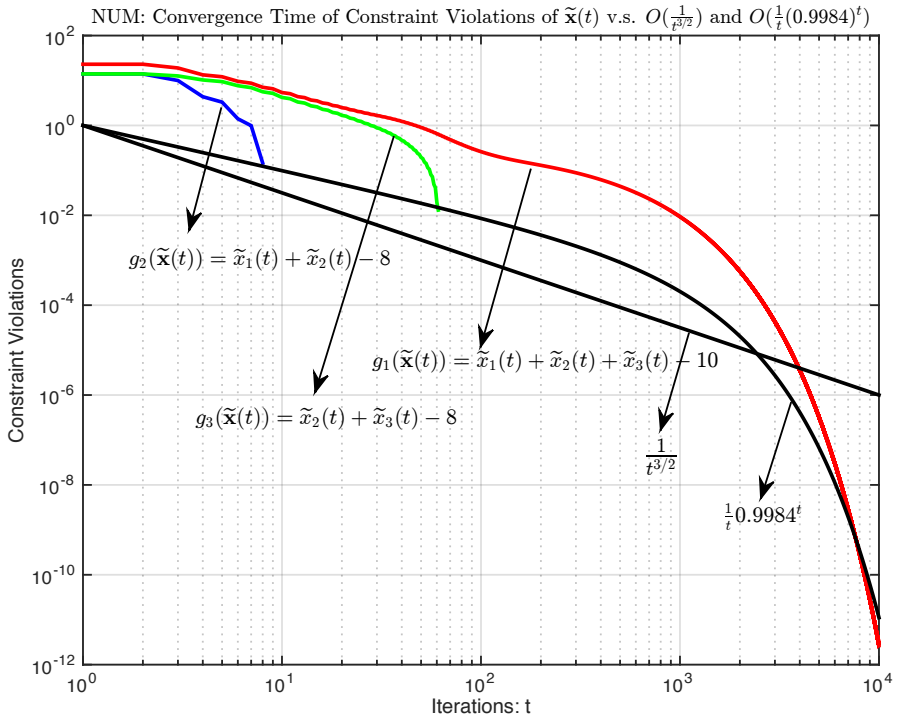

Figure 2.5: Convergence time of $\widetilde{\mathbf{x}}(t)$ from Algorithm 1.1 for a NUM problem.

2.5.2 Linear Constrained Quadratic Program

Consider the following quadratic program (QP)

$$\begin{aligned} \min \quad & \mathbf{x}^\mathsf{T}\mathbf{P}\mathbf{x} + \mathbf{c}^\mathsf{T}\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{aligned}$$

where $\mathbf{P} = \begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix}$, $\mathbf{c} = [1,1]^\mathsf{T}$, $\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ and $\mathbf{b} = [-2,-1]^\mathsf{T}$. The optimal solution to this quadratic program is $\mathbf{x}^* = [-1,-1]^\mathsf{T}$ and the optimal value is 8.

If P is a diagonal matrix, the dual subgradient method can yield a distributed solution. It can be checked that the objective function is strongly convex with modulus $\sigma = 0.34$ and each row of the linear inequality constraint is Lipschitz continuous with modulus $\zeta = \sqrt{2}$. Figure 2.6 verifies the convergence of $\overline{\mathbf{x}}(t)$ for the objective and constraint functions yielded by Algorithm 1.1 with $c = \frac{\sigma}{2\zeta^2} = 0.34/4$, $\lambda_1(0) = 0$ and $\lambda_2(0) = 0$. Figure 2.7 verifies the convergence time

of $\overline{\mathbf{x}}(t)$ proven in Theorem 2.3 by showing that error decays like $O(\frac{1}{t})$ and suggests that the convergence time is actually $\Theta(\frac{1}{\epsilon})$ for this quadratic program.

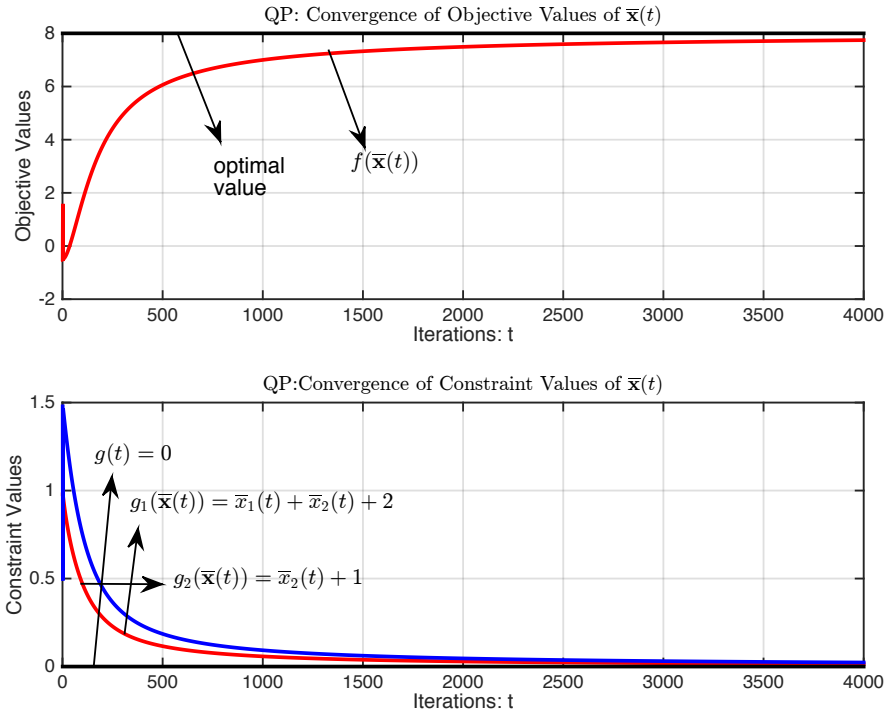

Figure 2.6: Convergence of $\overline{\mathbf{x}}(t)$ from Algorithm 1.1for a quadratic program.

Note that $\text{rank}(A) = 2$. By Corollary 2.4 this quadratic program satisfies Assumptions 2.1-2.4. Apply Algorithm 1.1 with $c = \frac{\sigma}{2\zeta^2} = 0.34/4$ and $\lambda_1(0) = \lambda_2(0) = \lambda_3(0) = 0$ to this quadratic program. Figure 2.8 verifies the convergence of the objective and constraint functions. Figure 2.9 verifies the results in Corollary 2.4 that the convergence time of $\widetilde{\mathbf{x}}(t)$ is $O(\log(\frac{1}{\epsilon}))$ by showing that error decays like $O(\frac{1}{t}0.9935^t)$. If we compare Figure 2.9 and Figure 2.7, we can observe that Algorithm $\widetilde{\mathbf{x}}(t)$ converges much faster than $\overline{\mathbf{x}}(t)$.

2.5.3 Large Scale Quadratic Program

Consider quadratic program $\min_{\mathbf{x}\in\mathbb{R}^N}\{\mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x} + \mathbf{d}^\mathsf{T}\mathbf{x} : \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ where $\mathbf{Q}, \mathbf{A} \in \mathbb{R}^{N\times N}$ and $\mathbf{d}, \mathbf{b} \in \mathbb{R}^N$. $\mathbf{Q} = \mathbf{U}\Sigma\mathbf{U}^H \in \mathbb{R}^{N\times N}$ where $\mathbf{U}$ is the orthonormal basis for a random $N \times N$ zero mean and unit variance normal matrix and Σ is the diagonal matrix with entries from uniform $[1, 3]$. $\mathbf{A}$ is a random $N \times N$ zero mean and unit variance normal matrix. $\mathbf{d}$ and $\mathbf{b}$ are random vectors with entries from uniform $[0, 1]$. In a PC with a 4 core 2.7GHz Intel i7 CPU

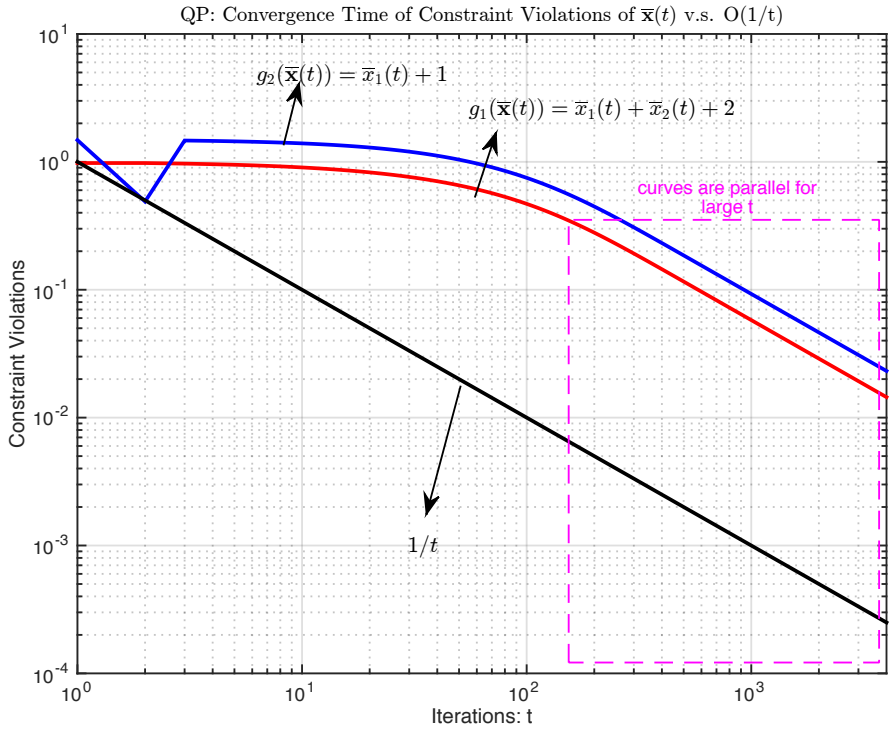

Figure 2.7: Convergence time of $\overline{\mathbf{x}}(t)$ from Algorithm 1.1 for a quadratic program.

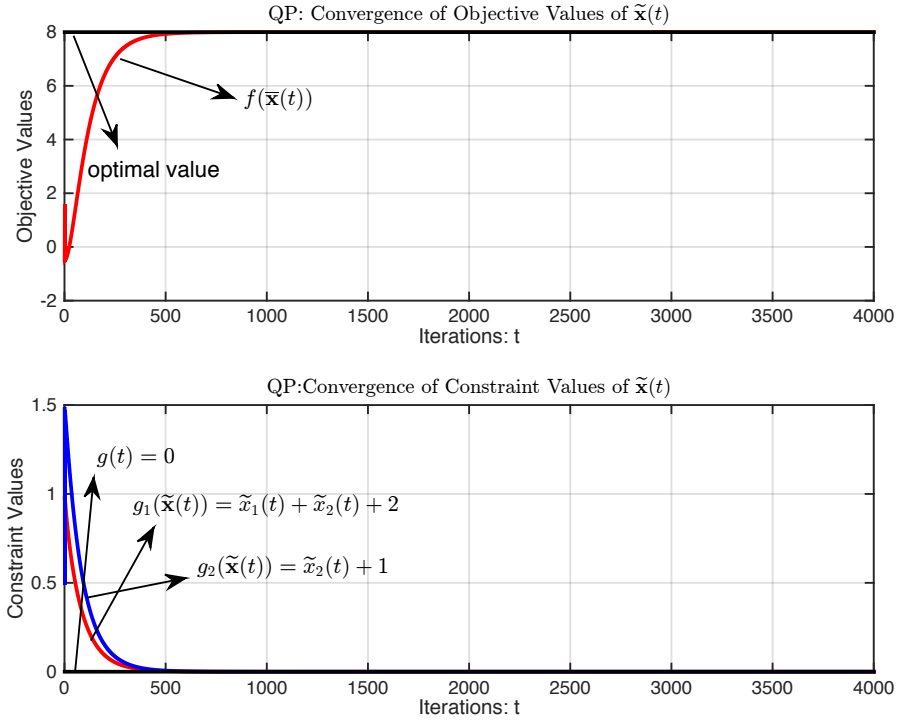

Figure 2.8: Convergence of $\widetilde{\mathbf{x}}(t)$ from Algorithm 1.1 for a quadratic program.

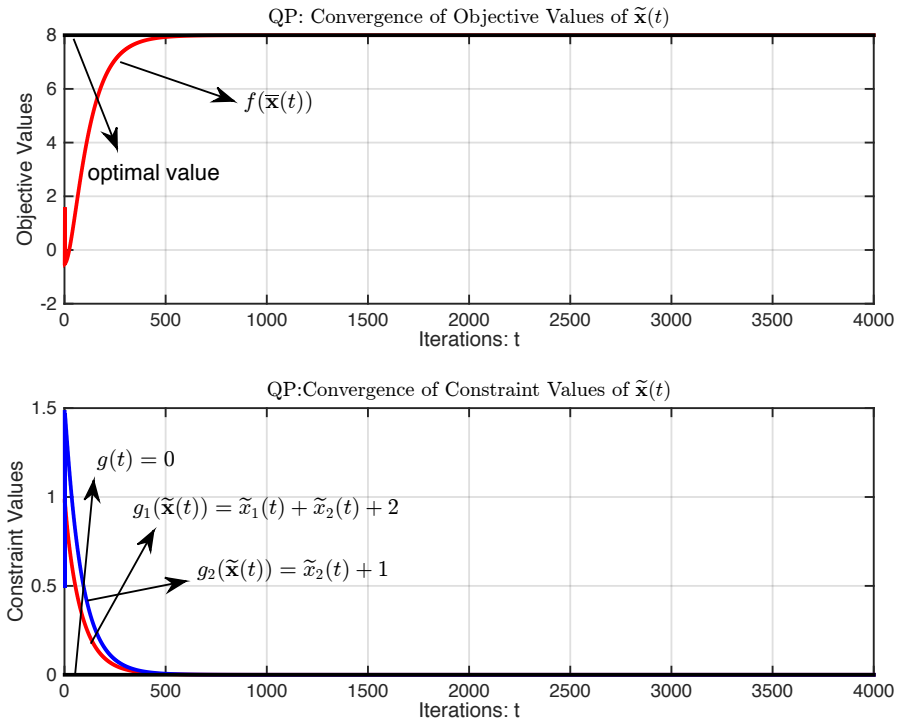

Figure 2.9: Convergence time of $\widetilde{\mathbf{x}}(t)$ from Algorithm 1.1 for a quadratic program.

and 16GB Memory, we run both Algorithm 1.1 and quadprog from Matlab, which by default is using the interior point method, over randomly generated large scale quadratic programs with $N = 400, 600, 800, 1000$ and 1200. For different problem size N, the running time is the average over 100 random quadratic programs and is plotted in Figure 2.10. To solve these large scale quadratic programs, the dual subgradient method has updates $\mathbf{x}(t) = -\frac{1}{2}\mathbf{Q}^{-1}[\boldsymbol{\lambda}^{\mathsf{T}}(t)\mathbf{A} + \mathbf{d}]$ and $\boldsymbol{\lambda}(t+1) = \max\{\boldsymbol{\lambda}(t) + c[\mathbf{A}\mathbf{x}(t) - b], \mathbf{0}\}$ at each iteration t. Note that we only need to compute the inverse of large matrix $\mathbf{Q}$ once and then use it during all iterations. In our numerical simulations, Algorithm 1.1 is terminated when the error (both objective violations and constraint violations) of $\widetilde{\mathbf{x}}(t)$ is less than 1e-5.

2.6 Chapter Summary

This chapter studies the convergence time of the dual subgradient method strongly convex programs. This chapter shows that the convergence time of the dual subgradient method with simple running averages for general strongly convex programs is $O(\frac{1}{\epsilon})$. This chapter also considers

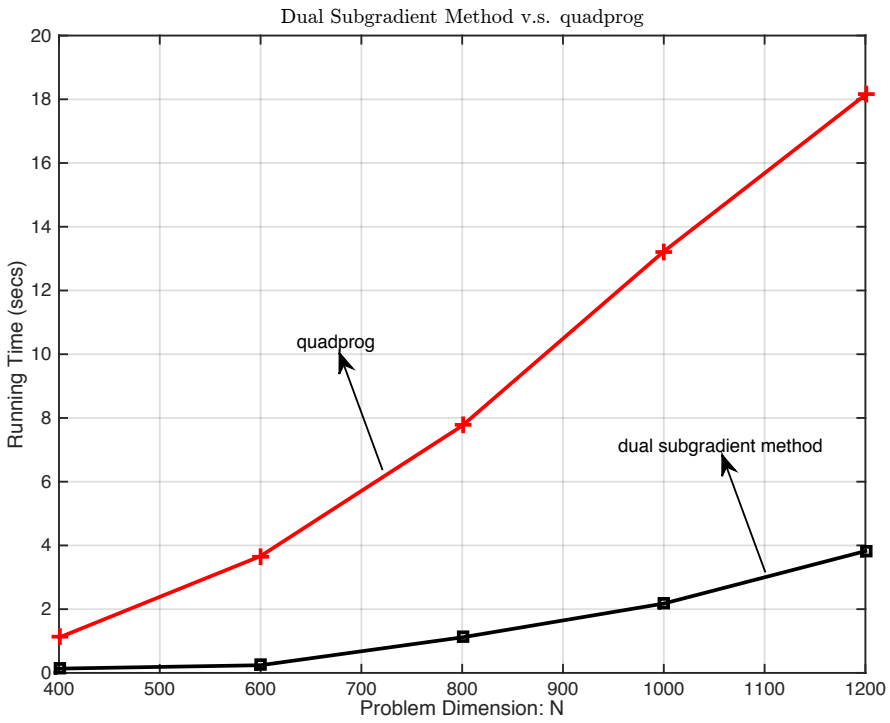

Figure 2.10: The average running time for large scale quadratic programs.

a variation of the primal averages, called the sliding running averages, and shows that if the dual function is locally quadratic then the convergence time is $O(\log(\frac{1}{\epsilon}))$.

2.7 Supplement to this Chapter

2.7.1 Proof of Lemma 2.6

Note that $\lambda_k(t+1) = \max\{\lambda_k(t) + cg_k(\mathbf{x}(t)), 0\}, \forall k \in \{1, 2, \ldots, m\}$ can be interpreted as the $\boldsymbol{\lambda}(t+1) = \mathcal{P}_{\mathbb{R}^m_+}\left[\boldsymbol{\lambda}(t) + c\mathbf{g}(\mathbf{x}(t))\right]$ where $\mathcal{P}_{\mathbb{R}^m_+}[\cdot]$ is the projection onto $\mathbb{R}^m_+$. As observed before, the dynamic of $\boldsymbol{\lambda}(t)$ can be interpreted as the projected gradient method with step size c to solve $\max_{\boldsymbol{\lambda}\in\mathbb{R}^m_+}\{q(\boldsymbol{\lambda})\}$. Thus, the proof given below is essentially the same as the convergence time proof of the projected gradient method for set constrained smooth optimization in [Nes04].

Fact 2.1. $\boldsymbol{\lambda}(t+1) = \text{argmax}_{\boldsymbol{\lambda}\in\mathbb{R}^m_+}\left\{q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)] - \frac{1}{2c}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)\|^2\right\}, \forall t \geq 0.$

Proof. Fix $t \geq 0$,

$$\begin{aligned}
\boldsymbol{\lambda}(t+1) =& \mathcal{P}_{\mathbb{R}_+^m}\left\{\boldsymbol{\lambda}(t) + c\mathbf{g}(\mathbf{x}(t))\right\} \\
=& \operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathbb{R}_+^m}\left\{\|\boldsymbol{\lambda} - [\boldsymbol{\lambda}(t) + c\mathbf{g}(\mathbf{x}(t))]\|^2\right\} \\
=& \operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathbb{R}_+^m}\left\{c^2\|\mathbf{g}(\mathbf{x}(t))\|^2 - 2c[\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)] + \|\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)\|^2\right\} \\
\overset{(a)}{=}& \operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathbb{R}_+^m}\left\{-q(\boldsymbol{\lambda}(t)) - [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)] + \frac{1}{2c}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)\|^2\right\} \\
=& \operatorname*{argmax}_{\boldsymbol{\lambda}\in\mathbb{R}_+^m}\left\{q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)] - \frac{1}{2c}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)\|^2\right\}
\end{aligned}$$

where (a) follows because the minimizer is unchanged when we remove constant term $c^2\|\mathbf{g}(\mathbf{x}(t))\|^2$, divide by factor $2c$, and add constant term $-q(\boldsymbol{\lambda}(t))$ in the objective function. □

Recall that $q(\boldsymbol{\lambda})$ is smooth with modulus $\gamma = \frac{\beta^2}{\sigma}$ by Lemma 2.5.

Fact 2.2. *If $c \leq \frac{1}{\gamma} = \frac{\sigma}{\beta^2}$, then $q(\boldsymbol{\lambda}(t+1)) \geq q(\boldsymbol{\lambda}(t)), \forall t \geq 0$.*

Proof. Fix $t \geq 0$,

$$\begin{aligned}
q(\boldsymbol{\lambda}(t+1)) \overset{(a)}{\geq}& q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
\overset{(b)}{\geq}& q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)] - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
\overset{(c)}{\geq}& q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(t)] - \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}(t)\|^2 \\
=& q(\boldsymbol{\lambda}(t))
\end{aligned}$$

where (a) follows from Lemma 1.1 and the fact that $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}(t)) = \mathbf{g}(\mathbf{x}(t))$; (b) follows from $c \leq \frac{1}{\gamma}$; and (c) follows form Fact 2.1. □

Fact 2.3. $[\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*] \geq \frac{1}{c}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*], \forall t \geq 0$

Proof. Fix $t \geq 0$. By the projection theorem (Proposition B.11(b) in [Ber99]), we have $\big[\boldsymbol{\lambda}(t+1) - \big(\boldsymbol{\lambda}(t) + c\mathbf{g}(\mathbf{x}(t))\big)\big]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*] \leq 0$. Thus, $[\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*] \geq \frac{1}{c}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)]^\mathsf{T}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*]$. □

Fact 2.4. *If $c \leq \frac{1}{\gamma} = \frac{\sigma}{\beta^2}$, then $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t+1)) \leq \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2, \forall t \geq 0$.*

Proof. Fix $t \geq 0$,

$$
\begin{aligned}
& q(\boldsymbol{\lambda}(t+1)) \\
\overset{(a)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
= & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^* + \boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
\overset{(b)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 + \frac{1}{c}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)]^{\mathsf{T}}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*] \\
\overset{(c)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
& + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 \\
\overset{(d)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 \\
\overset{(e)}{\geq} & q(\boldsymbol{\lambda}^*) + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2
\end{aligned}
$$

where (a) follows from Lemma 1.1 and the fact that $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}(t)) = \mathbf{g}(\mathbf{x}(t))$; (b) follows from Fact 2.3; (c) follows from the identity $\mathbf{u}^{\mathsf{T}}\mathbf{v} = \frac{1}{2}\|\mathbf{u}\|^2 + \frac{1}{2}\|\mathbf{v}\|^2 - \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|^2, \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^m$; (d) follows from $c \leq \frac{1}{\gamma}$; and (e) follows from the concavity of $q(\cdot)$.

Rearranging terms yields the desired result. □

Fix $c \leq \frac{1}{\gamma}$ and $t > 0$. By Fact 2.4, we have $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(\tau+1)) \leq \frac{1}{2c}\|\boldsymbol{\lambda}(\tau) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(\tau+1) - \boldsymbol{\lambda}^*\|^2, \forall \tau \in \{0, 1, \ldots, t-1\}$. Summing over τ and dividing by fact t yields

$$
\begin{aligned}
\frac{1}{t}\sum_{\tau=0}^{t-1}\left[q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(\tau+1))\right] \leq & \frac{1}{2ct}[\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2 - \|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2] \\
\leq & \frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2
\end{aligned}
$$

Note that $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(\tau+1)), \forall \tau \in \{0, 1, \ldots, t-1\}$ is a decreasing sequence by Fact 2.2. Thus, we have

$$
q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t) \leq \frac{1}{t}\sum_{\tau=0}^{t-1}\left[q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(\tau+1))\right] \leq \frac{1}{2ct}\|\boldsymbol{\lambda}(0) - \boldsymbol{\lambda}^*\|^2.
$$

2.7.2 Proof of Part 2 of Lemma 2.9

This part is essentially a local version of Theorem 12 in [NNG15], which shows that the projected gradient method for set constrained smooth convex optimization converge geometrically if the objective function satisfies a quadratic growth condition.

In this subsection, we provide a simple proof that directly follows from Fact 2.4 and Assumption 2.3. By Fact 2.4, we have

$$q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t+1)) \leq \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2, \forall t \geq 0. \tag{2.29}$$

By part 1, we know $\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq D_q, \forall t \geq T_q$. By Assumption 2.3, we have

$$q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t+1)) \geq L_q\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2, \forall t \geq T_q. \tag{2.30}$$

Combining (2.29) and (2.30) yields

$$(L_q + \frac{1}{2c})\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 \leq \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2, \forall t \geq T_q.$$

This can be written as

$$\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\| \leq \sqrt{\frac{1}{1+2cL_q}}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|, \forall t \geq T_q$$

Thus, this part follows by induction.

2.7.3 Proof of Lemma 2.11

Define $\tilde{h}(\mathbf{x}) = -h(\mathbf{x})$. Then $\tilde{h}$ is smooth with modulus γ and strongly convex with modulus L_c over the set $\mathcal{X}$. By definition of smooth functions, h must be differentiable over set $\mathcal{X}$. By Lemma 1.2, we have

$$\tilde{h}(\mathbf{y}) \geq \tilde{h}(\mathbf{x}) + [\nabla\tilde{h}(\mathbf{x})]^{\mathsf{T}}(\mathbf{y} - \mathbf{x}) + \frac{L_c}{2}\|\mathbf{y} - \mathbf{x}\|^2, \forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$$

By Lemma 1.1,

$$\tilde{h}(\mathbf{y}) \leq \tilde{h}(\mathbf{x}) + [\nabla \tilde{h}(\mathbf{x})]^{\mathsf{T}}(\mathbf{y}-\mathbf{x}) + \frac{\gamma}{2}\|\mathbf{y}-\mathbf{x}\|^2, \forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$$

Since $\mathcal{X}$ is not a singleton, we can choose distinct $\mathbf{x}, \mathbf{y} \in \mathcal{X}$. Combining the above two inequalities yields $L_c \leq \gamma$.

2.7.4 Proof of Part 2 of Lemma 2.12

By the first part of this lemma, $\boldsymbol{\lambda}(t) \in \{\boldsymbol{\lambda} \in \mathbb{R}^m_+ : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}, \forall t \geq T_c$. The remaining part of the proof is essentially a local version of the convergence time proof of the projected gradient method for set constrained smooth and strongly convex optimization [Nes04].

Recall that $q(\boldsymbol{\lambda})$ is smooth with modulus $\gamma = \frac{\beta^2}{\sigma}$ by Lemma 2.5. The next fact is an enhancement of Fact 2.4 using the locally strong concavity of the dual function.

Fact 2.5. *If* $c \leq \frac{1}{\gamma} = \frac{\sigma}{\beta^2}$*, then* $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t+1)) \leq (\frac{1}{2c} - \frac{L_c}{2})\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2, \forall t \geq T_c$.

Proof. Fix $t \geq T_c$,

$$\begin{aligned}
& q(\boldsymbol{\lambda}(t+1)) \\
\overset{(a)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
= & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^* + \boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
\overset{(b)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 + \frac{1}{c}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)]^{\mathsf{T}}[\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*] \\
\overset{(c)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{\gamma}{2}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\|^2 \\
& + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 \\
\overset{(d)}{\geq} & q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 \\
\overset{(e)}{\geq} & q(\boldsymbol{\lambda}^*) + \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 + (\frac{L_c}{2} - \frac{1}{2c})\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2
\end{aligned}$$

where (a) follows from Lemma 1.1 and the fact that $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}(t)) = \mathbf{g}(\mathbf{x}(t))$; (b) follows from Fact 2.3; (c) follows from the identity $\mathbf{u}^{\mathsf{T}}\mathbf{v} = \frac{1}{2}\|\mathbf{u}\|^2 + \frac{1}{2}\|\mathbf{v}\|^2 - \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|^2, \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^m$; (d) follows from $c \leq \frac{1}{\gamma}$; and (e) follows from the fact that $q(\cdot)$ is strongly concave over the set

$\{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}$ such that $q(\boldsymbol{\lambda}^*) \leq q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{L_c}{2}\|\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)\|^2$ by Lemma 1.2[4].

Rearranging terms yields the desired inequality. □

Note that $q(\boldsymbol{\lambda}^*) - q(\boldsymbol{\lambda}(t+1)) \geq 0, \forall t > 0$. Combining with Fact 2.5 yields $(\frac{1}{2c} - \frac{L_c}{2})\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|^2 - \frac{1}{2c}\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\|^2 \geq 0, \forall t \geq T_c$. Recall that $c \leq \frac{1}{\gamma}$ implies that $c \leq \frac{1}{L_c}$ by Lemma 2.11. Thus, we have

$$\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}^*\| \leq \sqrt{1 - cL_c}\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\|, \forall t \geq T_c$$

By induction, we have

$$\|\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*\| \leq \left(\sqrt{1 - cL_c}\right)^{t - T_c}\|\boldsymbol{\lambda}(T_c) - \boldsymbol{\lambda}^*\|, \forall t \geq T_c$$

2.7.5 Proof of Theorem 2.10

Lemma 2.13. *Let $q(\boldsymbol{\lambda}) : \mathbb{R}_+^m \to \mathbb{R}$ be a concave function and $q(\boldsymbol{\lambda})$ be maximized at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^* \geq \mathbf{0}$. Suppose the following conditions are satisfied:*

1. *Suppose $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = \mathbf{d} \leq \mathbf{0}$ and $\lambda_k^* d_k = 0, \forall k \in \{1, \ldots, m\}$. Denote $\mathcal{K} = \{k \in \{1, \ldots, m\} : d_k = 0\}$ and $l = |\mathcal{K}|$.*
2. *Suppose $\nabla^2_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = \mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^\mathsf{T}$ where $\boldsymbol{\Sigma} \prec 0$ is an $n \times n$ negative definite matrix and $\mathbf{U}$ is an $m \times n$ matrix. Let $\mathbf{U}'$ be an $l \times n$ submatrix of $\mathbf{U}$ and be composed by rows with indices in $\mathcal{K}$. Suppose that rank$(\mathbf{U}') = l$.*

Then, there exists $D_q > 0$ and $L_q > 0$ such that $q(\boldsymbol{\lambda}) \leq q(\boldsymbol{\lambda}^*) - L_q\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2$ for any $\boldsymbol{\lambda} \in \mathbb{R}_+^m$ and $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_q$.

Proof. Without loss of generality, assume that $\mathcal{K} = \{1, \ldots, l\}$. Denote $\mathbf{U} = \begin{bmatrix} \mathbf{U}' \\ \mathbf{U}'' \end{bmatrix}$ where $\mathbf{U}''$ is the $(m - l) \times n$ matrix composed by $(l+1)$-th to m-th rows of $\mathbf{U}$. Since $\mathbf{d} \leq \mathbf{0}$, let $\delta = \min_{\{l+1 \leq k \leq m\}}\{|d_k|\}$ such that $d_k \leq -\delta, \forall k \in \{l+1, \ldots, m\}$. For each $\boldsymbol{\lambda}$, we define $\boldsymbol{\mu}$ via

[4]Note that the dual function $q(\boldsymbol{\lambda})$ is differentiable and has gradient $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}(t)) = \mathbf{g}(\mathbf{x}(t))$ by the strong convexity of $f(\mathbf{x})$ and Proposition B.25 in [Ber99]. Applying Lemma 1.2 to $\tilde{q}(\boldsymbol{\lambda}) = -q(\boldsymbol{\lambda})$, which has gradient $\nabla_{\boldsymbol{\lambda}} \tilde{q}(\boldsymbol{\lambda}(t)) = -\mathbf{g}(\mathbf{x}(t))$ and is strongly convex over the set $\{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}$, yields $q(\boldsymbol{\lambda}^*) \leq q(\boldsymbol{\lambda}(t)) + [\mathbf{g}(\mathbf{x}(t))]^\mathsf{T}[\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)] - \frac{L_c}{2}\|\boldsymbol{\lambda}^* - \boldsymbol{\lambda}(t)\|^2$.

$\mu_k = \lambda_k - \lambda_k^*, \forall k \in \{1, \ldots, l\}, \mu_k = 0, \forall k \in \{l+1, \ldots, m\}$ and $\boldsymbol{\nu}$ via $\nu_k = 0, \forall k \in \{1, \ldots, l\}, \nu_k = \lambda_k - \lambda_k^*, \forall k \in \{l+1, \ldots, m\}$ such that $\boldsymbol{\lambda} - \boldsymbol{\lambda}^* = \boldsymbol{\mu} + \boldsymbol{\nu}$ and $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2 = \|\boldsymbol{\mu}\|^2 + \|\boldsymbol{\nu}\|^2$. Define l-dimension vector $\boldsymbol{\mu}' = [\mu_1, \ldots, \mu_l]$. Note that $\|\boldsymbol{\mu}'\| = \|\boldsymbol{\mu}\|$. By the first condition, $d_k \neq 0, \forall k \in \{l+1, \ldots, m\}$ implies that $\lambda_k^* = 0, \forall k \in \{l+1, \ldots, m\}$, which together with the fact that $\boldsymbol{\lambda} \geq \mathbf{0}$ implies that $\boldsymbol{\nu} \geq \mathbf{0}$. If $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|$ is sufficiently small, we have

$$
\begin{aligned}
q(\boldsymbol{\lambda}) &\overset{(a)}{=} q(\boldsymbol{\lambda}^*) + (\boldsymbol{\lambda} - \boldsymbol{\lambda}^*)^\mathsf{T} \nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) + (\boldsymbol{\lambda} - \boldsymbol{\lambda}^*)^\mathsf{T} \nabla_{\boldsymbol{\lambda}}^2 q(\boldsymbol{\lambda}^*)(\boldsymbol{\lambda} - \boldsymbol{\lambda}^*) + o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2) \\
&= q(\boldsymbol{\lambda}^*) + \sum_{k=1}^{l} \mu_k d_k + \sum_{k=l+1}^{m} \nu_k d_k + \boldsymbol{\mu}^\mathsf{T} \mathbf{U} \boldsymbol{\Sigma} \mathbf{U}^\mathsf{T} \boldsymbol{\mu}^\mathsf{T} + \boldsymbol{\nu}^\mathsf{T} \mathbf{U} \boldsymbol{\Sigma} \mathbf{U}^\mathsf{T} \boldsymbol{\nu}^\mathsf{T} + o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2) \\
&\overset{(b)}{\leq} q(\boldsymbol{\lambda}^*) - \sum_{k=l+1}^{m} \nu_k \delta + \boldsymbol{\mu}'^{,\mathsf{T}} \mathbf{U}' \boldsymbol{\Sigma} \mathbf{U}'^{,\mathsf{T}} \boldsymbol{\mu}'^{,\mathsf{T}} + o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2) \\
&\overset{(c)}{\leq} q(\boldsymbol{\lambda}^*) - \sum_{k=l+1}^{m} \nu_k \delta - \kappa \|\boldsymbol{\mu}'\|^2 + o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2) \\
&\overset{(d)}{<} q(\boldsymbol{\lambda}^*) - \kappa \|\boldsymbol{\nu}\|^2 - \kappa \|\boldsymbol{\mu}\|^2 + o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2) \\
&= q(\boldsymbol{\lambda}^*) - \kappa \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2 + o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2)
\end{aligned}
$$

where (a) follows from the second-order Taylor's expansion; (b) follows from the facts that $d_k = 0, \forall k \in \{1, \ldots, l\}$; $\boldsymbol{\nu} \geq 0$ and $d_k \leq -\delta$; the last $m - l$ elements of vector $\boldsymbol{\mu}$ are zeros; and $\boldsymbol{\Sigma} \prec 0$; (c) is true because $\kappa > 0$ exists when $\text{rank}(\mathbf{U}') = l$ and $\boldsymbol{\Sigma} \prec 0$; and (d) follows from $-\delta \leq -\kappa \nu_k, \forall k \in \{l+1, \ldots, m\}$, which is true as long as $\|\boldsymbol{\nu}\|$ is sufficiently small; and $\|\boldsymbol{\mu}'\| = \|\boldsymbol{\mu}\|$.

By the definition of $o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2)$, for any $\kappa > 0$, we have $o(\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2) \leq \frac{\kappa}{2} \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2$ as long as $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|$ is sufficiently small. Thus, there exists $D_q > 0$ such that

$$
q(\boldsymbol{\lambda}) \leq q(\boldsymbol{\lambda}^*) - L_q \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|^2, \forall \boldsymbol{\lambda} \in \{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_q\}
$$

where $L_s = \kappa/2$. □

Lemma 2.14. *Let $q(\boldsymbol{\lambda}) : \mathbb{R}_+^m \to \mathbb{R}$ be a second-order continuously differentiable concave function maximized at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^* \geq \mathbf{0}$. If $\nabla_{\boldsymbol{\lambda}}^2 q(\boldsymbol{\lambda}^*) \prec 0$, then there exists $D_c > 0$ and $L_c > 0$ such that $q(\cdot)$ is strongly concave on the set $\boldsymbol{\lambda} \in \{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq D_c\}$*

Proof. This lemma trivially follows from the continuity of $\nabla_{\boldsymbol{\lambda}}^2 q(\boldsymbol{\lambda})$. □

Proof of Part 1 of Theorem 2.10:

Note that Assumption 2.1 is trivially true. Assumption 2.2 follows from the assumption[5] that $\text{rank}(\nabla \mathbf{g}_{\mathcal{K}}(\mathbf{x}^*)^\mathsf{T}) = l$. To show that Assumption 2.3 holds, we need to apply Lemma 2.13.

By the strong convexity of $f(\mathbf{x})$ and Proposition B.25 in [Ber99], the dual function $q(\boldsymbol{\lambda})$ is differentiable and has gradient $\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = \mathbf{g}(\mathbf{x}^*)$. Thus, $\mathbf{d} = \nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) \leq \mathbf{0}$. By Assumption 2.2, i.e., the strong duality, we have $\lambda_k^* d_k = 0, \forall k \in \{1, \ldots, m\}$. Thus, the first condition in Lemma 2.13 is satisfied.

For $\boldsymbol{\lambda} \geq 0$, define $\mathbf{x}^*(\boldsymbol{\lambda}) = \text{argmin}_{\mathbf{x} \in \mathbb{R}^n} \left[f(\mathbf{x}) + \boldsymbol{\lambda}^\mathsf{T} \mathbf{g}(\mathbf{x}) \right]$ and denote $\mathbf{x}^*(\boldsymbol{\lambda}^*) = \mathbf{x}^*$. Note that $\mathbf{x}^*(\boldsymbol{\lambda})$ is a well-defined function because $f(\mathbf{x}) + \boldsymbol{\lambda}^\mathsf{T} \mathbf{g}(\mathbf{x})$ is strongly convex and hence is minimized at a unique point. By equation (6.9), page 598, in [Ber99], we have

$$\nabla^2_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = -\left[\nabla_{\mathbf{x}} \mathbf{g}(\mathbf{x}^*)\right]^\mathsf{T} \left[\nabla^2_{\mathbf{x}} f(\mathbf{x}^*) + \sum_{k=1}^m \lambda_k^* \nabla^2_{\mathbf{x}} g_k(\mathbf{x}^*)\right]^{-1} \left[\nabla_{\mathbf{x}} \mathbf{g}(\mathbf{x}^*)\right] \tag{2.31}$$

Note that $\nabla^2_{\mathbf{x}} f(\mathbf{x}^*) + \sum_{k=1}^m \lambda_k^* \nabla^2_{\mathbf{x}} g_k(\mathbf{x}^*) \succ 0$ because f is strongly convex and $g_k, k \in \{1, \ldots, m\}$ are convex. Thus, if $\text{rank}(\nabla_{\mathbf{x}} \mathbf{g}_{\mathcal{K}}(\mathbf{x}^*)^\mathsf{T}) = |\mathcal{K}|$, then the second condition of Lemma 2.13 is satisfied.

Proof of Part 2 of Theorem 2.10: Using the same argument, we can show that Assumptions 2.1-2.2 hold. By equation (2.31) and the assumption that $\text{rank}(\nabla_{\mathbf{x}} \mathbf{g}(\mathbf{x}^*)^\mathsf{T}) = m$, Assumption 2.4 follows from Lemma 2.14.

2.7.6 Proof of Theorem 2.11

- Proof of Part 1: Let $\mathbf{x}^*$ be the optimal solution to the problem (2.21)-(2.23). Since each column of $\mathbf{A}$ has at least one non-zero entry, we have $x_i^* \leq b^{\max}, \forall i \in \{1, 2, \ldots, n\}$ with $b^{\max} = \max_{1 \leq i \leq n} b_i$. Thus, the problem (2.24)-(2.26) is equivalent to the problem (2.21)-(2.23) since only a redundant constraint $\mathbf{x} \leq \mathbf{x}^{\max}$ is introduced.

- Proof of Part 2:

  - To show Assumption 2.1 holds: It follows from the strong convexity of $\sum_{i=1}^n -w_i \log(x_i)$ over set $\mathcal{X} = \{\mathbf{0} \leq \mathbf{x} \leq \mathbf{x}^{\max}\}$.

[5]The assumption that $\text{rank}(\nabla_{\mathbf{x}} \mathbf{g}_{\mathcal{K}}(\mathbf{x}^*)^\mathsf{T}) = l$ is known as the non-degenerate constraint qualification or linear independence constraint qualification, which along with Slater's constraint qualification, is one of various constraint qualifications implying strong duality [BSS06, SB94].

– To show Assumption 2.2 holds: If $x_i^* = 0, \forall 1 \leq i \leq n$, then the objective function is $+\infty$. Thus, $x_i^* > 0, \forall 1 \leq i \leq n$. Since $x_i^* \leq b^{\max}, \forall 1 \leq i \leq n$ and $x_i^{\max} > b^{\max}, \forall 1 \leq i \leq n$, we have $x_i < x_i^{\max}, \forall 1 \leq i \leq n$. Thus, constraints $\mathbf{x} \geq 0$ and $\mathbf{x} \leq \mathbf{x}^{\max}$ are inactive. The active inequality constraints can only be those among $\mathbf{Ax} \leq \mathbf{b}$. Thus, if $\text{rank}(\mathbf{A}) = m$ then the linear dependence constraint qualification is satisfied and the strong duality holds [BSS06, SB94]. Note that the strong duality also holds in the problem (2.21)-(2.23) because its active inequality constraints also can only be those among $\mathbf{Ax} \leq \mathbf{b}$.

– To show Assumption 2.3 holds: Define the Lagrangian dual function of the problem (2.21)-(2.23) as

$$\tilde{q}(\boldsymbol{\lambda}) = \min_{\mathbf{x} \geq \mathbf{0}} \Big\{ \sum_{i=1}^{n} -w_i \log(x_i) + \boldsymbol{\lambda}^{\mathsf{T}}(\mathbf{Ax} - \mathbf{b}) \Big\}.$$

Note that $\text{argmin}_{\mathbf{x} \geq \mathbf{0}} \big\{ \sum_{i=1}^{n} -w_i \log(x_i) + \boldsymbol{\lambda}^{\mathsf{T}}(\mathbf{Ax} - \mathbf{b}) \big\} = \big[[\frac{1}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_1}]_0^{\infty}, \ldots, [\frac{1}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_n}]_0^{\infty} \big]^{\mathsf{T}}$ where the $[\cdot]_a^b$ denotes the projection onto the interval $[a, b]$. As argued above, the strong duality holds for problem (2.21)-(2.23). Let $\mathbf{x}^*$ be the optimal solution to the above convex program and $\boldsymbol{\lambda}^* \geq \mathbf{0}$ be the corresponding dual variables. By strong duality, $\mathbf{x}^* = \text{argmin}_{\mathbf{x} \geq \mathbf{0}} \big\{ \sum_{i=1}^{n} -w_i \log(x_i) + (\boldsymbol{\lambda}^*)^{\mathsf{T}}(\mathbf{Ax} - \mathbf{b}) \big\}$, i.e, $x_i^* = \text{argmin}_{x_i \geq 0} \big\{ -w_i \log(x_i) + (\boldsymbol{\lambda}^*)^{\mathsf{T}}\mathbf{a}_i x_i \big\}, \forall 1 \leq i \leq n$. Thus, we have $x_i^* = \big[\frac{w_i}{(\boldsymbol{\lambda}^*)^{\mathsf{T}}\mathbf{a}_i} \big]_0^{\infty}, \forall 1 \leq i \leq n$. In the proof of part 1 of this theorem, we show that $0 < x_i^* \leq b^{\max}, \forall 1 \leq i \leq n$. Thus, $0 < \frac{w_i}{(\boldsymbol{\lambda}^*)^{\mathsf{T}}\mathbf{a}_i} \leq b^{\max}, \forall 1 \leq i \leq n$ and $x_i^* = \big[\frac{w_i}{(\boldsymbol{\lambda}^*)^{\mathsf{T}}\mathbf{a}_i} \big]_0^{\infty} = \frac{w_i}{(\boldsymbol{\lambda}^*)^{\mathsf{T}}\mathbf{a}_i}, \forall 1 \leq i \leq n$.

Now consider the problem (2.24)-(2.26). $\mathbf{x}^*$ is still the optimal solution and $\boldsymbol{\lambda}^*$ is still the corresponding dual variable when the Lagrangian dual function is defined as $q(\boldsymbol{\lambda}) = \min_{\mathbf{0} \leq \mathbf{x} \leq \mathbf{x}^{\max}} \big\{ \sum_{i=1}^{n} -w_i \log(x_i) + \boldsymbol{\lambda}^{\mathsf{T}}(\mathbf{Ax} - \mathbf{b}) \big\}$. Note that

$$\underset{\mathbf{0} \leq \mathbf{x} \leq \mathbf{x}^{\max}}{\text{argmin}} \Big\{ \sum_{i=1}^{n} -w_i \log(x_i) + \boldsymbol{\lambda}^{\mathsf{T}}(\mathbf{Ax} - \mathbf{b}) \Big\} = \big[\big[\frac{w_1}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_1} \big]_0^{x_1^{\max}}, \ldots, \big[\frac{w_n}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_n} \big]_0^{x_n^{\max}} \big]^{\mathsf{T}}.$$

By the fact that $0 < \frac{w_i}{(\boldsymbol{\lambda}^*)^{\mathsf{T}}\mathbf{a}_i} < x_i^{\max}, \forall i \in \{1, 2, \ldots, n\}$ and by the continuity of functions $\frac{w_i}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_i}$,, we know

$$\big[\big[\frac{w_1}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_1} \big]_0^{x_1^{\max}}, \ldots, \big[\frac{w_n}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_n} \big]_0^{x_n^{\max}} \big] = \big[\big[\frac{w_1}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_1} \big]_0^{\infty}, \ldots, \big[\frac{w_n}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_n} \big]_0^{\infty} \big] = \big[\frac{w_1}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_1}, \ldots, \frac{w_n}{\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{a}_n} \big]$$

when $\boldsymbol{\lambda}$ is sufficiently near $\boldsymbol{\lambda}^*$. That is, $q(\boldsymbol{\lambda}) = \tilde{q}(\boldsymbol{\lambda})$ when $\boldsymbol{\lambda}$ is sufficiently near $\boldsymbol{\lambda}^*$. Next, we show that the dual function $q(\boldsymbol{\lambda})$ is locally quadratic in a neighborhood of $\boldsymbol{\lambda}^*$ by using Lemma 2.13. Consider $\boldsymbol{\lambda} \in \mathbb{R}^m_+$ such that $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|$ is sufficiently small, or equivalently, $\boldsymbol{\lambda}$ is sufficiently close to $\boldsymbol{\lambda}^*$. For such $\boldsymbol{\lambda}$, we have

$$\underset{\mathbf{0}\leq\mathbf{x}\leq\mathbf{x}^{\max}}{\operatorname{argmin}} \Big\{ \sum_{i=1}^{n} -w_n \log(x_i) + \boldsymbol{\lambda}^\mathsf{T}(\mathbf{A}\mathbf{x} - \mathbf{b}) \Big\} = \big[\frac{w_1}{\boldsymbol{\lambda}^\mathsf{T}\mathbf{a}_1}, \ldots, \frac{w_n}{\boldsymbol{\lambda}^\mathsf{T}\mathbf{a}_n}\big]^\mathsf{T}$$

Thus,

$$\begin{aligned} q(\boldsymbol{\lambda}) &= \sum_{i=1}^{n} \Big[- w_i \log\big(\frac{1}{\boldsymbol{\lambda}^\mathsf{T}\mathbf{a}_i}\big) + \frac{w_i \boldsymbol{\lambda}^\mathsf{T}\mathbf{a}_i}{\boldsymbol{\lambda}^\mathsf{T}\mathbf{a}_i} \Big] - \boldsymbol{\lambda}^\mathsf{T}\mathbf{b} \\ &= \sum_{i=1}^{n} \Big[- w_i \log\big(\frac{1}{\boldsymbol{\lambda}^\mathsf{T}\mathbf{a}_i}\big) \Big] + \sum_{i=1}^{n} w_i - \boldsymbol{\lambda}^\mathsf{T}\mathbf{b} \end{aligned}$$

for $\boldsymbol{\lambda} \in \mathbb{R}^m_+$ such that $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|$ is sufficiently small. Note that $q(\boldsymbol{\lambda})$ is infinitely differentiable at any $\boldsymbol{\lambda} \in \mathbb{R}^m_+$ such that $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|$ is sufficiently small. Taking the first-order and second-order derivatives at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$ yields

$$\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = \sum_{i=1}^{n} \frac{w_i \mathbf{a}_i}{(\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_i} - \mathbf{b}, \tag{2.32}$$

$$\nabla^2_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}^*) = -\sum_{i=1}^{n} \frac{w_i \mathbf{a}_i \mathbf{a}_i^\mathsf{T}}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_i)^2} = \mathbf{A}\operatorname{diag}\big([-\frac{w_1}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_i)^2}, \ldots, -\frac{w_n}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_n)^2}]\big)\mathbf{A}^\mathsf{T}, \tag{2.33}$$

where $\operatorname{diag}\big([-\frac{w_1}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_i)^2}, \ldots, -\frac{w_n}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_n)^2}]\big)$ denotes the diagonal matrix with diagonal entries $-\frac{w_1}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_i)^2}, \ldots, -\frac{w_n}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_n)^2}$. Note that $\frac{w_i}{((\boldsymbol{\lambda}^*)^\mathsf{T}\mathbf{a}_i)^2} > 0, \forall 1 \leq i \leq n$. Thus, if $\operatorname{rank}(\mathbf{A}') = m'$, then Assumption 2.3 holds by Lemma 2.13.

- Proof of Part 3: Using the same arguments in the proof of part 2, we can show that Assumptions 2.1-2.2 hold. By equation (2.33) and the fact that $\operatorname{rank}(\mathbf{A}) = m$, Assumption 2.4 follows from Lemma 2.14.

Chapter 3

New Lagrangian Methods for Constrained Convex Programs

Fix positive integers n and m. Consider general constrained convex programs given by:

$$\min\ f(\mathbf{x}) \tag{3.1}$$

$$\text{s.t. } g_k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \ldots, m\}, \tag{3.2}$$

$$\mathbf{x} \in \mathcal{X}, \tag{3.3}$$

where the set $\mathcal{X} \subseteq \mathbb{R}^n$ is a closed convex set; the function $f(\mathbf{x})$ is convex on $\mathcal{X}$; and the functions $g_k(\mathbf{x}), \forall k \in \{1, 2, \ldots, m\}$ are convex on $\mathcal{X}$. Note that the functions $f(\mathbf{x}), g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})$ are not necessarily differentiable. Denote the stacked vector of multiple functions $g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})$ as $\mathbf{g}(\mathbf{x}) = \big[g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})\big]^{\mathsf{T}}$. Throughout this chapter, the convex program (3.1)-(3.3) is required to satisfy the following assumptions:

Assumption 3.1 (Lipschitzness)**.**

- *There exists a (possibly non-unique) optimal solution* $\mathbf{x}^* \in \mathcal{X}$ *that solves the convex program* (3.1)-(3.3)*.*
- *There exists a constant* β *such that* $\|\mathbf{g}(\mathbf{x}_1) - \mathbf{g}(\mathbf{x}_2)\| \leq \beta \|\mathbf{x}_1 - \mathbf{x}_2\|$ *for all* $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$*. That is, the function* $\mathbf{g}(\mathbf{x})$ *is Lipschitz continuous on* $\mathcal{X}$ *with modulus* β*.*

Assumption 3.2 (Existence of Lagrange Multipliers)**.** *Condition 1.1 holds for the convex program* (3.1)-(3.3)*. That is, there exists a Lagrange multiplier vector* $\boldsymbol{\lambda}^* = [\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*]^{\mathsf{T}} \geq \mathbf{0}$ *such that*

$$q(\boldsymbol{\lambda}^*) = f(\mathbf{x}^*),$$

where $\mathbf{x}^*$ *is an optimal solution to the problem* (3.1)-(3.3) *and* $q(\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x})+\sum_{k=1}^{m}\lambda_k g_k(\mathbf{x})\}$ *is the Lagrangian dual function of the problem* (3.1)-(3.3).

As reviewed in Section 1.2 in Chapter 1, existing Lagrangian methods for general (possibly non-differentiable non-strongly convex) convex programs with (possibly nonlinear) functional constraints have a slow $O(\frac{1}{\epsilon^2})$ convergence time. In this chapter, we present two new Lagrangian methods, both of which have a fast $O(\frac{1}{\epsilon})$ convergence time. The first algorithm is originally developed in our paper [YN17e] and the second algorithm is originally developed in our paper [YN16c] and our technical report [YN17d].

The first algorithm works for general (possibly non-differentiable) constrained convex programs under Assumptions 3.1 and 3.2 and updates the primal variables by solving an unconstrained convex minimization that can be decomposed into independent smaller subproblems when $f(\mathbf{x})$ and $g_k(\mathbf{x})$ are separable. This new algorithm directly improves Algorithm 1.3, the drift-plus-penalty method for deterministic convex programs, or equivalently, Algorithm 1.1, the dual subgradient method, which only achieves a slow $O(\frac{1}{\epsilon^2})$ convergence time with a similar primal update scheme.

The second algorithm further requires that $f(\mathbf{x})$ and $g_k(\mathbf{x})$ in the convex program (3.1)-(3.3) are smooth; and updates the the primal variables by following a projected gradient update that can be distributively implemented even when $f(\mathbf{x})$ or $g_k(\mathbf{x})$ are not separable. This new algorithm directly improves Algorithm 1.2, the primal-dual subgradient method, which only achieves a slow $O(\frac{1}{\epsilon^2})$ convergence time with a similar primal update scheme.

3.1 New Dual Type Algorithm for General Constrained Convex Programs

Consider the following algorithm described in Algorithm 3.1. This algorithm computes both primal variables $\mathbf{x}(t) \in \mathcal{X}$ and dual variables $\mathbf{Q}(t) = \big[Q_1(t), \ldots, Q_m(t)\big]^{\mathsf{T}}$, called *virtual queue* vectors in the drift-plus-penalty technique, at iterations $t \in \{0, 1, 2, \ldots\}$. One main result of this chapter is that, whenever the parameter α in Algorithm 3.1 is chosen to satisfy $\alpha \geq \beta^2/2$, the running average $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbf{x}(\tau)$ closely approximates a solution to the convex program (3.1)-(3.3) and has an approximation error that decays like $O(1/t)$.

Algorithm 3.1 New Dual Type Algorithm for General Constrained Convex Programs

Let $\alpha > 0$ be a constant parameter. Choose any $\mathbf{x}(-1) \in \mathcal{X}$. Initialize $Q_k(0) = \max\{0, -g_k(\mathbf{x}(-1))\}, \forall k \in \{1, 2, \ldots, m\}$. At each iteration $t \in \{0, 1, 2, \ldots\}$, update $\mathbf{x}(t)$ and $\mathbf{Q}(t+1)$ as follows:

- Update primal variables via

$$\mathbf{x}(t) = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \big\{ f(\mathbf{x}) + \big[\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))\big]^{\mathsf{T}} \mathbf{g}(\mathbf{x}) + \alpha \|\mathbf{x} - \mathbf{x}(t-1)\|^2 \big\}.$$

- Update virtual queues via

$$Q_k(t+1) = \max\big\{ -g_k(\mathbf{x}(t)), Q_k(t) + g_k(\mathbf{x}(t))\big\}, \forall k \in \{1, 2, \ldots, m\}. \tag{3.4}$$

- Output the running average $\overline{\mathbf{x}}(t+1)$ given by

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

  as the solution at iteration $t+1$.

Algorithm 3.1 is similar to Algorithm 1.3, the DPP technique for deterministic convex programs, with the following distinctions:

1. The Lagrange multiplier ("virtual queue") update equation for $Q_k(t)$ is modified to take a max with $-g_k(\mathbf{x}(t))$, rather than simply project onto the nonnegative real numbers as the traditional update rule $Q_k(t+1) = \max\{Q_k(t) + g_k(\mathbf{x}(t)), 0\}$ used in Algorithm 1.3 .

2. The minimization step augments the $Q_k(t)$ weights with $g_k(\mathbf{x}(t-1))$ values obtained on the previous step. These $g_k(\mathbf{x}(t-1))$ quantities, when multiplied by constraint functions $g_k(\mathbf{x})$, yield a cross-product term in the primal update. This cross term together with another newly introduced quadratic term in the primal update can cancel a quadratic term in an upper bound of the Lyapunov drift such that a finer analysis of the drift-plus-penalty leads to the fast $O(\frac{1}{\epsilon})$ convergence time.

3. A quadratic term, which is similar to a term used in proximal algorithms [PB13], is introduced. This provides a strong convexity "pushback". The pushback is not sufficient to alone cancel the main drift components, but it cancels residual components introduced by the new $g_k(\mathbf{x}(t-1))$ weight.

At the same time, Algorithm 3.1 preserves the desirable properties possessed by Algorithm 1.3. That is, if the functions $f(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ are separable with respect to components or blocks

of $\mathbf{x}$, then the primal updates for $\mathbf{x}(t)$ can be decomposed into several smaller independent subproblems, each of which only involves a component or block of $\mathbf{x}(t)$.

3.2 Basic Properties from Virtual Queue Update Equations

This section presents important facts from the virtual queue update equation (3.4).

3.2.1 Properties of Virtual Queues

Lemma 3.1. *Let $\mathbf{Q}(t), t \in \{0, 1, \ldots\}$ be the sequence of virtual queue vectors yielded by the update equation* (3.4)*. Then,*

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

Proof.

1. Fix $k \in \{1, 2, \ldots, m\}$. Note that $Q_k(0) \geq 0$ by initialization $Q_k(0) = \max\{0, -g_k(\mathbf{x}(-1))\}$. Assume $Q_k(t) \geq 0$ and consider iteration $t+1$. If $g_k(\mathbf{x}(t)) \geq 0$, then $Q_k(t+1) = \max\{-g_k(\mathbf{x}(t)), Q_k(t) + g_k(\mathbf{x}(t))\} \geq Q_k(t) + g_k(\mathbf{x}(t)) \geq 0$. If $g_k(\mathbf{x}(t)) < 0$, then $Q_k(t+1) = \max\{-g_k(\mathbf{x}(t)), Q_k(t) + g_k(\mathbf{x}(t))\} \geq -g_k(\mathbf{x}(t)) > 0$. Thus, $Q_k(t+1) \geq 0$. The result follows by induction.

2. Fix $k \in \{1, 2, \ldots, m\}$. Note that $Q_k(0) + g_k(\mathbf{x}(-1)) \geq 0$ by initialization rule $Q_k(0) = \max\{0, -g_k(\mathbf{x}(-1))\} \geq -g_k(\mathbf{x}(-1))$. For $t \geq 1$, by the virtual queue update equation, we have

$$Q_k(t) = \max\{-g_k(\mathbf{x}(t-1)), Q_k(t-1) + g_k(\mathbf{x}(t-1))\} \geq -g_k(\mathbf{x}(t-1)),$$

   which implies that $Q_k(t) + g_k(\mathbf{x}(t-1)) \geq 0$.

3. Consider $t = 0$. Fix $k \in \{1, 2, \ldots, m\}$. Consider the cases $g_k(\mathbf{x}(-1)) \geq 0$ and $g_k(\mathbf{x}(-1)) < 0$ separately. If $g_k(\mathbf{x}(-1)) \geq 0$, then $Q_k(0) = \max\{0, -g_k(\mathbf{x}(-1))\} = 0$ and so $|Q_k(0)| \leq$

$|g_k(\mathbf{x}(-1))|$. If $g_k(\mathbf{x}(-1)) < 0$, then $Q_k(0) = \max\{0, -g_k(\mathbf{x}(-1))\} = -g_k(\mathbf{x}(-1)) = |g_k(\mathbf{x}(-1))|$. Thus, in both cases, we have $|Q_k(0)| \leq |g_k(\mathbf{x}(-1))|$. Squaring both sides and summing over $k \in \{1, 2, \ldots, m\}$ yields $\|\mathbf{Q}(0)\|^2 \leq \|\mathbf{g}(\mathbf{x}(-1))\|^2$.

Consider $t \geq 1$. Fix $k \in \{1, 2, \ldots, m\}$. Consider the cases $g_k(\mathbf{x}(t-1)) \geq 0$ and $g_k(\mathbf{x}(t-1)) < 0$ separately. If $g_k(\mathbf{x}(t-1)) \geq 0$, then

$$\begin{aligned} Q_k(t) &= \max\{-g_k(\mathbf{x}(t-1)), Q_k(t-1) + g_k(\mathbf{x}(t-1))\} \\ &\geq Q_k(t-1) + g_k(\mathbf{x}(t-1)) \\ &\overset{(a)}{\geq} g_k(\mathbf{x}(t-1)) \\ &= |g_k(\mathbf{x}(t-1))| \end{aligned}$$

where (a) follows from part 1. If $g_k(\mathbf{x}(t-1)) < 0$, then

$$\begin{aligned} Q_k(t) &= \max\{-g_k(\mathbf{x}(t-1)), Q_k(t-1) + g_k(\mathbf{x}(t-1))\} \\ &\geq -g_k(\mathbf{x}(t-1)) \\ &= |g_k(\mathbf{x}(t-1))|. \end{aligned}$$

Thus, in both cases, we have $|Q_k(t)| \geq |g_k(\mathbf{x}(t-1))|$. Squaring both sides and summing over $k \in \{1, 2, \ldots, m\}$ yields $\|\mathbf{Q}(t)\|^2 \geq \|\mathbf{g}(\mathbf{x}(t-1))\|^2$.

□

Lemma 3.2. *Let $\mathbf{Q}(t), t \in \{0, 1, \ldots\}$ be the sequence of virtual queue vectors yielded by the update equation (3.4). At each iteration $t \in \{1, 2, \ldots\}$,*

$$Q_k(t) \geq \sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)), \forall k \in \{1, 2, \ldots, m\}. \tag{3.5}$$

Proof. Fix $k \in \{1, 2, \ldots, m\}$ and $t \geq 1$. For any $\tau \in \{0, \ldots, t-1\}$ the update rule of Algorithm 3.1 gives:

$$\begin{aligned} Q_k(\tau+1) &= \max\{-g_k(\mathbf{x}(\tau)), Q_k(\tau) + g_k(\mathbf{x}(\tau))\} \\ &\geq Q_k(\tau) + g_k(\mathbf{x}(\tau)). \end{aligned}$$

Hence, $Q_k(\tau+1) - Q_k(\tau) \geq g_k(\mathbf{x}(\tau))$. Summing over $\tau \in \{0,\ldots,t-1\}$ and using $Q_k(0) \geq 0$ gives the result. □

3.2.2 Properties of the Drift

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

$$\Delta(t) = L(t+1) - L(t) = \frac{1}{2}\big[\|\mathbf{Q}(t+1)\|^2 - \|\mathbf{Q}(t)\|^2\big]. \tag{3.6}$$

Lemma 3.3. *Let $\mathbf{Q}(t), t \in \{0,1,\ldots\}$ be the sequence of virtual queue vectors yielded by the update equation (3.4). At each iteration $t \in \{0,1,2,\ldots\}$, an upper bound of the Lyapunov drift is given by*

$$\Delta(t) \leq [\mathbf{Q}(t)]^\mathsf{T}\mathbf{g}(\mathbf{x}(t)) + \|\mathbf{g}(\mathbf{x}(t))\|^2. \tag{3.7}$$

Proof. The virtual queue update equation $Q_k(t+1) = \max\{-g_k(\mathbf{x}(t)), Q_k(t) + g_k(\mathbf{x}(t))\}, \forall k \in \{1,2,\ldots,m\}$ can be rewritten as

$$Q_k(t+1) = Q_k(t) + \tilde{g}_k(\mathbf{x}(t)), \forall k \in \{1,2,\ldots,m\}, \tag{3.8}$$

where

$$\tilde{g}_k(\mathbf{x}(t)) = \begin{cases} g_k(\mathbf{x}(t)), & \text{if } Q_k(t) + g_k(\mathbf{x}(t)) \geq -g_k(\mathbf{x}(t)) \\ -Q_k(t) - g_k(\mathbf{x}(t)), & \text{else} \end{cases} \quad \forall k.$$

Fix $k \in \{1, 2, \ldots, m\}$. Squaring both sides of (3.8) and dividing by 2 yield:

$$\begin{aligned}
&\frac{1}{2}[Q_k(t+1)]^2 \\
=&\frac{1}{2}[Q_k(t)]^2 + \frac{1}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + Q_k(t)\tilde{g}_k(\mathbf{x}(t)) \\
=&\frac{1}{2}[Q_k(t)]^2 + \frac{1}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + Q_k(t)g_k(\mathbf{x}(t)) + Q_k(t)[\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))] \\
\overset{(a)}{=}&\frac{1}{2}[Q_k(t)]^2 + \frac{1}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + Q_k(t)g_k(\mathbf{x}(t)) - [\tilde{g}_k(\mathbf{x}(t)) + g_k(\mathbf{x}(t))][\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))] \\
=&\frac{1}{2}[Q_k(t)]^2 - \frac{1}{2}[\tilde{g}_k(\mathbf{x}(t))]^2 + Q_k(t)g_k(\mathbf{x}(t)) + [g_k(\mathbf{x}(t))]^2 \\
\leq&\frac{1}{2}[Q_k(t)]^2 + Q_k(t)g_k(\mathbf{x}(t)) + [g_k(\mathbf{x}(t))]^2,
\end{aligned}$$

where (a) follows from the fact that $Q_k(t)[\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))] = -[\tilde{g}_k(\mathbf{x}(t)) + g_k(\mathbf{x}(t))] \cdot [\tilde{g}_k(\mathbf{x}(t)) - g_k(\mathbf{x}(t))]$, which can be shown by considering $\tilde{g}_k(\mathbf{x}(t)) = g_k(\mathbf{x}(t))$ and $\tilde{g}_k(\mathbf{x}(t)) \neq g_k(\mathbf{x}(t))$. Summing over $k \in \{1, 2, \ldots, m\}$ yields

$$\frac{1}{2}\|\mathbf{Q}(t+1)\|^2 \leq \frac{1}{2}\|\mathbf{Q}(t)\|^2 + [\mathbf{Q}(t)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) + \|\mathbf{g}(\mathbf{x}(t))\|^2.$$

Rearranging the terms yields the desired result. □

3.3 Convergence Time Analysis of Algorithm 3.1

This section analyzes the convergence time of Algorithm 3.1 for the convex program (3.1)-(3.3) under Assumptions 3.1-3.2.

3.3.1 An Upper Bound of the Drift-Plus-Penalty Expression

Lemma 3.4. *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. If* $\alpha \geq \frac{1}{2}\beta^2$ *in Algorithm 3.1, then for all* $t \geq 0$*, we have*

$$\begin{aligned}
&\Delta(t) + f(\mathbf{x}(t)) \\
\leq& f(\mathbf{x}^*) + \alpha\big[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2\big],
\end{aligned}$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1.*

Proof. Fix $t \geq 0$. Note that Lemma 3.1 implies that $\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))$ is component-wise nonnegative. Hence, the function $f(\mathbf{x}) + \big[\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))\big]^{\mathsf{T}}\mathbf{g}(\mathbf{x})$ is convex with respect to $\mathbf{x}$ on $\mathcal{X}$. Since $\alpha\|\mathbf{x} - \mathbf{x}(t-1)\|^2$ is strongly convex with respect to $\mathbf{x}$ with modulus 2α, it follows that

$$f(\mathbf{x}) + \big[\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))\big]^{\mathsf{T}}\mathbf{g}(\mathbf{x}) + \alpha\|\mathbf{x} - \mathbf{x}(t-1)\|^2$$

is strongly convex with respect to $\mathbf{x}$ with modulus 2α.

Since $\mathbf{x}(t)$ is chosen to minimize the above strongly convex function, by Corollary 1.2, we have

$$\begin{aligned}
&f(\mathbf{x}(t)) + \big[\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))\big]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) + \alpha\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2 \\
\leq & f(\mathbf{x}^*) + \underbrace{\big[\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))\big]^{\mathsf{T}}\mathbf{g}(\mathbf{x}^*)}_{\leq 0} + \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2 \\
\overset{(a)}{\leq} & f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2, \qquad (3.9)
\end{aligned}$$

where (a) follows by using the fact that $g_k(\mathbf{x}^*) \leq 0$ for all $k \in \{1, 2, \ldots, m\}$ and $Q_k(t) + g_k(\mathbf{x}(t-1)) \geq 0$ (i.e., part 2 in Lemma 3.1) to eliminate the term marked by an underbrace.

Note that $\mathbf{u}_1^{\mathsf{T}}\mathbf{u}_2 = \frac{1}{2}\big[\|\mathbf{u}_1\|^2 + \|\mathbf{u}_2\|^2 - \|\mathbf{u}_1 - \mathbf{u}_2\|^2\big]$ for any $\mathbf{u}_1, \mathbf{u}_2 \in \mathbb{R}^m$. Thus, we have

$$[\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) = \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(t-1))\|^2 + \|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}(t))\|^2\big]. \qquad (3.10)$$

Substituting (3.10) into (3.9) and rearranging terms yields

$$\begin{aligned}
&f(\mathbf{x}(t)) + [\mathbf{Q}(t)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) \\
\leq & f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \alpha\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}(t))\|^2 \\
& - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2 \\
\overset{(a)}{\leq} & f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2 + (\frac{1}{2}\beta^2 - \alpha)\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 \\
& - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2 \\
\overset{(b)}{\leq} & f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2,
\end{aligned}$$

where (a) follows from the fact that $\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}(t))\| \leq \beta\|\mathbf{x}(t) - \mathbf{x}(t-1)\|$, which further follows from the assumption that $\mathbf{g}(\mathbf{x})$ is Lipschitz continuous with modulus β; (b) follows from the fact $\alpha \geq \frac{1}{2}\beta^2$.

Summing (3.7) with the above inequality and cancelling common terms on both sides yields

$$\Delta(t) + f(\mathbf{x}(t)) \leq f(\mathbf{x}^*) + \alpha\big[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2\big].$$

□

3.3.2 Objective Value Violations

Lemma 3.5. *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. Let* $\mathbf{x}^*$ *be an optimal solution of the problem* (3.1)-(3.3) *and* β *be the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1.*

1. *If* $\alpha \geq \frac{1}{2}\beta^2$ *in Algorithm 3.1, then for all* $t \geq 1$*, we have*

$$\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq tf(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2.$$

2. *If* $\alpha > \frac{1}{2}\beta^2$ *in Algorithm 3.1, then for all* $t \geq 1$*, we have*

$$\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq tf(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{\alpha}{2\alpha - \beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{\|\mathbf{Q}(t)\|^2}{2}.$$

Proof. By Lemma 3.4, we have $\Delta(\tau) + f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|] + \frac{1}{2}[\|\mathbf{g}(\mathbf{x}(\tau))\|^2 - \|\mathbf{g}(\mathbf{x}(\tau-1))\|^2]$ for all $\tau \in \{0, 1, 2, \ldots\}$. Summing over $\tau \in \{0, 1, \ldots, t-1\}$ yields

$$\begin{aligned}
&\sum_{\tau=0}^{t-1} \Delta(\tau) + \sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \\
\leq& tf(\mathbf{x}^*) + \alpha\sum_{\tau=0}^{t-1}[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|^2] + \frac{1}{2}\sum_{\tau=0}^{t-1}[\|\mathbf{g}(\mathbf{x}(\tau))\|^2 - \|\mathbf{g}(\mathbf{x}(\tau-1))\|^2].
\end{aligned}$$

Recalling that $\Delta(\tau) = L(\tau+1) - L(\tau)$ and simplifying summations yields

$$\begin{aligned}
&L(t) - L(0) + \sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \\
\leq & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(-1))\|^2.
\end{aligned}$$

Rearranging terms; and substituting $L(0) = \frac{1}{2}\|\mathbf{Q}(0)\|^2$ and $L(t) = \frac{1}{2}\|\mathbf{Q}(t)\|^2$ yields

$$\begin{aligned}
&\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \\
\leq & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(-1))\|^2 \\
&+ \frac{1}{2}\|\mathbf{Q}(0)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
\overset{(a)}{\leq} & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2, \qquad (3.11)
\end{aligned}$$

where (a) follows from the fact that $\|\mathbf{Q}(0)\| \leq \|\mathbf{g}(\mathbf{x}(-1))\|$, i.e., part 3 in Lemma 3.1.

Next, we present the proof of both parts:

1. This part follows from the observation that equation (3.11) can be further simplified as

$$\begin{aligned}
\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) &\overset{(a)}{\leq} t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
&\overset{(b)}{\leq} t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2,
\end{aligned}$$

   where (a) follows by ignoring the non-positive term $-\alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2$ on the right side and (b) follows from the fact that $\|\mathbf{Q}(t)\| \geq \|\mathbf{g}(\mathbf{x}(t-1))\|$, i.e., part 3 in Lemma 3.1.

2. This part follows by rewriting equation (3.11) as

$$
\begin{aligned}
&\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \\
\leq & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}^*) + \mathbf{g}(\mathbf{x}^*)\|^2 \\
& - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
= & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}^*)\|^2 \\
& + [\mathbf{g}(\mathbf{x}^*)]^{\mathsf{T}}[\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}^*)] + \frac{1}{2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
\overset{(a)}{\leq} & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}^*)\|^2 \\
& + \|\mathbf{g}(\mathbf{x}^*)\|\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}^*)\| + \frac{1}{2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
\overset{(b)}{\leq} & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 + \frac{1}{2}\beta^2\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 \\
& + \beta\|\mathbf{g}(\mathbf{x}^*)\|\|\mathbf{x}^* - \mathbf{x}(t-1)\| + \frac{1}{2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
= & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 - \big(\alpha - \frac{1}{2}\beta^2\big)\big[\|\mathbf{x}^* - \mathbf{x}(t-1)\| - \frac{1}{2}\frac{\beta}{\alpha - \frac{1}{2}\beta^2}\|\mathbf{g}(\mathbf{x}^*)\|\big]^2 \\
& + \frac{\alpha}{2\alpha - \beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
\overset{(c)}{\leq} & t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{\alpha}{2\alpha - \beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2,
\end{aligned}
$$

where (a) follows from the Cauchy-Schwarz inequality; (b) follows from the fact that $\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}^*)\| \leq \beta\|\mathbf{x}^* - \mathbf{x}(t-1)\|$, which further follows from the assumption that $\mathbf{g}(\mathbf{x})$ is Lipschitz continuous with modulus β; and (c) follows from the fact that $\alpha > \frac{1}{2}\beta^2$.

□

Theorem 3.1 (Objective Value Violations of Algorithm 3.1)**.** *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. If* $\alpha \geq \frac{1}{2}\beta^2$ *in Algorithm 3.1, for all* $t \geq 1$*, we have*

$$f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*) + \frac{\alpha}{t}\|\mathbf{x}^* - \mathbf{x}(-1)\|^2,$$

where $\mathbf{x}^*$ *is an optimal solution to the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1.*

Proof. Fix $t \geq 1$. By part 1 in Lemma 3.5, we have

$$\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq tf(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2$$
$$\Rightarrow \frac{1}{t}\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq f(\mathbf{x}^*) + \frac{\alpha}{t}\|\mathbf{x}^* - \mathbf{x}(-1)\|^2.$$

Since $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbf{x}(\tau)$ and $f(\mathbf{x})$ is convex, by Jensen's inequality it follows that

$$f(\overline{\mathbf{x}}(t)) \leq \frac{1}{t}\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)).$$

□

The above theorem shows that under Algorithm 3.1, the error gap between $f(\overline{\mathbf{x}}(t))$ and the optimal value $f(\mathbf{x}^*)$ is at most $O(\frac{1}{t})$. This holds for any initial guess vector $\mathbf{x}(-1) \in \mathcal{X}$. Of course, choosing $\mathbf{x}(-1)$ close to $\mathbf{x}^*$ is desirable because it reduces the coefficient $\alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2$.

3.3.3 Constraint Violations

The next Lemma follows from Assumption 3.2 and Lemma 3.2.

Lemma 3.6. *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. Let* $\mathbf{x}(t), \mathbf{Q}(t), t \in \{0, 1, \ldots\}$ *be sequences generated by Algorithm 3.1. Then,*

$$\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \geq tf(\mathbf{x}^*) - \|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t)\|, \quad \forall t \geq 1,$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *defined in Assumption 3.1; and* $\boldsymbol{\lambda}^*$ *is a Lagrange multiplier vector satisfying Assumption 3.2.*

Proof. Define Lagrangian dual function $q(\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) + \sum_{k=1}^m \lambda_k g_k(\mathbf{x})\}$. For all $\tau \in \{0, 1, \ldots\}$, by Assumption 3.2, we have

$$f(\mathbf{x}^*) = q(\boldsymbol{\lambda}^*) \overset{(a)}{\leq} f(\mathbf{x}(\tau)) + \sum_{k=1}^m \lambda_k^* g_k(\mathbf{x}(\tau)),$$

where (a) follows the definition of $q(\boldsymbol{\lambda}^*)$. Thus, we have

$$f(\mathbf{x}(\tau)) \geq f(\mathbf{x}^*) - \sum_{k=1}^{m} \lambda_k^* g_k(\mathbf{x}(\tau)), \forall \tau \in \{0, 1, \ldots\}.$$

Summing over $\tau \in \{0, 1, \ldots, t-1\}$ yields

$$\begin{aligned}
\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \geq & t f(\mathbf{x}^*) - \sum_{\tau=0}^{t-1} \sum_{k=1}^{m} \lambda_k^* g_k(\mathbf{x}(\tau)) \\
= & t f(\mathbf{x}^*) - \sum_{k=1}^{m} \lambda_k^* \Big[\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)) \Big] \\
\overset{(a)}{\geq} & t f(\mathbf{x}^*) - \sum_{k=1}^{m} \lambda_k^* Q_k(t) \\
\overset{(b)}{\geq} & t f(\mathbf{x}^*) - \|\boldsymbol{\lambda}^*\| \|\mathbf{Q}(t)\|,
\end{aligned}$$

where (a) follows from Lemma 3.2 and the fact that $\lambda_k^* \geq 0, \forall k \in \{1, 2, \ldots, m\}$; and (b) follows from the Cauchy-Schwarz inequality. □

Lemma 3.7. *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. If* $\alpha > \frac{\beta^2}{2}$ *in Algorithm 3.1, then for all* $t \geq 1$*, the virtual queue vector satisfies*

$$\|\mathbf{Q}(t)\| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\alpha}\|\mathbf{x}^* - \mathbf{x}(-1)\| + \sqrt{\frac{\alpha}{\alpha - \frac{1}{2}\beta^2}}\|\mathbf{g}(\mathbf{x}^*)\|,$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1; and* $\boldsymbol{\lambda}^*$ *is a Lagrange multiplier vector satisfying Assumption 3.2.*

Proof. Fix $t \geq 1$. By part 2 in Lemma 3.5, we have

$$\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq t f(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{\alpha}{2\alpha - \beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2.$$

By Lemma 3.6, we have

$$\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \geq t f(\mathbf{x}^*) - \|\boldsymbol{\lambda}^*\| \|\mathbf{Q}(t)\|.$$

Combining the last two inequalities and cancelling the common term $tf(\mathbf{x}^*)$ on both sides yields

$$\begin{aligned}
&\frac{1}{2}\|\mathbf{Q}(t)\|^2 - \left(\alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{\alpha}{2\alpha - \beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2\right) \leq \|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t)\| \\
\Rightarrow &\left(\|\mathbf{Q}(t)\| - \|\boldsymbol{\lambda}^*\|\right)^2 \leq \|\boldsymbol{\lambda}^*\|^2 + 2\alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{\alpha}{\alpha - \frac{1}{2}\beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2 \\
\Rightarrow &\|\mathbf{Q}(t)\| \leq \|\boldsymbol{\lambda}^*\| + \sqrt{\|\boldsymbol{\lambda}^*\|^2 + 2\alpha\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \frac{\alpha}{\alpha - \frac{1}{2}\beta^2}\|\mathbf{g}(\mathbf{x}^*)\|^2} \\
\overset{(a)}{\Rightarrow} &\|\mathbf{Q}(t)\| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\alpha}\|\mathbf{x}^* - \mathbf{x}(-1)\| + \sqrt{\frac{\alpha}{\alpha - \frac{1}{2}\beta^2}}\|\mathbf{g}(\mathbf{x}^*)\|,
\end{aligned}$$

where (a) follows from the basic inequality $\sqrt{a+b+c} \leq \sqrt{a} + \sqrt{b} + \sqrt{c}$ for any $a, b, c \geq 0$. □

Theorem 3.2 (Constraint Violations of Algorithm 3.1)**.** *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. If* $\alpha > \frac{\beta^2}{2}$ *in Algorithm 3.1, then for all* $t \geq 1$*, the constraint functions satisfy*

$$g_k(\overline{\mathbf{x}}(t)) \leq \frac{1}{t}\left(2\|\boldsymbol{\lambda}^*\| + \sqrt{2\alpha}\|\mathbf{x}^* - \mathbf{x}(-1)\| + \sqrt{\frac{\alpha}{\alpha - \frac{1}{2}\beta^2}}\|\mathbf{g}(\mathbf{x}^*)\|\right), \forall k \in \{1, 2, \ldots, m\},$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1; and* $\boldsymbol{\lambda}^*$ *is a Lagrange multiplier vector satisfying Assumption 3.2.*

Proof. Fix $t \geq 1$ and $k \in \{1, 2, \ldots, m\}$. Recall that $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbf{x}(\tau)$. Thus,

$$\begin{aligned}
g_k(\overline{\mathbf{x}}(t)) &\overset{(a)}{\leq} \frac{1}{t}\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)) \\
&\overset{(b)}{\leq} \frac{Q_k(t)}{t} \\
&\leq \frac{\|\mathbf{Q}(t)\|}{t} \\
&\overset{(c)}{\leq} \frac{1}{t}\left(2\|\boldsymbol{\lambda}^*\| + \sqrt{2\alpha}\|\mathbf{x}^* - \mathbf{x}(-1)\| + \sqrt{\frac{\alpha}{\alpha - \frac{1}{2}\beta^2}}\|\mathbf{g}(\mathbf{x}^*)\|\right),
\end{aligned}$$

where (a) follows from the convexity of $g_k(\mathbf{x}), k \in \{1, 2, \ldots, m\}$ and Jensen's inequality; (b) follows from Lemma 3.2; and (c) follows from Lemma 3.7. □

3.3.4 Convergence Time of Algorithm 3.1

The next theorem summarizes Theorems 3.1 and 3.2.

Theorem 3.3 (Convergence Time of Algorithm 3.1)**.** *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.2. If* $\alpha > \frac{\beta^2}{2}$ *in Algorithm 3.1, then for all* $t \geq 1$*, we have*

$$\begin{aligned} f(\overline{\mathbf{x}}(t)) \leq & f(\mathbf{x}^*) + \frac{\alpha}{t}\|\mathbf{x}^* - \mathbf{x}(-1)\|^2, \\ g_k(\overline{\mathbf{x}}(t)) \leq & \frac{1}{t}\Big(2\|\boldsymbol{\lambda}^*\| + \sqrt{2\alpha}\|\mathbf{x}^* - \mathbf{x}(-1)\| + \sqrt{\frac{\alpha}{\alpha - \frac{1}{2}\beta^2}}\|\mathbf{g}(\mathbf{x}^*)\|\Big), \forall k \in \{1, 2, \ldots, m\}, \end{aligned}$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1; and* $\boldsymbol{\lambda}^*$ *is a Lagrange multiplier vector satisfying Assumption 3.2. In summary, Algorithm 3.1 ensures error decays like* $O(\frac{1}{t})$ *and provides an* ϵ*-approximate solution with convergence time* $O(\frac{1}{\epsilon})$*.*

3.3.5 Convex Programs with Linear Equality Constraints

So far, it is assumed that there is no linear equality constraint in the convex program (3.1)-(3.3). In fact, if the convex program (3.1)-(3.3) contains a linear equality constraint given by $h(\mathbf{x}) = 0$, we can replace it with two inequality constraints $h(\mathbf{x}) \leq 0$ and $-h(\mathbf{x}) \leq 0$, both of which are convex inequality constraints, to rewrite the original convex program into the form of (3.1)-(3.3). After that, we can further apply Algorithm 3.1 to solve the reformulated convex program with an $O(1/\epsilon)$ convergence time. However, by doing this, two virtual queues, rather than one, are needed for each linear equality constraint and it is obvious that more virtual queues incurs more computation and storage overhead in the implementation of Algorithm 3.1.

By looking into the proof of Lemma 3.4, we realize that one reason why the virtual queue update equation (3.4) update $Q_k(t)$ as the larger one between $-g_k(\mathbf{x}(t-1))$ and $Q_k(t-1)+g_k(\mathbf{x}(t-1))$ is to yield a non-negative coefficient vector $\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))$ such that $\big[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))\big]^{\mathsf{T}}\mathbf{g}(\mathbf{x})$ involved in the primal update is convex. This is necessary since a convex function multiplying a negative constant is in general no longer convex. However, if $g_k(\mathbf{x})$ is a linear function, then $cg_k(\mathbf{x})$ is convex no matter the constant c is positive or negative.

In fact, if $g_k(\mathbf{x})$ is linear and the convex program (3.1)-(3.3) has a linear equality constraint given by $g_k(\mathbf{x}) = 0$, then it suffices to update the corresponding virtual queue $Q_k(t)$ using the equation:

$$Q_k(t+1) = Q_k(t) + g_k(\mathbf{x}(t)), \forall t \in \{0, 1, 2, \ldots\}. \tag{3.12}$$

and initialize the virtual queue with $Q_k(0) = 0$ to ensure the same $O(1/\epsilon)$ convergence time of Algorithm 3.1 without modifying any other steps.

To simply the analysis, we now consider an extreme case of constrained convex programs where all functional constrains are linear equality constraints given by $\mathbf{g}(\mathbf{x}) = 0$. Instead of replacing $\mathbf{g}(\mathbf{x}) = 0$ with $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$ and $\mathbf{g}(\mathbf{x}) = \mathbf{0}$ and applying the original Algorithm 3.1, we keep all equality constraints unchanged, and initialize $Q_k(0) = 0, \forall k$ and replace (3.4) with (3.12) in Algorithm 3.1. In the reminder of this section, we sketch the $O(1/t)$ convergence rate analysis of such a modification of Algorithm 3.1.

Note that $Q_k(0) = 0$ and virtual queue update equation (3.12) guarantees that $\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)) = Q_k(t), \forall t \geq 1$, which can be easily proven by using the same argument in the proof of Lemma 3.2. Thus, if we can show that $\|\mathbf{Q}(t)\|$ is bounded from above by a constant for all t, then we can establish the $O(\frac{1}{t})$ convergence rate of constraint violations.

Squaring both sides of (3.12) and summing over $k \in \{1, 2, \ldots, m\}$ yields

$$\Delta(t) = [\mathbf{Q}(t)]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2 \tag{3.13}$$

which is a drift identity that is even tighter than the drift bound in Lemma 3.3.

Recall that all linear functions are Lipschitz continuous, we assume $\mathbf{g}(\mathbf{x})$ is Lipschitz continuous with modulus β. Note that $\mathbf{Q}(t)$ updated by (3.12) ensures $f(\mathbf{x}) + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t))]^{\mathsf{T}}\mathbf{g}(\mathbf{x})$ is convex as long as $\mathbf{g}$ is linear. By using similar steps in the proof Lemma 3.4 and using (3.13) rather than (3.7) in the last step, we can obtain a simpler drift-plus-penalty bound given by

$$\Delta(t) + f(\mathbf{x}(t)) \leq f(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2]$$

With the above drift-plus-penalty bound, we can prove the $O(1/t)$ convergence rate of objective and constraint violations following steps similar to those in Sections 3.3.2 and 3.3.3.

The $O(1/t)$ convergence rate for constrained convex programs with both inequality constraints and linear equality constraints can be established by trivially combining the steps in the previous sections and the steps in this section.

3.4 New Primal-Dual Type Algorithm for Smooth Constrained Convex Programs

In this section, we further assume that the convex program (3.1)-(3.3) has smooth objective and constraint functions, i.e., the following assumption holds.

Assumption 3.3 (Smoothness)**.**

- *Let function $f(\mathbf{x})$ be smooth with modulus L_f, i.e., $\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L_f\|\mathbf{x}_1 - \mathbf{x}_2\|$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$. For each $k \in \{1, 2, \ldots, m\}$, let function $g_k(\mathbf{x})$ be smooth with modulus L_{g_k}, i.e., $\|\nabla g_k(\mathbf{x}_1) - \nabla g_k(\mathbf{x}_2)\| \leq L_{g_k}\|\mathbf{x}_1 - \mathbf{x}_2\|$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$. Denote $\mathbf{L_g} = [L_{g_1}, \ldots, L_{g_m}]^{\mathsf{T}}$.*

Now consider another new Lagrangian method described in Algorithm 3.2. Note that Algorithm 3.2 involves a positive step size sequence $\{\gamma(t), t \geq 0\}$. We consider the following two rules for choosing $\gamma(t)$ in Algorithm 3.2.

- Constant $\gamma(t)$: Choose positive step sizes $\gamma(t)$ via

$$\gamma(t) = \gamma < \frac{1}{\beta^2 + L_f}, \forall t \geq 0 \tag{3.14}$$

- Non-increasing $\gamma(t)$: Choose positive step sizes $\gamma(t)$ via

$$\gamma(t) = \begin{cases} \frac{1}{\beta^2 + L_f + [\mathbf{Q}(0) + \mathbf{g}(\mathbf{x}(-1))]^{\mathsf{T}}\mathbf{L_g}}, & t = 0 \\ \min\left\{\gamma(t-1), \frac{1}{\beta^2 + L_f + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{L_g}}\right\}, & t \geq 1 \end{cases} \tag{3.15}$$

  Note that part 2 of Lemma 3.1 ensures $\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1)) \geq \mathbf{0}, \forall t \geq 0$. Thus, (3.15) ensures $\gamma(t) > 0, \forall t \geq 0$.

Note that Algorithm 3.2 uses the same virtual queue update equation (3.4) used in Algorithm 3.1 but modifies the update of primal variables $\mathbf{x}(t)$ from a minimization problem to a simple projection, which is similar to the primal update in Algorithm 1.2.

Recall that if $f(\mathbf{x})$ or $g_k(\mathbf{x})$ are not separable, the primal update of $\mathbf{x}(t)$ in Algorithm 3.1 is not decomposable and requires to jointly solve a set constrained convex minimization, which can have huge computation complexity especially when the dimension n is large. In contrast, the projection used in Algorithm 3.2 can be distributively implemented as long as the gradient

Algorithm 3.2 New Primal-Dual Type Algorithm for Smooth Constrained Convex Programs

Let $\{\gamma(t), t \geq 0\}$ be a sequence of positive step sizes. Choose any $\mathbf{x}(-1) \in \mathcal{X}$. Initialize $Q_k(0) = \max\{0, -g_k(\mathbf{x}(-1))\}, \forall k \in \{1, 2, \ldots, m\}$. At each iteration $t \in \{0, 1, 2, \ldots\}$, update $\mathbf{x}(t)$ and $\mathbf{Q}(t+1)$ as follows:

- Update primal variables via

$$\mathbf{x}(t) = \mathcal{P}_{\mathcal{X}}\big[\mathbf{x}(t-1) - \gamma(t)\mathbf{d}(t)\big],$$

  where $\mathcal{P}_{\mathcal{X}}[\cdot]$ is the projection onto convex set $\mathcal{X}$ and $\mathbf{d}(t) = \nabla f(\mathbf{x}(t-1)) + \sum_{k=1}^{m}[Q_k(t) + g_k(\mathbf{x}(t-1))]\nabla g_k(\mathbf{x}(t-1))$ is the gradient of function $\phi(\mathbf{x}) = f(\mathbf{x}) + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x})$ at point $\mathbf{x} = \mathbf{x}(t-1)$.

- Update virtual queues via the equation (3.4) in Algorithm 3.1.

- Output the running average $\overline{\mathbf{x}}(t+1)$ given by

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

  as the solution at iteration $t+1$.

---

is known and the set $\mathcal{X}$ is a Cartesian product. Thus, Algorithm 3.2 is suitable for large scale convex programs with non-separable $f(\mathbf{x})$ or $g_k(\mathbf{x})$ since its per-iteration complexity is much less than that in Algorithm 3.1.

For constrained convex programs with non-separable $f(\mathbf{x})$ or $g_k(\mathbf{x})$, the primal update of $\mathbf{x}(t)$ in Algorithm 1.2 also has low complexity since it follows a similar projection update. However, Algorithm 1.2 has a slow $O(1/\epsilon^2)$ convergence time as reviewed in Section 1.2. Another drawback of Algorithm 1.2 is that its implementation requires to know an upper bound of the optimal Lagrange multiplier vector $\boldsymbol{\lambda}^*$ (defined in Assumption 3.2), which is typically unavailable in practice.

In this section, we show that Algorithm 3.2 has the same $O(\frac{1}{\epsilon})$ convergence time as Algorithm 3.1 for the smooth constrained convex programs (3.1)-(3.3) and its implementation does not require any knowledge of the optimal Lagrange multiplier vector $\boldsymbol{\lambda}^*$.

3.4.1 An Upper Bound of the Drift-Plus-Penalty Expression

Since Algorithm 3.2 uses the same virtual queue update equation (3.4) used in Algorithm 3.1, Lemmas 3.1-3.3 proven in Section 3.2 and Lemma 3.6 proven in Section 3.3.3 still hold for Algorithm 3.2. The convergence time analysis of Algorithm 3.2 folllows a structure similar to

that of Algorithm 3.1 and is presented in the remainder of this section. The first key step is to derive an upper bound for the drift-plus-penalty expression under Algorithm 3.2.

Lemma 3.8. *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.3. For all* $t \geq 0$ *in Algorithm 3.2, we have*

$$\begin{aligned}\Delta(t) + f(\mathbf{x}(t)) \leq & f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2] + \frac{1}{2}[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2] \\ & + \frac{1}{2}\big[\beta^2 + L_f + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{L}_{\mathbf{g}} - \frac{1}{\gamma(t)}\big]\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2,\end{aligned}$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$*, both of which are defined in Assumption 3.1; and* L_f *and* $\mathbf{L}_{\mathbf{g}}$ *are defined in Assumption 3.3.*

Proof. Fix $t \geq 0$. The projection operator can be reinterpreted as an optimization problem as follows:

$$\begin{aligned}
& \mathbf{x}(t) = \mathcal{P}_{\mathcal{X}}\left[\mathbf{x}(t-1) - \gamma(t)\mathbf{d}(t)\right] \\
\overset{(a)}{\Leftrightarrow} \quad & \mathbf{x}(t) = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \left\{\left\|\mathbf{x} - [\mathbf{x}(t-1) - \gamma(t)\mathbf{d}(t)]\right\|^2\right\} \\
\Leftrightarrow \quad & \mathbf{x}(t) = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \left\{\|\mathbf{x} - \mathbf{x}(t-1)\|^2 + 2\gamma(t)[\mathbf{d}(t)]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t-1)] + [\gamma(t)]^2\|\mathbf{d}(t)\|^2\right\} \\
\overset{(b)}{\Leftrightarrow} \quad & \mathbf{x}(t) = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \Big\{f(\mathbf{x}(t-1)) + \sum_{k=1}^{m}[Q_k(t) + g_k(\mathbf{x}(t-1))]g_k(\mathbf{x}(t-1)) + \mathbf{d}^{\mathsf{T}}(t)[\mathbf{x} - \mathbf{x}(t-1)] \\
& + \frac{1}{2\gamma(t)}\|\mathbf{x} - \mathbf{x}(t-1)\|^2\Big\} \\
\overset{(c)}{\Leftrightarrow} \quad & \mathbf{x}(t) = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \left\{\phi(\mathbf{x}(t-1)) + [\nabla\phi(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t-1)] + \frac{1}{2\gamma(t)}\|\mathbf{x} - \mathbf{x}(t-1)\|^2\right\},
\end{aligned} \tag{3.16}$$

where (a) follows from the definition of the projection onto a convex set; (b) follows from the fact the minimizing solution does not change when we remove constant term $[\gamma(t)]^2\|\mathbf{d}(t)\|^2$, multiply positive constant $\frac{1}{2\gamma(t)}$ and add constant term $f(\mathbf{x}(t-1)) + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t-1))$ in the objective function; and (c) follows by defining

$$\phi(\mathbf{x}) = f(\mathbf{x}) + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}). \tag{3.17}$$

Note that part 2 in Lemma 3.1 implies that $\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))$ is component-wise nonnegative

for all $k \in \{1, 2, \ldots, m\}$. Hence, function $\phi(\mathbf{x})$ is convex with respect to $\mathbf{x}$ on $\mathcal{X}$.

Since $\frac{1}{2\gamma(t)}\|\mathbf{x} - \mathbf{x}(t-1)\|^2$ is strongly convex with respect to $\mathbf{x}$ with modulus $\frac{1}{\gamma(t)}$, it follows that

$$\phi(\mathbf{x}(t-1)) + [\nabla\phi(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t-1)] + \frac{1}{2\gamma(t)}\|\mathbf{x} - \mathbf{x}(t-1)\|^2$$

is strongly convex with respect to $\mathbf{x}$ with modulus $\frac{1}{\gamma(t)}$.

Since $\mathbf{x}(t)$ is chosen to minimize the above strongly convex function, by Corollary 1.2, we have

$$\begin{aligned}
&\phi(\mathbf{x}(t-1)) + [\nabla\phi(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t) - \mathbf{x}(t-1)] + \frac{1}{2\gamma(t)}\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2 \\
\leq &\phi(\mathbf{x}(t-1)) + [\nabla\phi(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}^* - \mathbf{x}(t-1)] + \frac{1}{2\gamma(t)}\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \frac{1}{2\gamma(t)}\|\mathbf{x}^* - \mathbf{x}(t)\|^2 \\
\overset{(a)}{\leq} &\phi(\mathbf{x}^*) + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2] \\
\overset{(b)}{=} &f(\mathbf{x}^*) + \underbrace{[\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}^*)}_{\leq 0} + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2] \\
\overset{(c)}{\leq} &f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2], \qquad (3.18)
\end{aligned}$$

where (a) follows from the fact that $\phi(\mathbf{x})$ is convex with respect to $\mathbf{x}$ on $\mathcal{X}$; (b) follows from the definition of function $\phi(\mathbf{x})$ in (3.17); and (c) follows by using the fact that $g_k(\mathbf{x}^*) \leq 0$ and $Q_k(t) + g_k(\mathbf{x}(t-1)) \geq 0$ (i.e., part 2 in Lemma 3.1) for all $k \in \{1, 2, \ldots, m\}$ to eliminate the term marked by an underbrace.

Recall that $f(\mathbf{x})$ is smooth on $\mathcal{X}$ with modulus L_f by Assumption 3.3. By Lemma 1.1, we have

$$f(\mathbf{x}(t)) \leq f(\mathbf{x}(t-1)) + [\nabla f(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t) - \mathbf{x}(t-1)] + \frac{L_f}{2}\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2. \qquad (3.19)$$

Recall that each $g_k(\mathbf{x})$ is smooth on $\mathcal{X}$ with modulus L_{g_k} by Assumption 3.3. Thus, $[Q_k(t) +$

$g_k(\mathbf{x}(t-1))]g_k(\mathbf{x})$ is smooth with modulus $[Q_k(t)+g_k(\mathbf{x}(t-1))]L_{g_k}$. By Lemma 1.1, we have

$$\begin{aligned}
&[Q_k(t)+g_k(\mathbf{x}(t-1))]g_k(\mathbf{x}(t)) \\
\leq&[Q_k(t)+g_k(\mathbf{x}(t-1))]g_k(\mathbf{x}(t-1))+[Q_k(t)+g_k(\mathbf{x}(t-1))][\nabla g_k(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t)-\mathbf{x}(t-1)] \\
&+\frac{[Q_k(t)+g_k(\mathbf{x}(t-1))]L_{g_k}}{2}\|\mathbf{x}(t)-\mathbf{x}(t-1)\|^2. \qquad (3.20)
\end{aligned}$$

Summing (3.20) over $k\in\{1,2,\ldots,m\}$ yields

$$\begin{aligned}
&[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) \qquad (3.21) \\
\leq&[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t-1))+\sum_{k=1}^{m}[Q_k(t)+g_k(\mathbf{x}(t-1))][\nabla g_k(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t)-\mathbf{x}(t-1)] \\
&+\frac{[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{L}_{\mathbf{g}}}{2}\|\mathbf{x}(t)-\mathbf{x}(t-1)\|^2. \qquad (3.22)
\end{aligned}$$

Summing up (3.19) and (3.22) together yields

$$\begin{aligned}
&f(\mathbf{x}(t))+[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) \\
\leq&f(\mathbf{x}(t-1))+[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t-1))+[\nabla f(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t)-\mathbf{x}(t-1)] \\
&+\sum_{k=1}^{m}[Q_k(t)+g_k(\mathbf{x}(t-1))][\nabla g_k(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t)-\mathbf{x}(t-1)] \\
&+\frac{L_f+[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{L}_{\mathbf{g}}}{2}\|\mathbf{x}(t)-\mathbf{x}(t-1)\|^2 \\
\overset{(a)}{=}&\phi(\mathbf{x}(t-1))+[\nabla\phi(\mathbf{x}(t-1))]^{\mathsf{T}}[\mathbf{x}(t)-\mathbf{x}(t-1)]+\frac{L_f+[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{L}_{\mathbf{g}}}{2}\|\mathbf{x}(t)-\mathbf{x}(t-1)\|^2, \qquad (3.23)
\end{aligned}$$

where (a) follows from the definition of function $\phi(\mathbf{x})$ in (3.17).

Substituting (3.18) into (3.23) yields

$$\begin{aligned}
&f(\mathbf{x}(t))+[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{g}(\mathbf{x}(t)) \\
\leq&f(\mathbf{x}^*)+\frac{1}{2\gamma(t)}[\|\mathbf{x}^*-\mathbf{x}(t-1)\|^2-\|\mathbf{x}^*-\mathbf{x}(t)\|^2] \\
&+\frac{1}{2}\big[L_f+[\mathbf{Q}(t)+\mathbf{g}(\mathbf{x}(t-1))]^{\mathsf{T}}\mathbf{L}_{\mathbf{g}}-\frac{1}{\gamma(t)}\big]\|\mathbf{x}(t)-\mathbf{x}(t-1)\|^2. \qquad (3.24)
\end{aligned}$$

Note that $\mathbf{u}_1^\mathsf{T}\mathbf{u}_2 = \frac{1}{2}[\|\mathbf{u}_1\|^2 + \|\mathbf{u}_2\|^2 - \|\mathbf{u}_1 - \mathbf{u}_2\|^2]$ for any $\mathbf{u}_1, \mathbf{u}_2 \in \mathbb{R}^m$. Thus, we have

$$[\mathbf{g}(\mathbf{x}(t-1))]^\mathsf{T}\mathbf{g}(\mathbf{x}(t)) = \frac{1}{2}[\|\mathbf{g}(\mathbf{x}(t-1))\|^2 + \|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}(t))\|^2]. \quad (3.25)$$

Substituting (3.25) into (3.24) and rearranging terms yields

$$\begin{aligned}
&f(\mathbf{x}(t)) + [\mathbf{Q}(t)]^\mathsf{T}\mathbf{g}(\mathbf{x}(t)) \\
\leq& f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2] - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2 \\
&+ \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}(t))\|^2 + \frac{1}{2}\big[L_f + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^\mathsf{T}\mathbf{L}_\mathbf{g} - \frac{1}{\gamma(t)}\big]\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2 \\
\overset{(a)}{\leq}& f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2] - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2 \\
&+ \frac{1}{2}\big[\beta^2 + L_f + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^\mathsf{T}\mathbf{L}_\mathbf{g} - \frac{1}{\gamma(t)}\big]\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2,
\end{aligned}$$

where (a) follows from the fact that $\|\mathbf{g}(\mathbf{x}(t-1)) - \mathbf{g}(\mathbf{x}(t))\| \leq \beta\|\mathbf{x}(t) - \mathbf{x}(t-1)\|$, which further follows from the assumption that $\mathbf{g}(\mathbf{x})$ is Lipschitz continuous with modulus β.

Summing (3.7) to the above inequality and cancelling the common terms on both sides yields

$$\begin{aligned}
&\Delta(t) + f(\mathbf{x}(t)) \\
\leq& f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2] + \frac{1}{2}[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2] \\
&+ \frac{1}{2}\big[\beta^2 + L_f + [\mathbf{Q}(t) + \mathbf{g}(\mathbf{x}(t-1))]^\mathsf{T}\mathbf{L}_\mathbf{g} - \frac{1}{\gamma(t)}\big]\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2.
\end{aligned}$$

□

3.4.2 Smooth Constrained Convex Programs with Linear $\mathbf{g}(\mathbf{x})$

This subsection shows that if each $g_k(\mathbf{x})$ is a linear function, then it suffices to choose constant parameters $\gamma(t) = \gamma < \frac{1}{\beta^2 + L_f}$ in Algorithm 3.2 to solve the smooth constrained convex program (3.1)-(3.3) with an $O(1/\epsilon)$ convergence time.

The next corollary follows directly from Lemma 3.8 by noting that $\mathbf{L}_\mathbf{g} = \mathbf{0}$ when each $g_k(\mathbf{x})$ is a linear function.

Corollary 3.1. *Consider the convex program* (3.1)-(3.3) *where each* $g_k(\mathbf{x})$ *is a linear function under Assumptions 3.1-3.3. If we choose* $\gamma(t)$ *according to* (3.14) *in Algorithm 3.2, i.e.,* $\gamma(t) =$

$\gamma < \frac{1}{\beta^2+L_f}$, *then for all* $t \geq 0$, *we have*

$$\begin{aligned}
&\Delta(t) + f(\mathbf{x}(t)) \\
\leq & f(\mathbf{x}^*) + \frac{1}{2\gamma}\big[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2\big]
\end{aligned}$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *and* β *is the Lipschitz modulus of* $\mathbf{g}(\mathbf{x})$, *both of which are defined in Assumptions 3.1; and* L_f *is the constant defined in Assumption 3.3.*

Proof. Note that if each $g_k(\mathbf{x})$ is a linear function, then we have $\mathbf{L_g} = \mathbf{0}$. Fix $t \geq 0$. By Lemma 3.8 with $\gamma(t) = \gamma < \frac{1}{\beta^2+L_f}$ and $\mathbf{L_g} = \mathbf{0}$, we have

$$\begin{aligned}
&\Delta(t) + f(\mathbf{x}(t)) \\
\leq & f(\mathbf{x}^*) + \frac{1}{2\gamma}\big[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2\big] \\
& + \frac{1}{2}\big(\beta^2 + L_f - \frac{1}{\gamma}\big)\|\mathbf{x}(t) - \mathbf{x}(t-1)\|^2 \\
\overset{(a)}{\leq} & f(\mathbf{x}^*) + \frac{1}{2\gamma}\big[\|\mathbf{x}^* - \mathbf{x}(t-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(t))\|^2 - \|\mathbf{g}(\mathbf{x}(t-1))\|^2\big]
\end{aligned}$$

where (a) follows from $\gamma < \frac{1}{\beta^2+L_f}$. □ □

Theorem 3.4. *Consider the convex program* (3.1)-(3.3) *where each* $g_k(\mathbf{x})$ *is a linear function under Assumptions 3.1- 3.3. Let* $\mathbf{x}^*$ *be an optimal solution. Let* $\boldsymbol{\lambda}^*$ *be a Lagrange multiplier vector satisfying Assumption 3.2. If we choose* $\gamma(t)$ *according to* (3.14) *in Algorithm 3.2, then for all* $t \geq 1$, *we have*

1. $$f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*) + \frac{1}{2\gamma t}\|\mathbf{x}^* - \mathbf{x}(-1)\|^2.$$

2. $$g_k(\overline{\mathbf{x}}(t)) \leq \frac{1}{t}\left(2\|\boldsymbol{\lambda}^*\| + \sqrt{\frac{1}{\gamma}}\|\mathbf{x}^* - \mathbf{x}(-1)\| + \sqrt{\frac{\frac{1}{\gamma}}{\frac{1}{\gamma} - \beta^2}}\|\mathbf{g}(\mathbf{x}^*)\|\right).$$

where $\mathbf{x}^*$ *is an optimal solution of the problem* (3.1)-(3.3) *defined in Assumptions 3.1;* $\boldsymbol{\lambda}^*$ *is a Lagrange multiplier vector satisfying Assumption 3.2; and* L_f *is the constant defined in Assumption 3.3. That is, Algorithm 3.2 ensures error decays like* $O(1/t)$ *and provides an* ϵ*-approximate solution with convergence time* $O(1/\epsilon)$.

Proof. Note that Corollary 3.1 provides a drift-plus-penalty bound similar to the one derived in Lemma 3.4 for Algorithm 3.1. The only modification is replacing α with $\frac{1}{2\gamma}$. Following the same proof steps in Sections 3.3.2-3.3.4, we can prove the current theorem. □

3.4.3 Smooth Constrained Convex Programs with Non-Linear $\mathbf{g}(\mathbf{x})$

For the smooth constrained convex program (3.1)-(3.3) with possibly nonlinear $\mathbf{g}(\mathbf{x})$, the following assumption is further assumed:

Assumption 3.4.

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

This subsection proves that if the convex program (3.1)-(3.3) with possibly nonlinear $\mathbf{g}(\mathbf{x})$ satisfies Assumptions 3.1-3.4, then it suffices to choose non-increasing step sizes $\gamma(t)$ according to (3.15) in Algorithm 3.2 to solve the convex program (3.1)-(3.3) with an $O(1/\epsilon)$ convergence time.

Lemma 3.9. *Consider the convex program* (3.1)-(3.3) *under Assumptions 3.1-3.4. If we choose non-increasing $\gamma(t)$ in Algorithm 3.2 according to* (3.15)*, then we have*

1. $\sum_{\tau=0}^{t} \frac{1}{2\gamma(\tau)} \left[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|^2\right] \leq \frac{1}{2\gamma(t)} R^2, \forall t \geq 0$;
2. $\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq t f(\mathbf{x}^*) + \frac{1}{2\gamma(t-1)} R^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2, \forall t \geq 1$;
3. $\|\mathbf{Q}(t+1)\| \leq 2\|\boldsymbol{\lambda}^*\| + R\sqrt{\frac{1}{\gamma(t)}} + C, \forall t \geq 0$;

where $\mathbf{x}^*$ is an optimal solution of the problem (3.1)-(3.3) *defined in Assumptions 3.1; $\boldsymbol{\lambda}^*$ is a Lagrange multiplier vector satisfying Assumption 3.2; and R and C are constants defined in Assumption 3.4.*

Proof.

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

$$\begin{aligned}
&\sum_{\tau=0}^{t} \frac{1}{2\gamma(\tau)} \big[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|^2\big] \\
=&\frac{1}{2\gamma(0)}\|\mathbf{x}^* - \mathbf{x}(-1)\|^2 + \sum_{\tau=0}^{t-1}[\frac{1}{2\gamma(\tau+1)} - \frac{1}{2\gamma(\tau)}]\|\mathbf{x}^* - \mathbf{x}(\tau)\|^2 - \frac{1}{2\gamma(t)}\|\mathbf{x}^* - \mathbf{x}(t)\|^2 \\
\overset{(a)}{\leq}&\frac{1}{2\gamma(0)}R^2 + \sum_{\tau=0}^{t-1}[\frac{1}{2\gamma(\tau+1)} - \frac{1}{2\gamma(\tau)}]R^2 \\
=&\frac{1}{2\gamma(t)}R^2
\end{aligned}$$

   where (a) follows because $\|\mathbf{x}^* - \mathbf{x}(\tau)\| \leq R, \forall \tau \geq 0$ by Assumption 3.4 and $\gamma(\tau+1) \leq \gamma(\tau), \forall \tau \geq 0$ by (3.15).

2. Fix $t \geq 1$. By Lemma 3.8, for all $\tau \in \{0, 1, 2, \ldots\}$, we have

$$\begin{aligned}
&\Delta(\tau) + f(\mathbf{x}(\tau)) \\
\leq& f(\mathbf{x}^*) + \frac{1}{2\gamma(\tau)}\big[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(\tau))\|^2 - \|\mathbf{g}(\mathbf{x}(\tau-1))\|^2\big] \\
&+ \frac{1}{2}\big[\beta^2 + L_f + [\mathbf{Q}(\tau) + \mathbf{g}(\mathbf{x}(\tau-1))]^\intercal \mathbf{L}_\mathbf{g} - \frac{1}{\gamma(\tau)}\big]\|\mathbf{x}(\tau) - \mathbf{x}(\tau-1)\|^2 \\
\overset{(a)}{\leq}& f(\mathbf{x}^*) + \frac{1}{2\gamma(\tau)}\big[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|^2\big] + \frac{1}{2}\big[\|\mathbf{g}(\mathbf{x}(\tau))\|^2 - \|\mathbf{g}(\mathbf{x}(\tau-1))\|^2\big]
\end{aligned}$$

   where (a) follows because each $\gamma(\tau)$ chosen according to (3.15) ensures $\beta^2 + L_f + [\mathbf{Q}(\tau) + \mathbf{g}(\mathbf{x}(\tau-1))]^\intercal \mathbf{L}_\mathbf{g} - \frac{1}{\gamma(\tau)} \leq 0$.

   Summing over $\tau \in \{0, 1, 2, \ldots, t-1\}$ and rearranging terms yields

$$\begin{aligned}
&\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \\
\leq& t f(\mathbf{x}^*) + \sum_{\tau=0}^{t-1} \frac{1}{2\gamma(\tau)}\big[\|\mathbf{x}^* - \mathbf{x}(\tau-1)\|^2 - \|\mathbf{x}^* - \mathbf{x}(\tau)\|^2\big] \\
&+ \frac{1}{2}\sum_{\tau=0}^{t-1}\big[\|\mathbf{g}(\mathbf{x}(\tau))\|^2 - \|\mathbf{g}(\mathbf{x}(\tau-1))\|^2\big] - \sum_{\tau=0}^{t-1}\Delta(\tau) \\
\overset{(a)}{\leq}& t f(\mathbf{x}^*) + \frac{1}{2\gamma(t-1)}R^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{g}(\mathbf{x}(-1))\|^2 + \frac{1}{2}\|\mathbf{Q}(0)\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\
\overset{(b)}{\leq}& t f(\mathbf{x}^*) + \frac{1}{2\gamma(t-1)}R^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2
\end{aligned}$$

where (a) follows from part 1 of this lemma and by recalling that $\Delta(\tau) = \frac{1}{2}\|\mathbf{Q}(\tau+1)\|^2 - \frac{1}{2}\|\mathbf{Q}(\tau)\|^2$; and (b) follows because $\|\mathbf{Q}(0)\|^2 \leq \|\mathbf{g}(\mathbf{x}(-1))\|^2$ by part 3 in Lemma 3.1.

3. By part 2 of this lemma, we have

$$\begin{aligned}\sum_{\tau=0}^{t} f(\mathbf{x}(\tau)) \leq & (t+1)f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}R^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t))\|^2 - \frac{1}{2}\|\mathbf{Q}(t+1)\|^2 \\ \leq & (t+1)f(\mathbf{x}^*) + \frac{1}{2\gamma(t)}R^2 + \frac{1}{2}C^2 - \frac{1}{2}\|\mathbf{Q}(t+1)\|^2 \end{aligned} \tag{3.26}$$

where (a) follows from $\|\mathbf{g}(\mathbf{x}(t))\| \leq C$ by Assumption 3.4. By Lemma 3.6, we have

$$\sum_{\tau=0}^{t} f(\mathbf{x}(\tau)) \geq (t+1)f(\mathbf{x}^*) - \|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t+1)\| \tag{3.27}$$

Combining (3.26) and (3.27), cancelling common terms and rearranging terms yields

$$\begin{aligned} & \frac{1}{2}\|\mathbf{Q}(t+1)\|^2 - \|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t+1)\| - \frac{1}{2\gamma(t)}R^2 - \frac{1}{2}C^2 \leq 0 \\ \Rightarrow & \Big[\|\mathbf{Q}(t+1)\| - \|\boldsymbol{\lambda}^*\|\Big]^2 \leq \|\boldsymbol{\lambda}^*\|^2 + \frac{1}{\gamma(t)}R^2 + C^2 \\ \Rightarrow & \|\mathbf{Q}(t+1)\| \leq \|\boldsymbol{\lambda}^*\| + \sqrt{\|\boldsymbol{\lambda}^*\|^2 + \frac{1}{\gamma(t)}R^2 + C^2} \\ \overset{(a)}{\Rightarrow} & \|\mathbf{Q}(t+1)\| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{\frac{1}{\gamma(t)}}R + C \end{aligned} \tag{3.28}$$

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 3.10. *Consider the convex program* (3.1)-(3.3) *with possibly nonlinear* $g_k(\mathbf{x})$ *under Assumptions 3.1-3.4. If we choose non-increasing* $\gamma(t)$ *according to* (3.15) *in Algorithm 3.2, then*

$$\gamma(t) \geq \gamma^{\min}, \forall t \geq 0$$

with constant

$$\gamma^{\min} = \frac{1}{\left(\sqrt{\beta^2 + L_f + 2\|\boldsymbol{\lambda}^*\|\|\mathbf{L_g}\| + 2C\|\mathbf{L_g}\|} + R\|\mathbf{L_g}\|\right)^2} \tag{3.29}$$

where $\beta, \boldsymbol{\lambda}^*, L_f, \mathbf{L_g}$ R and C are constants defined in Assumptions 3.1-3.4.

Proof. This lemma can be proven by induction as follows. By (3.15), we have

$$\begin{aligned}\gamma(0) =& \frac{1}{\beta^2 + L_f + [\mathbf{Q}(0) + \mathbf{g}(\mathbf{x}(-1))]^\mathsf{T}\mathbf{L_g}} \\ \overset{(a)}{\geq}& \frac{1}{\beta^2 + L_f + \|\mathbf{Q}(0) + \mathbf{g}(\mathbf{x}(-1))\|\|\mathbf{L_g}\|} \\ \overset{(b)}{\geq}& \frac{1}{\beta^2 + L_f + 2C\|\mathbf{L_g}\|} \\ \geq& \gamma^{\min}\end{aligned}$$

where (a) follows from the Cauchy-Schwarz inequality; and (b) follows from $\|\mathbf{Q}(0)+\mathbf{g}(\mathbf{x}(-1))\| \leq \|\mathbf{Q}(0)\| + \|\mathbf{g}(\mathbf{x}(-1))\| \leq 2\|\mathbf{g}(\mathbf{x}(-1))\| \leq 2C$ where the second inequality follows from part 3 of Lemma 3.1 and the third inequality follows from Assumption 3.4. Thus, we have $\gamma(0) \geq \gamma^{\min}$.

Now assume $\gamma(t) \geq \gamma^{\min}$ holds for $t = t_0$ and consider $t = t_0 + 1$. By (3.15), $\gamma(t_0 + 1)$ is given by

$$\gamma(t_0 + 1) = \min\left\{\gamma(t_0), \frac{1}{\beta^2 + L_f + [\mathbf{Q}(t_0 + 1) + \mathbf{g}(\mathbf{x}(t_0))]^\mathsf{T}\mathbf{L_g}}\right\}$$

Since $\gamma(t_0) \leq \gamma^{\min}$ by the induction hypothesis, to prove $\gamma(t_0 + 1) \geq \gamma^{\min}$, it remains to prove

$$\frac{1}{\beta^2 + L_f + [\mathbf{Q}(t_0 + 1) + \mathbf{g}(\mathbf{x}(t_0))]^\mathsf{T}\mathbf{L_g}} \geq \gamma^{\min}$$

By part 3 of Lemma 3.9, we have

$$\begin{aligned}\|\mathbf{Q}(t_0 + 1)\| \leq& 2\|\boldsymbol{\lambda}^*\| + R\sqrt{\frac{1}{\gamma(t_0)}} + C \\ \overset{(a)}{\leq}& 2\|\boldsymbol{\lambda}^*\| + R\sqrt{\frac{1}{\gamma^{\min}}} + C \end{aligned} \tag{3.30}$$

where (a) follows the hypothesis in the induction. Thus, we have

$$
\begin{aligned}
&\frac{1}{\beta^2+L_f+[\mathbf{Q}(t_0+1)+\mathbf{g}(\mathbf{x}(t_0))]^{\mathsf{T}}\mathbf{L_g}}\\
\overset{(a)}{\geq}&\frac{1}{\beta^2+L_f+\|\mathbf{Q}(t_0+1)+\mathbf{g}(\mathbf{x}(t_0))\|\|\mathbf{L_g}\|}\\
\overset{(b)}{\geq}&\frac{1}{\beta^2+L_f+\|\mathbf{Q}(t_0+1)\|\|\mathbf{L_g}\|+\|\mathbf{g}(\mathbf{x}(t_0))\|\|\mathbf{L_g}\|}\\
\overset{(c)}{\geq}&\frac{1}{\beta^2+L_f+\big[2\|\boldsymbol{\lambda}^*\|+R\sqrt{\frac{1}{\gamma^{\min}}}+C\big]\|\mathbf{L_g}\|+C\|\mathbf{L_g}\|}\\
=&\frac{1}{\beta^2+L_f+2\|\boldsymbol{\lambda}^*\|\|\mathbf{L_g}\|+2C\|\mathbf{L_g}\|+R\|\mathbf{L_g}\|\sqrt{\frac{1}{\gamma^{\min}}}}\\
\overset{(d)}{=}&\frac{1}{\beta^2+L_f+2\|\boldsymbol{\lambda}^*\|\|\mathbf{L_g}\|+2C\|\mathbf{L_g}\|+(R\|\mathbf{L_g}\|)^2+R\|\mathbf{L_g}\|\sqrt{\beta^2+L_f+2\|\boldsymbol{\lambda}^*\|\|\mathbf{L_g}\|+2C\|\mathbf{L_g}\|}}\\
\overset{(e)}{\geq}&\frac{1}{\big(\sqrt{\beta^2+L_f+2\|\boldsymbol{\lambda}^*\|\|\mathbf{L_g}\|+2C\|\mathbf{L_g}\|}+R\|\mathbf{L_g}\|\big)^2}\\
=&\gamma^{\min}
\end{aligned}
$$

where (a) follows from the Cauchy-Schwarz inequality; (b) follows from the triangle inequality; (c) follows from (3.30) and $\|\mathbf{g}(\mathbf{x}(t_0))\| \leq C$ by Assumption 3.4; (d) follows by substituting $\gamma^{\min} = \frac{1}{\left[\sqrt{\beta^2+L_f+2\|\boldsymbol{\lambda}^*\|\|\mathbf{L_g}\|+2C\|\mathbf{L_g}\|}+R\|\mathbf{L_g}\|\right]^2}$; and (e) follow from the basic inequality $z_1^2+z_2^2+z_1z_2 \leq (z_1+z_2)^2$ for any $z_1, z_2 \geq 0$.

Thus, we have $\gamma(t_0+1) \geq \gamma^{\min}$. This lemma follows by induction. □ □

The next theorem summarizes the $O(1/\epsilon)$ convergence time of Algorithm 3.2 for the smooth constrained convex program (3.1)-(3.3) with possibly nonlinear $g_k(\mathbf{x})$.

Theorem 3.5. *Consider the convex program* (3.1)-(3.3) *with possibly nonlinear* $g_k(\mathbf{x})$ *under Assumptions 3.1- 3.4. Let* $\mathbf{x}^*$ *be an optimal solution and* $\boldsymbol{\lambda}^*$ *be a Lagrange multiplier vector satisfying Assumption 3.2. If we choose non-increasing* $\gamma(t)$ *according to* (3.15) *in Algorithm 3.2, then for all* $t \geq 1$*, we have*

1.
$$f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*)+\frac{1}{t}\frac{R^2}{2\gamma^{\min}}.$$

2.
$$g_k(\overline{\mathbf{x}}(t)) \leq \frac{1}{t}\big(\|\boldsymbol{\lambda}^*\|+R\sqrt{\frac{1}{\gamma^{\min}}}+C\big), \forall k \in \{1,2,\ldots,m\}.$$

where $\gamma^{\min}$ is the constant defined in Lemma 3.10; $\mathbf{x}^*$ is an optimal solution of the problem (3.1)-(3.3) *defined in Assumptions 3.1; $\boldsymbol{\lambda}^*$ is a Lagrange multiplier vector satisfying Assumption 3.2; and R and C are constants defined in Assumption 3.4. That is, Algorithm 3.2 ensures error decays like $O(1/t)$ and provides an ϵ-approximate solution with convergence time $O(1/\epsilon)$.*

Proof.

1. Fix $t \geq 1$. By part 2 of Lemma 3.9, we have

$$\begin{aligned}\sum_{\tau=0}^{t-1} f(\mathbf{x}(\tau)) \leq & t f(\mathbf{x}^*) + \frac{1}{2\gamma(t-1)} R^2 + \frac{1}{2}\|\mathbf{g}(\mathbf{x}(t-1))\|^2 - \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\ \overset{(a)}{\leq} & t f(\mathbf{x}^*) + \frac{1}{2\gamma^{\min}} R^2\end{aligned}$$

   where (a) follows from $\gamma(t-1) \geq \gamma^{\min}$ by Lemma 3.10 and $\|\mathbf{Q}(t)\| \geq \|\mathbf{g}(\mathbf{x}(t-1))\|$ by Lemma 3.1.

   Recall that $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbf{x}(\tau)$. Dividing both sides by t and using Jensen's inequality for convex function $f(\mathbf{x})$ yields $f(\overline{\mathbf{x}}(t)) \leq f(\mathbf{x}^*) + \frac{1}{t}\frac{R^2}{2\gamma^{\min}}$.

2. Fix $t \geq 1$ and $k \in \{1, 2, \ldots, m\}$. Recall that $\overline{\mathbf{x}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbf{x}(\tau)$. Thus,

$$\begin{aligned} g_k(\overline{\mathbf{x}}(t)) \overset{(a)}{\leq} & \frac{1}{t}\sum_{\tau=0}^{t-1} g_k(\mathbf{x}(\tau)) \\ \overset{(b)}{\leq} & \frac{g_k(t)}{t} \\ \leq & \frac{\|\mathbf{Q}(t)\|}{t} \\ \overset{(c)}{\leq} & \frac{1}{t}\big(2\|\boldsymbol{\lambda}^*\| + R\sqrt{\frac{1}{\gamma^{\min}}} + C\big),\end{aligned}$$

   where (a) follows from the convexity of $g_k(\mathbf{x}), k \in \{1, 2, \ldots, m\}$ and Jensen's inequality; (b) follows from Lemma 3.2; and (c) follows because $\|\mathbf{Q}(t)\| \leq 2\|\boldsymbol{\lambda}^*\| + R\sqrt{\frac{1}{\gamma(t-1)}} + C$ by part 3 of Lemma 3.9 and $\gamma(t-1) \geq \gamma^{\min}$ by Lemma 3.10.

□ □

3.5 Chapter Summary

This chapter considers two new Lagrangian methods to constrained solve convex programs. The first algorithm can solve general convex programs with possibly non-differentiable objective or constraint functions. and has a parallel implementation when the objective and constraint functions are separable. The second algorithm can solve convex programs with smooth objective and constraint functions. At each iteration, the second algorithm updates the primal variable $\mathbf{x}(t)$ using a simple projected gradient update, which can be distributively implemented even if the objective or constraint functions are not separable. Both algorithms are proven to have a fast $O(\frac{1}{\epsilon})$ convergence time.

Chapter 4

New Backpressure Algorithms for Joint Rate Control and Routing

In multi-hop data networks, the problem of joint rate control and routing is to accept data into the network to maximize certain utilities and to make routing decisions at each node such that all accepted data are delivered to intended destinations without overflowing any queue in intermediate nodes. The original backpressure algorithm proposed in the seminal work [TE92] by Tassiulas and Ephremides addresses this problem by assuming that incoming data are given and are inside the network stability region and develops a routing strategy to deliver all incoming data without overflowing any queue. In the context of [TE92], there is essentially no utility maximization consideration in the network. The backpressure algorithm is further extended by a drift-plus-penalty technique to deal with both utility maximization and queue stability [Nee03, GNT06, Nee10]. Alternative extensions for both utility maximization and queue stabilization are developed in [ES06, Sto05, LS04, LMS06]. The above extended backpressure algorithms have different dynamics and/or may yield different utility-delay tradeoff results. However, all of them rely on "backpressure" quantities, which are the differential backlogs between neighboring nodes.

It has been observed in [NMR05, ES06, LS04, LSXS15] that the drift-plus-penalty and other alternative algorithms can be interpreted as first order Lagrangian methods for constrained optimization. In addition, these backpressure algorithms follow certain fundamental utility-delay tradeoffs. For instance, the primal-dual type backpressure algorithm in [ES06] achieves an $O(\epsilon)$ utility optimality gap with an $O(1/\epsilon^2)$ queue length. That is, a small utility optimality gap (corresponding to a small ϵ) is available only at the cost of a large queue length. The drift-plus-penalty backpressure algorithm [Nee10], which has the best utility-delay tradeoff among all

existing first order Lagrangian methods for general networks, can only achieve an $O(\epsilon)$ utility optimality gap with an $O(1/\epsilon)$ queue length. Under certain restrictive assumptions over the network, a better $[O(\epsilon), O(\log(1/\epsilon))]$ tradeoff is achieved via an exponential Lyapunov function in [Nee06], and an $[O(\epsilon), O(\log^2(1/\epsilon))]$ tradeoff is achieved via a LIFO-backpressure algorithm in [HMNK13].

Fundamental lower bounds on utility-delay tradeoffs in [BG02, Nee07, ES12, Nee16, Nee06] show that, for various stochastic network settings, a large queue delay is unavoidable if a small utility optimality gap is demanded. These works consider certain hard problems with stochastic behavior. It leaves open the question of whether or not performance can be improved for networks that fall outside these hard cases. The current chapter investigates network flow problems that can be written as (deterministic) convex programs, which are not restricted to the prior lower bounds. We pursue the question of whether or not improved tradeoffs are possible. Can optimal utility be approached with constant queue sizes?

Recently, there have been many attempts in obtaining new variations of backpressure algorithms for deterministic network flow problems by applying Newton's method to the Lagrangian dual function. In the recent work [LSXS15], the authors develop a Newton's method for joint rate control and routing. However, the utility-delay tradeoff in [LSXS15] is still $[O(\epsilon), O(1/\epsilon^2)]$; and the algorithm requires a centralized projection step although Newton directions can be approximated in a distributed manner. Work [WOJ13] considers a network flow control problem where the path of each flow is given (and hence there is no routing part in the problem), and proposes a decentralized Newton based algorithm for rate control. Work [ZRJ13] considers network routing without an end-to-end utility and only shows the stability of the proposed Newton based backpressure algorithm. All of the above Netwon's method based algorithms rely on distributed approximations for the inverse of Hessians, whose computations still require certain coordinations for the local information updates and propagations and do not scale well with the network size. In contrast, the first order Lagrangian methods do not need global network topology information. Rather, each node only needs the queue length information of its neighbors.

In this chapter, we propose two new backpressure algorithms that are as simple as the existing algorithms in [Nee10, ES06, LS04] but have a better utility-delay tradeoff. The first new backpressure algorithm is originally developed in our paper [YN17b] and the second new backpressue algorithm is developed in our technical report [YN17c]. The first algorithm is almost a

straightforward application of Algorithm 3.1, the new Lagrangian method developed in Chapter 3 for constrained convex programs, to the network utility maximization with node flow balance constraints. However, this backpressure algorithm involves a global algorithm parameter that depends on the number of sessions and the number of links in the underlying network. The second algorithm is developed by adapting the general Lagrangian method developed in Chapter 3 for the specific network optimization problem such that only local algorithm parameters, which can be locally determined by each node, are used.

The new backpressue algorithms achieve a vanishing utility optimality gap that decays like $O(1/t)$, where t is the number of iterations. They also guarantee that the queue length at each node is always bounded by a fixed constant of the same order as the optimal Lagrange multiplier of the network optimization problem. This improves on the utility-delay tradeoffs of prior work. In particular, it improves the steady-state $[O(\epsilon), O(1/\epsilon^2)]$ utility-delay tradeoff in [ES06] and the $[O(\epsilon), O(1/\epsilon)]$ utility-delay tradeoff of the drift-plus-penalty algorithm in [Nee10], both of which yield an unbounded queue length to have a vanishing utility optimality gap. Indeed, the steady-state utility-delay tradeoff of our algorithm is $[0, O(1)]$. They are the first algorithms to achieve zero utility gap and finite queue lengths for joint rate control and routing in multi-hop data networks. The convergence time to reach this limiting performance is also faster than prior work.

The new backpressure algorithms differ from existing first order backpressure algorithms in the following aspects:

1. The "backpressure" quantities in this paper are with respect to newly introduced weights. These are different from queues used in other backpressure algorithms, but can still be locally tracked and updated.

2. The rate control and routing decision rule involves a quadratic term that is similar to a term used in proximal algorithms [PB13].

Note that the benefit of introducing a quadratic term in network optimization has been observed in [LS06]. Work [LS06] developed a distributive rate control algorithm for network utility maximization (NUM) problems with given routing paths that can be reformulated as a special case of the problem treated in this paper. The algorithm of [LS06] considers a fixed set of predetermined paths for each session and does not scale well when treating all (typically

exponentially many) possible paths of a general network. The algorithm proposed in [LS06] is not a backpressure type and does not have queue length or convergence time guarantees. The source session rates yielded during the execution of that algorithm can violate link capacity constraints and hence are infeasible before convergence.

4.1 System Model and Problem Formulation

Consider a slotted data network with normalized time slots $t \in \{0, 1, 2, \ldots\}$. This network is represented by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{L} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of directed links. Let $|\mathcal{N}| = N$ and $|\mathcal{L}| = L$. This network is shared by F end-to-end sessions denoted by a set $\mathcal{F}$. For each end-to-end session $f \in \mathcal{F}$, the source node $\text{Src}(f)$ and destination node $\text{Dst}(f)$ are given but the routes are not specified. Each session f has a continuous and concave utility function $U_f(x_f)$ that represents the "satisfaction" received by accepting x_f amount of data for session f into the network at each slot. Unlike [ES06, LSXS15] where $U_f(\cdot)$ is assumed to be differentiable and strongly concave, this paper considers general concave utility functions $U_f(\cdot)$, including those that are neither differentiable nor strongly concave. Formally, each utility function U_f is defined over an interval $\text{dom}(U_f)$, called the *domain* of the function. It is assumed throughout that either $\text{dom}(U_f) = [0, \infty)$ or $\text{dom}(U_f) = (0, \infty)$, the latter being important for *proportionally fair utilities* [KMT98] $U_f(x) = \log(x)$ that have singularities at $x = 0$.

Denote the capacity of link l as C_l and assume it is a fixed and positive constant.[1] Define $\mu_l^{(f)}$ as the amount of session f's data routed at link l that is to be determined by our algorithm. Note that in general, the network may be configured such that some session f is forbidden to use link l. For each link l, define $\mathcal{S}_l \subseteq \mathcal{F}$ as the set of sessions that are allowed to use link l. The case of unrestricted routing is treated by defining $\mathcal{S}_l = \mathcal{F}$ for all links l.

Note that if $l = (n, m)$ with $n, m \in \mathcal{N}$, then $\mu_l^{(f)}$ and C_l can also be respectively written as $\mu_{(n,m)}^{(f)}$ and $C_{(n,m)}$. For each node $n \in \mathcal{N}$, denote the sets of its incoming links and outgoing links as $\mathcal{I}(n)$ and $\mathcal{O}(n)$, respectively. Note that $x_f, \forall f \in \mathcal{F}$ and $\mu_l^{(f)}, \forall l \in \mathcal{L}, \forall f \in \mathcal{F}$ are the decision variables of a joint rate control and routing algorithm. If the global network topology information is available, the optimal joint rate control and routing can be formulated as the

[1] As stated in [LSXS15], this is a suitable model for wireline networks and wireless networks with fixed transmission power and orthogonal channels.

following multi-commodity network flow problem:

$$\max_{x_f, \mu_l^{(f)}} \sum_{f \in \mathcal{F}} U_f(x_f) \tag{4.1}$$

$$\text{s.t.} \quad x_f \mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l \in \mathcal{I}(n)} \mu_l^{(f)} \leq \sum_{l \in \mathcal{O}(n)} \mu_l^{(f)}, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus \{\text{Dst}(f)\} \tag{4.2}$$

$$\sum_{f \in \mathcal{F}} \mu_l^{(f)} \leq C_l, \forall l \in \mathcal{L}, \tag{4.3}$$

$$\mu_l^{(f)} \geq 0, \forall l \in \mathcal{L}, \forall f \in \mathcal{S}_l, \tag{4.4}$$

$$\mu_l^{(f)} = 0, \forall l \in \mathcal{L}, \forall f \in \mathcal{F} \setminus \mathcal{S}_l, \tag{4.5}$$

$$x_f \in \text{dom}(U_f), \forall f \in \mathcal{F} \tag{4.6}$$

where $\mathbf{1}_{\{\cdot\}}$ is an indicator function; (4.2) represents the node flow conservation constraints relaxed by replacing equalities with inequalities, meaning that the total rate of flow f into node n is less than or equal to the total rate of flow f out of the node (since, in principle, we can always send fake data for departure links when the inequality is loose); and (4.3) represents link capacity constraints. Note that for each flow f, there is no constraint (4.2) at its destination node $\text{Dst}(f)$ since all incoming data are consumed by this node.

The above formulation includes network utility maximization with fixed paths as special cases. In the case when each session only has one single given path, e.g., the network utility maximization problem considered in [LL99], we could modify the sets $\mathcal{S}_l$ used in constraints (4.4) and (4.5) to reflect this fact. For example, if link l_1 is only used for sessions f_1 and f_2, then $\mathcal{S}_{l_1} = \{f_1, f_2\}$. Similarly, the case [LS06] where each flow is restricted to using links from a set of predefined paths can be treated by modifying the sets $\mathcal{S}_l$ accordingly. See Section 4.6.1 for more discussions.

The solution to the problem (4.1)-(4.6) corresponds to the optimal joint rate control and routing. However, to solve this convex program at a single computer, we need to know the global network topology and the solution is a centralized one, which is not practical for large data networks. As observed in [NMR05, ES06, LS04, LSXS15], various versions of backpressure algorithms can be interpreted as distributed solutions to the problem (4.1)-(4.6) from first order Lagrangian methods.

Two mild assumptions are made concerning the problem (4.1)-(4.6).

Assumption 4.1 (Feasibility). *The problem* (4.1)-(4.6) *has at least one optimal solution vector* $[x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F},l\in\mathcal{L}}$.

Assumption 4.2 (Existence of Lagrange Multipliers). *Condition 1.1 holds for the convex program* (4.1)-(4.6). *Specifically, define convex set*

$$\mathcal{C} = \{[x_f; \mu_l^{(f)}]_{f\in\mathcal{F},l\in\mathcal{L}} : (4.3)\text{-}(4.6) \text{ hold}\}.$$

Assume there exists a Lagrange multiplier vector $\boldsymbol{\lambda}^* = [\lambda_n^{(f),*}]_{f\in\mathcal{F},n\in\mathcal{N}\setminus\{Dst(f)\}} \geq \mathbf{0}$ *such that*

$$q(\boldsymbol{\lambda}^*) = \max\{(4.1) : (4.2)\text{-}(4.6)\}$$

where $q(\boldsymbol{\lambda}) = \sup_{[x_f;\mu_l^{(f)}]\in\mathcal{C}} \big\{\sum_{f\in\mathcal{F}} U_f(x_f) - \sum_{f\in\mathcal{F}}\sum_{n\in\mathcal{N}\setminus\{Dst(f)\}} \lambda_n^{(f)}\big[x_f \mathbf{1}_{\{n=Src(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)} - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}\big]\big\}$ *is the Lagrangian dual function of the problem* (4.1)-(4.6) *by treating* (4.3)-(4.6) *as a convex set constraint.*

Assumptions 4.1 and 4.2 hold in most cases of interest. For example, the Slater condition guarantees Assumption 4.2. Since the constraints (4.2)-(4.6) are linear, Proposition 6.4.2 in [BNO03] ensures that Lagrange multipliers exist whenever constraints (4.2)-(4.6) are feasible and when the utility functions U_f are either defined over open sets (such as $U_f(x) = \log(x)$ with $\text{dom}(U_f) = (0,\infty)$) or can be *concavely extended* to open sets, meaning that there is an $\epsilon > 0$ and a concave function $\widetilde{U}_f : (-\epsilon,\infty) \to \mathbb{R}$ such that $\widetilde{U}_f(x) = U_f(x)$ whenever $x \geq 0$.[2]

Fact 4.1 (Replacing Inequality with Equality). *If Assumption 4.1 holds, the problem* (4.1)-(4.6) *has an optimal solution vector* $[x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F},l\in\mathcal{L}}$ *such that all constraints* (4.2) *take equalities.*

Proof. Note that each $\mu_l^{(f)}$ can appear on the left side in at most one constraint (4.2) and appear on the right side in at most one constraint (4.2). Let $[x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F},l\in\mathcal{L}}$ be an optimal solution vector such that at least one inequality constraint (4.2) is loose. Note that we can reduce the value of $\mu_l^{(f),*}$ on the right side of a loose (4.2) until either that constraint holds with equality, or until $\mu_l^{(f),*}$ reduces to 0. The objective function value does not change, and no constraints are violated. We can repeat the process until all inequality constraints (4.2) are tight. □

[2]If $\text{dom}(U_f) = [0,\infty)$, such concave extension is possible if the right-derivative of U_f at $x = 0$ is finite (such as for $U_f(x) = \log(1+x)$ or $U_f(x) = \min[x, 3]$). Such an extension is impossible for the example $U_f(x) = \sqrt{x}$ because the slope is infinite at $x = 0$. Nevertheless, Lagrange multipliers often exist even for these utility functions, such as when the Slater condition holds [BNO03].

4.2 New Backpressure Algorithms

4.2.1 Discussion of Various Queueing Models

At each node, an independent queue backlog is maintained for each session. At each slot t, let $x_f(t)$ be the source session rates; and let $\mu_l^{(f)}(t)$ be the link session rates. Some prior work enforces the constraints (4.2) via virtual queues $Y_n^{(f)}(t)$ of the following form:

$$Y_n^{(f)}(t+1) = \max\big\{Y_n^{(f)}(t) + x_f(t)\mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), 0\big\}. \tag{4.7}$$

While this virtual equation is a meaningful approximation, it differs from reality in that new injected data are allowed to be transmitted immediately, or equivalently, a single packet is allowed to enter and leave many nodes within the same slot. Further, there is no clear connection between the virtual queues $Y_n^{(f)}(t)$ in (4.7) and the actual queues in the network. Indeed, it is easy to construct examples that show there can be an arbitrarily large difference between the $Y_n^{(f)}(t)$ value in (4.7) and the physical queue size in actual networks (see Section 4.6.2 for an illustrating example).

An actual queueing network has queues $Z_n^{(f)}(t)$ with the following dynamics:

$$Z_n^{(f)}(t+1) \leq \max\Big\{Z_n^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), 0\Big\} + x_f(t)\mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t). \tag{4.8}$$

This is faithful to actual queue dynamics and does not allow data to be retransmitted over multiple hops in one slot. Note that (4.8) is an inequality because the new arrivals from other nodes may be strictly less than $\sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t)$ because those other nodes may not have enough backlog to send. The model (4.8) allows for any decisions to be made to fill the transmission values $\mu_l^{(f)}(t)$ in the case that $Z_n^{(f)}(t) \leq \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t)$, provided that (4.8) holds.

This chapter develops new algorithms that converges to the optimal utility defined by the problem (4.1)-(4.6), and that produce worst-case bounded queues on the actual queueing network, that is, with actual queues that evolve as given in (4.8). To begin, it is convenient to introduce

the following virtual queue equation

$$Q_n^{(f)}(t+1) = Q_n^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t) + x_f(t)\mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t), \tag{4.9}$$

where $Q_n^{(f)}(t)$ represents a **virtual queue** value associated with session f at node n. At first glance, this model (4.9) appears to be only an approximation, perhaps even a worse approximation than (4.7), because it allows the $Q_n^{(f)}(t)$ values to be negative. Indeed, we use $Q_n^{(f)}(t)$ only as virtual queues to inform the algorithm and do not treat them as actual queues. However, this paper shows that using these virtual queues to choose the $\boldsymbol{\mu}(t)$ decisions ensures not only that the desired constraints (4.2) are satisfied, but that the resulting $\boldsymbol{\mu}(t)$ decisions create bounded queues $Z_n^{(f)}(t)$ in the **actual network**, where the actual queues evolve according to (4.8). In short, our algorithms can be faithfully implemented with respect to actual queueing networks, and converge to exact optimality on those networks.

The next lemma shows that if an algorithm can guarantee virtual queues $Q_n^{(f)}(t)$ defined in (4.9) are bounded, then actual physical queues satisfying (4.8) are also bounded.

Lemma 4.1. *Consider a network flow problem described by the problem* (4.1)-(4.6). *For all* $l \in \mathcal{L}$ *and* $f \in \mathcal{F}$, *let* $\mu_l^{(f)}(t), x_f(t)$ *be decisions yielded by a dynamic algorithm. Suppose* $Y_n^{(f)}(t)$, $Z_n^{(f)}(t)$, $Q_n^{(f)}(t)$ *evolve by* (4.7)-(4.9) *with initial conditions* $V_n^{(f)}(0) = Z_n^{(f)}(0) = Q_n^{(f)}(0) = 0$. *If there exists a constant* $B > 0$ *such that* $|Q_n^{(f)}(t)| \leq B, \forall t$, *then*

1. $Z_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l$ *for all* $t \in \{0, 1, 2, \ldots\}$.
2. $Y_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l$ *for all* $t \in \{0, 1, 2, \ldots\}$.

Proof.

1. Fix $f \in \mathcal{F}, n \in \mathcal{N} \setminus \{\mathrm{Dst}(f)\}$. Define an auxiliary virtual queue $\widehat{Q}_n^{(f)}(t)$ that is initialized by $\widehat{Q}_n^{(f)}(0) = B + \sum_{l\in\mathcal{O}(n)} C_l$ and evolves according to (4.9). It follows that $\widehat{Q}_n^{(f)}(t) = Q_n^{(f)}(t) + B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$. Since by assumption $Q_n^{(f)}(t) \geq -B, \forall t$, we have $\widehat{Q}_n^{(f)}(t) \geq \sum_{l\in\mathcal{O}(n)} C_l \geq \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), \forall t$. This implies that $\widehat{Q}_n^{(f)}(t)$ also satisfies:

$$\widehat{Q}_n^{(f)}(t+1) = \max\left\{\widehat{Q}_n^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), 0\right\} + x_f(t)\mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t), \forall t, \tag{4.10}$$

which is identical to (4.8) except the inequality is replaced by an equality. Since $Z_n^{(f)}(0) = 0 < \widehat{Q}_n^{(f)}(0)$ and $\widehat{Q}_n^{(f)}(t)$ satisfies (4.10), it follows by inductions that $Z_n^{(f)}(t) \leq \widehat{Q}_n^{(f)}(t), \forall t$. Since $\widehat{Q}_n^{(f)}(t) = Q_n^{(f)}(t) + B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$, and $Q_n^{(f)}(t) \leq B, \forall t$, we have $\widehat{Q}_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$. It follows that $Z_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$.

2. Fix $f \in \mathcal{F}, n \in \mathcal{N} \setminus \{\text{Dst}(f)\}$. By (4.10),

$$\begin{aligned}
&\widehat{Q}_n^{(f)}(t+1)\\
=&\max\big\{\widehat{Q}_n^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), 0\big\} + x_f(t)\mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t)\\
=&\max\big\{\widehat{Q}_n^{(f)}(t) + x_f(t)\mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), \quad x_f(t)\mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t)\big\}\\
\overset{(a)}{\geq}&\max\big\{\widehat{Q}_n^{(f)}(t) + x_f(t)\mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)}(t) - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}(t), \quad 0\big\}
\end{aligned}$$

where (a) follows from the fact that $\mu_l^{(f)}(t), x_f(t), \forall f, l, t$ are non-negative. Note that the right side of the above equation is identical to the right side of (4.7) except that $Y_n^{(f)}(t)$ is rewritten as $\widehat{Q}_n^{(f)}(t)$. Since $Y_n^{(f)}(0) = 0 < \widehat{Q}_n^{(f)}(0)$, by induction, we have $Y_n^{(f)}(t) \leq \widehat{Q}_n^{(f)}(t), \forall t$. Since $\widehat{Q}_n^{(f)}(t) = Q_n^{(f)}(t) + B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$ and $Q_n^{(f)}(t) \leq B, \forall t$, we have $\widehat{Q}_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$. It follows that $Y_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$.

□

4.2.2 New Backpressure Algorithms

In this subsection, we propose two new backpressure algorithms that yield source session rates $x_f(t)$ and link session rates $\mu_l^{(f)}(t)$ at each slot such that the physical queues for each session at each node are bounded by a constant and the time average utility satisfies

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{f\in\mathcal{F}} U_f(x_f(t)) \geq \sum_{f\in\mathcal{F}} U_f(x_f^*) - O(\frac{1}{t}), \forall t,$$

where x_f^* are from the optimal solution to (4.1)-(4.6). Note that Jensen's inequality further implies that

$$\sum_{f\in\mathcal{F}} U_f\big(\frac{1}{t}\sum_{\tau=0}^{t-1} x_f(\tau)\big) \geq \sum_{f\in\mathcal{F}} U_f(x_f^*) - O(\frac{1}{t}), \forall t.$$

The two backpressure algorithm are described in Algorithm 4.1 and Algorithm 4.2, respectively. Similar to existing backpressure algorithms, the updates in both algorithms at each node n are fully distributed and only depend on weights at itself and its neighbor nodes. Unlike existing backpressure algorithms, the weights used to update decision variables $x_f(t)$ and $\mu_l^{(f)}(t)$ are not the virtual queues $Q_n^{(f)}(t)$ themselves, rather, they are augmented values $W_n^{(f)}(t)$ equal to the sum of the virtual queues and the amount of net injected data in the previous slot $t-1$. In addition, the updates involve an additional quadratic term, which is similar to a term used in proximal algorithms [PB13].

The only difference between Algorithm 4.1 and Algorithm 4.2 is that a single global parameter α is used in Algorithm 4.1 while each node n in Algorithm 4.2 owns its own local parameter α_n. In fact, Algorithm 4.1 is derived from the direct application of Algorithm 3.1, developed for general constrained convex programs in Chapter 3, to the problem (4.1)-(4.6) by treating the constraints (4.3)-(4.6) as a convex set constraint and by replacing linear inequality constraints (4.2) with linear equality constraints. Note that by Fact 4.1, to solve the problem (4.1)-(4.6), we can replace linear inequality constraints (4.2) with linear equality constraints without loss of optimality. In Section 3.3.5, it is mentioned that the equation (3.12) can be used as the virtual queue update equation for linear equality constraints in a convex program. Note that the equation (3.12) in the context of the problem (4.1)-(4.6) is identical to (4.9). Since (4.1) and (4.2) are separable, the primal update in Algorithm 3.1 can be decomposed into independent subproblems. Thus, it is easy to observe that Algorithm 4.1 is simply a distributive implementation of Algorithm 3.1 to solve the problem (4.1)-(4.6).

The global parameter α in Algorithm 4.1 is corresponding to the same parameter α in Algorithm 3.1. The results developed in Chapter 3 require $\alpha > \frac{1}{2}\beta^2$, where β is the Lipschitz modulus of the vectorized constraints (4.2). Define $\mathbf{x} = [x_f]_{f\in\mathcal{F}}$ as the stacked column vector of all source session rates and $\boldsymbol{\mu} = [\mu_l^{(f)}]_{f\in\mathcal{F},l\in\mathcal{L}}$ as the stacked column vector of all link session rates. Note that $\mathbf{x}$ has length $|\mathcal{F}|$ and $\boldsymbol{\mu}$ has length $|\mathcal{L}||\mathcal{F}|$. Thus, the constraints (4.2) can be vectorized as

$$\mathbf{g}(\mathbf{x},\boldsymbol{\mu}) = \mathbf{A}\mathbf{x} + \mathbf{R}\boldsymbol{\mu} \leq \mathbf{0}, \tag{4.11}$$

where $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{|\mathcal{F}|} \end{bmatrix}$ is a $|\mathcal{F}|(|\mathcal{N}|-1) \times |\mathcal{F}|$ source-node incidence matrix such that each sub-matrix $\mathbf{A}_f$ is a $\{0,1\}$ matrix of size $(|\mathcal{N}|-1) \times |\mathcal{F}|$ whose (n,f)-th entry is equal to 1 if and only if node n is the source node of session f; and $\mathbf{R} = \text{Diag}\{\mathbf{R}_1, \ldots, \mathbf{R}_{|\mathcal{F}|}\}$ is a block diagonal matrix with $\mathbf{R}_1, \ldots, \mathbf{R}_{|\mathcal{F}|}$ on its diagonal such that each sub-matrix $\mathbf{R}_f$ is a $\{\pm 1, 0\}$ node-arc incidence matrix of size $(|\mathcal{N}|-1) \times |\mathcal{L}|$ whose (n,l)-th entry is equal to 1 if and only if link l flows into node n and is equal to -1 if and only if link l flows out of node n. The Lipschitz modulus of the vectorized version of the constraints (4.2) is summarized in the next Lemma.

Lemma 4.2. *The vector function* $\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}) = \mathbf{A}\mathbf{x} + \mathbf{R}\boldsymbol{\mu}$ *is Lipschitz continuous with modulus*

$$\beta = \sqrt{|\mathcal{F}|} + \sqrt{2|\mathcal{L}|}. \tag{4.12}$$

Proof. Define column vector $\mathbf{y} = [\mathbf{x}; \boldsymbol{\mu}]$ and $\mathbf{B} = [\mathbf{A}, \mathbf{R}]$. The constraints (4.2) can be further rewritten as $\mathbf{g}(\mathbf{y}) = \mathbf{B}\mathbf{y} \leq \mathbf{0}$. Note that linear function $\mathbf{g}(\mathbf{y}) = \mathbf{B}\mathbf{y}$ is Lipschitz continuous with modulus $\|\mathbf{B}\|_2$ where $\|\mathbf{B}\|_2$ is the induced matrix l_2 norm defined as $\|\mathbf{B}\|_2 = \sup_{\mathbf{x} \neq \mathbf{0}} \{\frac{\|\mathbf{B}\mathbf{x}\|}{\|\mathbf{x}\|}\}$. Applying the matrix norm inequalities (for block matrices) $\|[\mathbf{H}_1, \mathbf{H}_2]\|_2 \leq \|\mathbf{H}_1\|_2 + \|\mathbf{H}_2\|_2$ and $\|\text{Diag}\{\mathbf{H}_1, \ldots, \mathbf{H}_K\}\|_2 \leq \max_{1 \leq k \leq K}\{\|\mathbf{H}_k\|_2\}$ yields $\|\mathbf{B}\|_2 \leq \|\mathbf{A}\|_2 + \|\mathbf{R}\|_2 \leq \|\mathbf{A}\|_2 + \max_{f \in \mathcal{F}}\{\|\mathbf{R}_f\|_2\}$. Note that exactly $|\mathcal{F}|$ entries in the matrix $\mathbf{A}$ are 1 and all the other entries are 0; and each matrix $\mathbf{R}_f$ has at most $2|\mathcal{L}|$ non-zero entries whose absolute values are 1. By the fact $\|\mathbf{H}\|_2 \leq \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |H_{ij}|}$ for any matrix $\mathbf{H} \in \mathbb{R}^{m \times n}$, we know $\|\mathbf{A}\|_2 \leq \sqrt{|\mathcal{F}|}$ and $\|\mathbf{R}_f\|_2 \leq \sqrt{2|\mathcal{L}|}, \forall f \in \mathcal{F}$. It follows that $\|\mathbf{B}\|_2 \leq \sqrt{|\mathcal{F}|} + \sqrt{2|\mathcal{L}|}$. □

To determine a large enough value of the parameter α in Algorithm 4.1 to guarantee its convergence, we need to known the number of sessions and the number links in the network. In many applications, these two values may not be globally known at each node. In addition, the value given by $\alpha > \frac{1}{2}\beta^2 = \frac{1}{2}(\sqrt{|\mathcal{F}|} + \sqrt{2|\mathcal{L}|})^2$ can be unnecessarily large for certain network topologies. (That is, the Lipschitz modulus in Lemma 4.2 can be loose in many cases since it is derived without taking the network topology into consideration.) Recall that by the results in Chapter 3, an unnecessarily large value of α in Algorithm 3.1 incurs slow convergence.

To resolve the above issues of Algorithm 4.1, we further develop Algorithm 4.2 by adapting the general Lagrangian methods developed in Chapter 3 for the multi-commodity network flow

problem (4.1)-(4.6). The complete analysis of Algorithm 4.1 is presented in our paper [YN17b]. In the remainder of this chapter, we analyze the performance of Algorithm 4.2; and show that each node n in Algorithm 4.2 determines its own parameter α_n based on local link connections and the value of each α_n is significantly smaller than the global parameter α required in Algorithm 4.1.

Algorithm 4.1 New Backpressure Algorithm with a Global Parameter

Let $\alpha > 0$ be a constant parameter. Initialize $x_f(-1) = 0$, $\mu_l^{(f)}(-1) = 0, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}$ and $Q_n^{(f)}(0) = 0, \forall n \in \mathcal{N}, \forall f \in \mathcal{F}$. At each time $t \in \{0, 1, 2, \ldots\}$, each node n does the following:

- For each $f \in \mathcal{F}$, if node n is not the destination node of session f, i.e., $n \neq \text{Dst}(f)$, then define weight $W_n^{(f)}(t)$:

$$W_n^{(f)}(t) = Q_n^{(f)}(t) + x_f(t-1)\mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l \in \mathcal{I}(n)} \mu_l^{(f)}(t-1) - \sum_{l \in \mathcal{O}(n)} \mu_l^{(f)}(t-1),$$

  If node n is the destination node, i.e., $n = \text{Dst}(f)$, then define $W_n^{(f)}(t) = 0$. Notify neighbor nodes (nodes k that can send session f to node n, i.e., $\forall k$ such that $f \in \mathcal{S}_{(k,n)}$) about this new $W_n^{(f)}(t)$ value.

- For each $f \in \mathcal{F}$, if node n is the source node of session f, i.e., $n = \text{Src}(f)$, choose $x_f(t)$ as the solution to

$$\begin{aligned} \max_{x_f} \quad & U_f(x_f) - W_n^{(f)}(t)x_f - \alpha\big[x_f - x_f(t-1)\big]^2 \\ \text{s.t.} \quad & x_f \in \text{dom}(U_f) \end{aligned}$$

- For all $(n, m) \in \mathcal{O}(n)$, choose $\{\mu_{(n,m)}^{(f)}(t), \forall f \in \mathcal{F}\}$ as the solution to the following convex program:

$$\begin{aligned} \max_{\mu_{(n,m)}^{(f)}} \quad & \sum_{f \in \mathcal{F}} \big[W_n^{(f)}(t) - W_m^{(f)}(t)\big]\mu_{(n,m)}^{(f)} - \alpha \sum_{f \in \mathcal{F}} \big[\mu_{(n,m)}^{(f)} - \mu_{(n,m)}^{(f)}(t-1)\big]^2 \\ \text{s.t.} \quad & \sum_{f \in \mathcal{F}} \mu_{(n,m)}^{(f)} \leq C_{(n,m)} \\ & \mu_{(n,m)}^{(f)} \geq 0, \forall f \in \mathcal{S}_{(n,m)} \\ & \mu_{(n,m)}^{(f)} = 0, \forall f \notin \mathcal{S}_{(n,m)} \end{aligned}$$

- For each $f \in \mathcal{F}$, if node n is not the destination of f, i.e., $n \neq \text{Dst}(f)$, update virtual queue $Q_n^{(f)}(t+1)$ by (4.9).

Algorithm 4.2 New Backpressure Algorithm with Local Parameters

Let $\alpha_n > 0, \forall n \in \mathcal{N}$ be constant parameters. Initialize $x_f(-1) = 0$, $\mu_l^{(f)}(-1) = 0, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}$ and $Q_n^{(f)}(0) = 0, \forall n \in \mathcal{N}, \forall f \in \mathcal{F}$. At each time $t \in \{0, 1, 2, \ldots\}$, each node n does the following:

- For each $f \in \mathcal{F}$, if node n is not the destination node of session f, i.e., $n \neq \text{Dst}(f)$, then define weight $W_n^{(f)}(t)$:

$$W_n^{(f)}(t) = Q_n^{(f)}(t) + x_f(t-1)\mathbf{1}_{\{n=\text{Src}(f)\}} + \sum_{l \in \mathcal{I}(n)} \mu_l^{(f)}(t-1) - \sum_{l \in \mathcal{O}(n)} \mu_l^{(f)}(t-1), \quad (4.13)$$

  If node n is the destination node, i.e., $n = \text{Dst}(f)$, then define $W_n^{(f)}(t) = 0$. Notify neighbor nodes (nodes k that can send session f to node n, i.e., $\forall k$ such that $f \in \mathcal{S}_{(k,n)}$) about this new $W_n^{(f)}(t)$ value.

- For each $f \in \mathcal{F}$, if node n is the source node of session f, i.e., $n = \text{Src}(f)$, choose $x_f(t)$ as the solution to

$$\max_{x_f} \quad U_f(x_f) - W_n^{(f)}(t)x_f - \alpha_n\big[x_f - x_f(t-1)\big]^2 \quad (4.14)$$

$$\text{s.t.} \quad x_f \in \text{dom}(U_f) \quad (4.15)$$

- For all $(n, m) \in \mathcal{O}(n)$, choose $\{\mu_{(n,m)}^{(f)}(t), \forall f \in \mathcal{F}\}$ as the solution to the following convex program:

$$\max_{\mu_{(n,m)}^{(f)}} \quad \sum_{f \in \mathcal{F}} \big[W_n^{(f)}(t) - W_m^{(f)}(t)\big]\mu_{(n,m)}^{(f)} - \big(\alpha_n + \alpha_m\big) \sum_{f \in \mathcal{F}} \big[\mu_{(n,m)}^{(f)} - \mu_{(n,m)}^{(f)}(t-1)\big]^2 \quad (4.16)$$

$$\text{s.t.} \quad \sum_{f \in \mathcal{F}} \mu_{(n,m)}^{(f)} \leq C_{(n,m)} \quad (4.17)$$

$$\mu_{(n,m)}^{(f)} \geq 0, \forall f \in \mathcal{S}_{(n,m)} \quad (4.18)$$

$$\mu_{(n,m)}^{(f)} = 0, \forall f \notin \mathcal{S}_{(n,m)} \quad (4.19)$$

- For each $f \in \mathcal{F}$, if node n is not the destination of f, i.e., $n \neq \text{Dst}(f)$, update virtual queue $Q_n^{(f)}(t+1)$ by (4.9).

4.2.3 Almost Closed-Form Updates in Algorithm 4.2

This subsection shows the decisions $x_f(t)$ and $\mu_l^{(f)}(t)$ in Algorithm 4.2 have either closed-form solutions or "almost" closed-form solutions at each iteration t.

Lemma 4.3. *Let $\hat{x}_f \equiv x_f(t)$ denote the solution to* (4.14)-(4.15).

1. *Suppose $dom(U_f) = [0, \infty)$ and $U_f(x_f)$ is differentiable. Let $h(x_f) = U_f'(x_f) - 2\alpha_n x_f + 2\alpha_n x_f(t-1) - W_n^{(f)}(t)$. If $h(0) < 0$, then $\hat{x}_f = 0$; otherwise $\hat{x}_f$ is the root to the equation $h(x_f) = 0$ and can be found by a bisection search.*

2. *Suppose $dom(U_f) = (0, \infty)$ and $U_f(x_f) = w_f \log(x_f)$ for some weight $w_f > 0$. Then:*

$$\hat{x}_f = \frac{2\alpha_n x_f(t-1) - W_n^{(f)}(t)}{4\alpha_n} + \frac{\sqrt{(W_n^{(f)}(t) - 2\alpha_n x_f(t-1))^2 + 8\alpha_n w_f}}{4\alpha_n}$$

Proof. Trivial. □

The problem (4.16)-(4.19) can be rewritten as follows by eliminating $\mu_{(n,m)}^{(f)}, f \notin \mathcal{S}_{(n,m)}$, completing the squares and replacing maximization with minimization. (Note that $K = |\mathcal{S}_{(n,m)}| \leq |\mathcal{F}|$.)

$$\min \quad \frac{1}{2}\sum_{k=1}^{K}(z_k - a_k)^2 \tag{4.20}$$

$$\text{s.t.} \quad \sum_{k=1}^{K} z_k \leq b \tag{4.21}$$

$$z_k \geq 0, \forall k \in \{1, 2, \ldots, K\} \tag{4.22}$$

Lemma 4.4. *The solution to the problem* (4.20)-(4.22) *is given by $z_k^* = \max\{0, a_k - \theta^*\}, \forall k \in \{1, 2, \ldots, K\}$ where $\theta^* \geq 0$ can be found either by a bisection search (See Section 4.6.3) or by Algorithm 4.3 with complexity $O(K \log K)$.*

Proof. A similar problem where (4.21) is replaced with an equality constraint in considered in [DSSSC08]. The optimal solution to this quadratic program is characterized by its KKT condition and a corresponding algorithm can be developed to obtain its KKT point. A complete proof is presented in Section 4.6.3. □

Algorithm 4.3 Algorithm to Solve Problem (4.20)-(4.22)

1. Check if $\sum_{k=1}^{K} \max\{0, a_k\} \leq b$ holds. If yes, let $\theta^* = 0$ and $z_k^* = \max\{0, a_k\}, \forall k \in \{1, 2, \ldots, K\}$ and terminate the algorithm; else, continue to the next step.
2. Sort all $a_k, \in \{1, 2, \ldots, K\}$ in a decreasing order π such that $a_{\pi(1)} \geq a_{\pi(2)} \geq \cdots \geq a_{\pi(K)}$. Define $S_0 = 0$.
3. For $k = 1$ to K
   - Let $S_k = S_{k-1} + a_k$. Let $\theta^* = \frac{S_k - b}{k}$.
   - If $\theta^* \geq 0$, $a_{\pi(k)} - \theta^* > 0$ and $a_{\pi(k+1)} - \theta^* \leq 0$, then terminate the loop; else, continue to the next iteration in the loop.
4. Let $z_k^* = \max\{0, a_k - \theta^*\}, \forall k \in \{1, 2, \ldots, K\}$ and terminate the algorithm.

Note that the step (3) in Algorithm 4.3 has complexity $O(K)$ and hence the overall complexity of Algorithm 4.3 is dominated by the sorting step (2) with complexity $O(K \log(K))$.

4.3 Performance Analysis of Algorithm 4.2

In this section, we show that the new backpressure algorithm has vanishing utility optimality gaps that decay like $O(1/t)$, where t is number of iterations, and finite queue lengths.

4.3.1 Preliminaries

Let $\mathbf{y} = [x_f; \mu_l^{(f)}]_{f \in \mathcal{F}, l \in \mathcal{L}}$ define a column vector. For each $f \in \mathcal{F}, n \in \mathcal{N} \setminus \{\text{Dst}(f)\}$, define

$$\mathbf{y}_n^{(f)} = \begin{cases} [x_f; \mu_l^{(f)}]_{l \in \mathcal{I}(n) \cup \mathcal{O}(n)} & \text{if } n = \text{Src}(f), \\ [\mu_l^{(f)}]_{l \in \mathcal{I}(n) \cup \mathcal{O}(n)} & \text{else}, \end{cases} \tag{4.23}$$

which is a column vector composed by the control actions appearing in each constraint (4.2); and introduce a function with respect to $\mathbf{y}_n^{(f)}$ as

$$g_n^{(f)}(\mathbf{y}_n^{(f)}) = x_f \mathbf{1}_{\{n = \text{Src}(f)\}} + \sum_{l \in \mathcal{I}(n)} \mu_l^{(f)} - \sum_{l \in \mathcal{O}(n)} \mu_l^{(f)} \tag{4.24}$$

Thus, the constraints (4.2) can be rewritten as

$$g_n^{(f)}(\mathbf{y}_n^{(f)}) \leq 0, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus \{\text{Dst}(f)\}.$$

Note that each vector $\mathbf{y}_n^{(f)}$ is a subvector of $\mathbf{y}$ and has length $d_n + 1$ where d_n is the degree of node n (the total number of outgoing links and incoming links) if node n is the source of session f; and has length d_n if node n is not the source of session f. Note that components in different vector variables $\mathbf{y}_n^{(f)}$ can overlap. The vector variables $\mathbf{y}$ and $\mathbf{y}_n^{(f)}$ are introduced only to simplify notation.

Fact 4.2. *Each function $g_n^{(f)}(\cdot)$ defined in* (4.24) *is Lipschitz continuous with respect to vector $\mathbf{y}_n^{(f)}$ with modulus*

$$\beta_n \leq \sqrt{d_n + 1}.$$

where d_n is the degree of node n.

Proof. This fact can be easily shown by noting that each $g_n^{(f)}(\mathbf{y}_n^{(f)})$ is a linear function with respect to vector $\mathbf{y}_n^{(f)}$ and has at most $d_n + 1$ non-zero coefficients that are equal to ± 1. □

Note that virtual queue update equation (4.9) can be rewritten as:

$$Q_n^{(f)}(t+1) = Q_n^{(f)}(t) + g_n^{(f)}(\mathbf{y}_n^{(f)}(t)), \tag{4.25}$$

and weight update equation (4.13) can be rewritten as:

$$W_n^{(f)}(t) = Q_n^{(f)}(t) + g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1)). \tag{4.26}$$

Define

$$L(t) = \frac{1}{2} \sum_{f \in \mathcal{F}} \sum_{n \in \mathcal{N} \setminus \mathrm{Dst}(f)} [Q_n^{(f)}(t)]^2 \tag{4.27}$$

and call it a *Lyapunov function*. In the remainder of this chapter, double summations are often compactly written as a single summation, e.g.,

$$\sum_{f \in \mathcal{F}} \sum_{n \in \mathcal{N} \setminus \mathrm{Dst}(f)} \big[\, \cdot \,\big] \triangleq \sum_{\substack{f \in \mathcal{F}, \\ n \in \mathcal{N} \setminus \mathrm{Dst}(f)}} \big[\, \cdot \,\big].$$

Define the *Lyapunov drift* as

$$\Delta(t) = L(t+1) - L(t).$$

The following lemma follows directly from equation (4.25).

Lemma 4.5. *At each iteration $t \in \{0, 1, \ldots\}$ in Algorithm 4.2, the Lyapunov drift is given by*

$$\Delta(t) = \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus Dst(f)}} \big[Q_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^f(t)) + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^f(t))]^2\big]. \tag{4.28}$$

Proof. Fix $f \in \mathcal{F}$ and $n \in \mathcal{N} \setminus \mathrm{Dst}(f)$, we have

$$\begin{aligned}\frac{1}{2}[Q_n^{(f)}(t+1)]^2 - \frac{1}{2}[Q_n^{(f)}(t)]^2 &\overset{(a)}{=} \frac{1}{2}[Q_n^{(f)}(t) + g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2 - \frac{1}{2}[Q_n^{(f)}(t)]^2 \\ &= Q_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^f(t)) + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^f(t))]^2 \end{aligned} \tag{4.29}$$

where (a) follows from (4.25).

By the definition of $\Delta(t)$, we have

$$\begin{aligned}\Delta(t) &= \frac{1}{2} \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \big[Q_n^{(f)}(t+1)\big]^2 - [Q_n^{(f)}(t)]^2\big] \\ &\overset{(a)}{=} \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \big[Q_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^f(t)) + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^f(t))]^2\big]\end{aligned}$$

where (a) follows from (4.29). □

Define $f(\mathbf{y}) = \sum_{f\in\mathcal{F}} U_f(x_f)$. At each time t, consider choosing a decision vector $\mathbf{y}(t)$ that includes elements in each subvector $\mathbf{y}_n^{(f)}(t)$ to solve the following problem:

$$\begin{aligned}\max_{\mathbf{y}}\ & f(\mathbf{y}) - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \big[W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f)}) + \alpha_n \|\mathbf{y}_n^{(f)} - \mathbf{y}_n^{(f)}(t-1)\|^2\big] \\ & - \sum_{\substack{f\in\mathcal{F},\\ n=\mathrm{Dst}(f)}} \alpha_n \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2 \qquad (4.30) \\ \text{s.t.}\ & (4.3)\text{-}(4.6) \qquad (4.31)\end{aligned}$$

The expression (4.30) is called a *modified drift-plus-penalty expression*. This results in the novel backpressure-type algorithm of Algorithm 4.2. Indeed, the decisions in Algorithm 4.2 were *derived* as the solution to the above problem (4.30)-(4.31). This is formalized in the next lemma.

Lemma 4.6. *At each iteration $t \in \{0, 1, \ldots\}$, the action $\mathbf{y}(t)$ jointly chosen in Algorithm 4.2 is the solution to the problem* (4.30)-(4.31).

Proof. The proof involves collecting terms associated with the $x_f(t)$ and $\mu_l^{(f)}(t)$ decisions. See Section 4.6.4 for details. □

Furthermore, the next lemma relates $h(\mathbf{y}^*)$ and $h(\mathbf{y}(t))$ yielded by action $\mathbf{y}(t)$ that aggregates all control actions jointly chosen in Algorithm 4.2 at each iteration $t \in \{0, 1, \ldots\}$.

Lemma 4.7. *Let $\mathbf{y}^* = [x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F}, l\in\mathcal{L}}$ be an optimal solution to problem* (4.1)-(4.6) *given in Fact 4.1, i.e., $g_n^{(f)}(\mathbf{y}_n^{(f),*}) = 0, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus Dst(f)$. If $\alpha_n \geq \frac{1}{2}(d_n + 1), \forall n \in \mathcal{N}$, where d_n is the degree of node n, then the action $\mathbf{y}(t) = [x_f(t); \mu_l^{(f)}(t)]_{f\in\mathcal{F}, l\in\mathcal{L}}$ jointly chosen in Algorithm 4.2 at each iteration $t \in \{0, 1, \ldots\}$ satisfies*

$$f(\mathbf{y}(t)) \geq f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \Delta(t),$$

where $\Phi(t) = \sum_{f\in\mathcal{F}, n\in\mathcal{N}} \left[\alpha_n \mathbf{1}_{\{n\neq Dst(f)\}} \|\mathbf{y}_n^{(f),*} - \mathbf{y}_n^{(f)}(t)\|^2 + \alpha_n \mathbf{1}_{\{n=Dst(f)\}} \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f),*} - \mu_l^{(f)}(t)]^2\right]$.

Proof. See Section 4.6.5. □

It remains to show that this modified backpressure algorithm leads to fundamentally improved performance.

4.3.2 Utility Optimality Gap Analysis

Define $\mathbf{Q}(t) = \left[Q_n^{(f)}(t)\right]_{f\in\mathcal{F}, n\in\mathcal{N}\setminus\{\mathrm{Dst}(f)\}}$ as the stacked column vector of all virtual queues $Q_n^{(f)}(t)$ defined in (4.9). Note that (4.27) can be rewritten as $L(t) = \frac{1}{2}\|\mathbf{Q}(t)\|^2$. Define vectorized constraints (4.2) as $\mathbf{g}(\mathbf{y}) = [g_n^{(f)}(\mathbf{y}_n^{(f)})]_{f\in\mathcal{F}, n\in\mathcal{N}\setminus\mathrm{Dst}(f)}$.

Lemma 4.8. *Let $\mathbf{y}^* = [x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F}, l\in\mathcal{L}}$ be an optimal solution to the problem* (4.1)-(4.6) *given in Fact 4.1, i.e., $g_n^{(f)}(\mathbf{y}_n^{(f),*}) = 0, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus Dst(f)$. If $\alpha_n \geq \frac{1}{2}(d_n + 1), \forall n \in \mathcal{N}$ in*

Algorithm 4.2, where d_n is the degree of node n, then for all $t \geq 1$,

$$\sum_{\tau=0}^{t-1} f(\mathbf{y}(\tau)) \geq tf(\mathbf{y}^*) - \zeta + \frac{1}{2}\|\mathbf{Q}(t)\|^2.$$

where $\zeta = \Phi(-1) = \sum_{f\in\mathcal{F},n\in\mathcal{N}} \left[\alpha_n \mathbf{1}_{\{n\neq Dst(f)\}}\|\mathbf{y}_n^{(f),*}\|^2 + \alpha_n \mathbf{1}_{\{n=Dst(f)\}} \sum_{l\in\mathcal{I}(n)} (\mu_l^{(f),*})^2\right]$ *is a constant.*

Proof. By Lemma 4.7, we have $f(\mathbf{y}(\tau)) \geq f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \Delta(\tau), \forall \tau \in \{0, 1, \ldots, t-1\}$. Summing over $\tau \in \{0, 1, \ldots, t-1\}$ yields

$$\begin{aligned}
\sum_{\tau=0}^{t-1} f(\mathbf{y}(\tau)) \geq & tf(\mathbf{y}^*) + \sum_{\tau=0}^{t-1}[\Phi(\tau) - \Phi(\tau-1)] + \sum_{\tau=0}^{t-1}\Delta(\tau) \\
= & tf(\mathbf{y}^*) + \Phi(t) - \Phi(-1) + \sum_{\tau=0}^{t-1}\Delta(\tau) \\
\overset{(a)}{\geq} & tf(\mathbf{y}^*) - \Phi(-1) + \sum_{\tau=0}^{t-1}\Delta(\tau)
\end{aligned}$$

where (a) follows from the fact that $\Phi(t) \geq 0, \forall t$.

Recall $\Delta(\tau) = L(\tau+1) - L(\tau)$, simplifying summations and rearranging terms yields

$$\begin{aligned}
\sum_{\tau=0}^{t-1} f(\mathbf{y}(\tau)) \geq & tf(\mathbf{y}^*) - \Phi(-1) + L(t) - L(0) \\
\overset{(a)}{=} & tf(\mathbf{y}^*) - \Phi(-1) + \frac{1}{2}\|\mathbf{Q}(t)\|^2
\end{aligned}$$

where (a) follows from the fact that $L(0) = \mathbf{0}$ and $L(t) = \frac{1}{2}\|\mathbf{Q}(t)\|^2$. □

The next theorem summarizes that Algorithm 4.2 yields a vanishing utility optimality gap that approaches zero like $O(\frac{1}{t})$.

Theorem 4.1. *Let* $\mathbf{y}^* = [x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F},l\in\mathcal{L}}$ *be an optimal solution to the problem* (4.1)-(4.6) *given in Fact 4.1, i.e.,* $g_n^{(f)}(\mathbf{y}_n^{(f),*}) = 0, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus Dst(f)$. *If* $\alpha_n \geq \frac{1}{2}(d_n + 1), \forall n \in \mathcal{N}$ *in Algorithm 4.2, where d_n is the degree of node n, then for all $t \geq 1$, we have*

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{f\in\mathcal{F}} U_f(x_f(\tau)) \geq \sum_{f\in\mathcal{F}} U_f(x_f^*) - \frac{1}{t}\zeta,$$

where ζ is a constant defined in Lemma 4.8. Moreover, if we define $\overline{x}_f(t) = \frac{1}{t}\sum_{\tau=0}^{t-1} x_f(\tau), \forall f \in$

$\mathcal{F}$, *then*

$$\sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) \geq \sum_{f\in\mathcal{F}} U_f(x_f^*) - \frac{1}{t}\zeta.$$

Proof. Recall that $f(\mathbf{y}) = \sum_{f\in\mathcal{F}} U_f(x_f)$. By Lemma 4.8, we have

$$\begin{aligned}\sum_{\tau=0}^{t-1}\sum_{f\in\mathcal{F}} U_f(x_f(\tau)) \geq & t\sum_{f\in\mathcal{F}} U_f(x_f^*) - \zeta + \frac{1}{2}\|\mathbf{Q}(t)\|^2 \\ \overset{(a)}{\geq} & t\sum_{f\in\mathcal{F}} U_f(x_f^*) - \zeta.\end{aligned}$$

where (a) follows from the trivial fact that $\|\mathbf{Q}(t)\|^2 \geq 0$.

Dividing both sides by a factor t yields the first inequality in this theorem. The second inequality follows from the concavity of $U_f(\cdot)$ and Jensen's inequality. □

4.3.3 Queue Length Analysis

Lemma 4.9. *Let* $\mathbf{Q}(t), t \in \{0, 1, \ldots\}$ *be the virtual queue vectors in Algorithm 4.2. For any* $t \geq 1$,

$$\mathbf{Q}(t) = \sum_{\tau=0}^{t-1} \mathbf{g}(\mathbf{y}(\tau))$$

Proof. This lemma follows directly from the fact that $\mathbf{Q}(0) = \mathbf{0}$ and queue update equation (4.9) can be written as $\mathbf{Q}(t+1) = \mathbf{Q}(t) + \mathbf{g}(\mathbf{y}(t))$. □

The next theorem shows the boundedness of all virtual queues $Q_n^{(f)}(t)$ in Algorithm 4.2.

Theorem 4.2. *Let* $\mathbf{y}^* = [x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F}, l\in\mathcal{L}}$ *be an optimal solution to the problem* (4.1)-(4.6) *given in Fact 4.1, i.e.,* $g_n^{(f)}(\mathbf{y}_n^{(f),*}) = 0, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus Dst(f)$, *and* $\boldsymbol{\lambda}^*$ *be a Lagrange multiplier vector given in Assumption 4.2. If* $\alpha_n \geq \frac{1}{2}(d_n+1)^2, \forall n \in \mathcal{N}$ *in Algorithm 4.2, where* d_n *is the degree of node* n, *then for all* $t \geq 1$,

$$|Q_n^{(f)}(t)| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\zeta}, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus \{Dst(f)\}.$$

where ζ *is a constant defined in Lemma 4.8.*

Proof. Let $q(\boldsymbol{\lambda}) = \sup_{\mathbf{y}\in\mathcal{C}} \left\{ f(\mathbf{y}) - \boldsymbol{\lambda}^{\mathsf{T}}\mathbf{g}(\mathbf{y}) \right\}$ be the Lagrangian dual function defined in Assumption 4.2. For all $\tau \in \{0, 1, \dots, \}$, by Assumption 4.2, we have

$$f(\mathbf{y}^*) = q(\boldsymbol{\lambda}^*) \overset{(a)}{\geq} f(\mathbf{y}(\tau)) - \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y}(\tau))$$

where (a) follows from the definition of $q(\boldsymbol{\lambda}^*)$. Rearranging terms yields

$$f(\mathbf{y}(\tau)) \leq f(\mathbf{y}^*) + \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y}(\tau)), \forall \tau \in \{0, 1, \dots\}.$$

Fix $t > 0$. Summing over $\tau \in \{0, 1, \dots, t-1\}$ yields

$$\begin{aligned}
\sum_{\tau=0}^{t-1} f(\mathbf{y}(\tau)) \leq & t f(\mathbf{y}^*) + \sum_{\tau=0}^{t-1} \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y}(\tau)) \\
= & t f(\mathbf{y}^*) + \boldsymbol{\lambda}^{*,\mathsf{T}} \sum_{\tau=0}^{t-1} \mathbf{g}(\mathbf{y}(\tau)) \\
\overset{(a)}{=} & t f(\mathbf{y}^*) + \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{Q}(t) \\
\overset{(b)}{\leq} & t f(\mathbf{y}^*) + \|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t)\|
\end{aligned}$$

where (a) follows form Lemma 4.9 and (b) follows from Cauchy-Schwarz inequality.

On the other hand, by Lemma 4.8, we have

$$\sum_{\tau=0}^{t-1} f(\mathbf{y}(\tau)) \geq t f(\mathbf{y}^*) - \zeta + \frac{1}{2}\|\mathbf{Q}(t)\|^2.$$

Combining the last two inequalities and cancelling the common terms yields

$$\begin{aligned}
\frac{1}{2}\|\mathbf{Q}(t)\|^2 - \zeta \leq \|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t)\| \Rightarrow & \left(\|\mathbf{Q}(t)\| - \|\boldsymbol{\lambda}^*\|\right)^2 \leq \|\boldsymbol{\lambda}^*\|^2 + 2\zeta \\
\Rightarrow & \|\mathbf{Q}(t)\| \leq \|\boldsymbol{\lambda}^*\| + \sqrt{\|\boldsymbol{\lambda}^*\|^2 + 2\zeta} \\
\overset{(a)}{\Rightarrow} & \|\mathbf{Q}(t)\| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\zeta}
\end{aligned}$$

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

Thus, for any $f \in \mathcal{F}$ and $n \in \mathcal{N} \setminus \{\mathrm{Dst}(f)\}$, we have

$$|Q_n^{(f)}(t)| \leq \|\mathbf{Q}(t)\| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\zeta}.$$

□

This theorem shows that the absolute values of all virtual queues $Q_n^{(f)}(t)$ are bounded by a constant $B = 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\zeta}$ from above. By Lemma 4.1 and discussions in Section 4.2.1, the actual physical queues $Z_n^{(f)}(t)$ evolving via (4.8) satisfy $Z_n^{(f)}(t) \leq 2B + \sum_{l\in\mathcal{O}(n)} C_l, \forall t$. This is summarized in the next corollary.

Corollary 4.1. *Let $\mathbf{y}^* = [x_f^*; \mu_l^{(f),*}]_{f\in\mathcal{F},l\in\mathcal{L}}$ be an optimal solution to the problem (4.1)-(4.6) given in Fact 4.1, i.e., $g_n^{(f)}(\mathbf{y}_n^{(f),*}) = 0, \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus Dst(f)$, and $\boldsymbol{\lambda}^*$ be a Lagrange multiplier vector given in Assumption 4.2. If $\alpha_n \geq \frac{1}{2}(d_n+1)^2, \forall n \in \mathcal{N}$ in Algorithm 4.2, where d_n is the degree of node n, then all actual physical queues $Z_n^{(f)}(t), \forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus \{Dst(f)\}$ in the network evolving via (4.8) satisfy*

$$Z_n^{(f)}(t) \leq 4\|\boldsymbol{\lambda}^*\| + 2\sqrt{2\zeta} + \sum_{l\in\mathcal{O}(n)} C_l, \quad \forall t.$$

where ζ is a constant defined in Lemma 4.8.

Define vector $\mathbf{x}^* = [x_f^*]_{f\in\mathcal{F}}$ and $\overline{\mathbf{x}}(t) = [\overline{x}_f(t)]_{f\in\mathcal{F}}$ where x_f^* and $\overline{x}_f(t)$ are defined in Theorem 4.1. Note that if each $U_f(\mathbf{x}_f)$ is strongly concave with respect to x_f, then $\mathbf{x}^*$ is unique by strong concavity. (However, $[\mu_l^{(f),*}]_{l\in\mathcal{L}}$ is not necessarily unique.) In this case, Corollary 4.2 shows $\overline{\mathbf{x}}(t)$ yielded by Algorithm 4.2 converges to the unique maximizer $\mathbf{x}^*$.

Corollary 4.2. *If the conditions in Theorem 4.1 hold and each $U_f(x_f)$ is strongly concave with respect to x_f, then Algorithm 4.2 guarantees $\overline{\mathbf{x}}(t) \to \mathbf{x}^*$ as $t \to \infty$.*

Proof. Assume each $U_f(x_f)$ is strongly concave with respect to x_f with modulus c_f. Let $c = \min_{f\in\mathcal{F}}\{c_f\}$. Note that $\mathcal{C} = \{[x_f; \mu_l^{(f)}]_{f\in\mathcal{F},l\in\mathcal{L}} : (4.3)\text{-}(4.6) \text{ hold}\}$ is a compact set. By Assumption 4.2, we have $\mathbf{y}^* = \text{argmax}_{\mathbf{y}\in\mathcal{C}}\{h(\mathbf{y}) - \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y})\}$. Recall that $h(\mathbf{y}) = \sum_{f\in\mathcal{F}} U_f(x_f)$ and $\mathbf{g}(\mathbf{y})$ are separable since they can be written as the sum of scalar functions in terms of x_f and $\mu_l^{(f)}$. Thus, x_f^* and $[\mu_l^{(f),*}]_{f\in\mathcal{F}}$ appear separably and maximize $h(\mathbf{y}) - \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y})$ to obtain the left-side of (4.32) where each x_f^* satisfying (4.6) maximizes a strongly concave part and each vector $[\mu_l^{(f),*}]_{f\in\mathcal{F}}$ satisfying (4.3)-(4.5) maximizes a concave part. Define $\overline{\mathbf{y}}(t) = \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbf{y}(\tau)$. Note that $\overline{\mathbf{y}}(t)$ satisfies (4.3)-(4.6) since each $\mathbf{y}(\tau)$ is generated by Algorithm 4.2. By Corollary

1.2, for all $t \geq 1$,

$$\begin{aligned}
&\sum_{f\in\mathcal{F}} U_f(x_f^*) - \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y}^*) \\
&\geq \sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) - \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\overline{\mathbf{y}}(t)) + \sum_{f\in\mathcal{F}} \frac{c_f}{2}(x_f^* - \overline{x}_f(t))^2 \\
&\overset{(a)}{\geq} \sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) - \boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\overline{\mathbf{y}}(t)) + \frac{c}{2}\|\overline{\mathbf{x}}(t) - \mathbf{x}^*\|^2 \\
&\overset{(b)}{=} \sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) - \boldsymbol{\lambda}^{*,\mathsf{T}}\frac{1}{t}\sum_{t=0}^{t-1}\mathbf{g}(\mathbf{y}[\tau]) + \frac{c}{2}\|\overline{\mathbf{x}}(t) - \mathbf{x}^*\|^2 \\
&\overset{(c)}{=} \sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) - \frac{1}{t}\boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{Q}(t) + \frac{c}{2}\|\overline{\mathbf{x}}(t) - \mathbf{x}^*\|^2
\end{aligned} \tag{4.32}$$

where (a) follows from $c = \min_{f\in\mathcal{F}}\{c_f\}$; (b) follows from the linearity of $\mathbf{g}(\cdot)$ and the definition of $\overline{\mathbf{y}}(t)$; and (c) follows from Lemma 4.9.

Recall that $\boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{g}(\mathbf{y}^*) = 0$ by strong duality of convex programs (Assumption 4.2). Thus, (4.32) implies

$$\begin{aligned}
&\frac{c}{2}\|\overline{\mathbf{x}}(t) - \mathbf{x}^*\|^2 \\
&\leq \sum_{f\in\mathcal{F}} U_f(x_f^*) - \sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) + \frac{1}{t}\boldsymbol{\lambda}^{*,\mathsf{T}}\mathbf{Q}(t) \\
&\overset{(a)}{\leq} \sum_{f\in\mathcal{F}} U_f(x_f^*) - \sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) + \frac{1}{t}\|\boldsymbol{\lambda}^*\|\|\mathbf{Q}(t)\| \\
&\overset{(b)}{\leq} \frac{1}{t}\zeta + \frac{1}{t}\|\boldsymbol{\lambda}^*\|(2\|\boldsymbol{\lambda}^*\| + \sqrt{2\zeta})
\end{aligned}$$

where (a) follows from the Cauchy-Schwarz inequality; and (b) follows from Theorem 4.1, which implies $\sum_{f\in\mathcal{F}} U_f(\overline{x}_f(t)) \geq \sum_{f\in\mathcal{F}} U_f(x_f^*) - \frac{1}{t}\zeta, \forall t \geq 1$, and Theorem 4.2, which implies $\|\mathbf{Q}(t)\| \leq 2\|\boldsymbol{\lambda}^*\| + \sqrt{2\zeta}, \forall t \geq 1$.

Taking limits $t \to \infty$ on both sides yields that $\overline{\mathbf{x}}(t) \to \mathbf{x}^*$ as $t \to \infty$. □

4.3.4 Performance of Algorithm 4.2

Theorems 4.1 and Corollary 4.1 together imply that Algorithm 4.2 with $\alpha_n \geq \frac{1}{2}(d_n + 1), \forall n \in \mathcal{N}$ can achieve a vanishing utility optimality gap that decays like $O(\frac{1}{t})$, where t is number of iterations, and guarantees the physical queues at each node for each session are always bounded

by a constant that is independent of the utility optimality gap.

This is superior to existing backpressure algorithms from [ES06, Nee10, LSXS15] that can achieve an $O(\frac{1}{V})$ utility gap only at the cost of an $O(V^2)$ or $O(V)$ queue length, where V is an algorithm parameter. To obtain a vanishing utility gap, existing backpressure algorithms in [ES06, Nee10, LSXS15] necessarily yield unbounded queues. To obtain a vanishing utility gap, existing backpressure algorithms in [ES06, Nee10] yield unbounded queues. We also comment that $O(V^2)$ queue bound in the primal-dual type backpressure algorithm [ES06] is actually of the order $V^2\|\boldsymbol{\lambda}^*\| + B_1$ where $\boldsymbol{\lambda}^*$ is the Lagrangian multiplier vector attaining strong duality and B_1 is a constant determined by the problem parameters. A recent work [Nee14] also shows that the $O(V)$ queue bound in the backpressure algorithm from drift-plus-penalty is of the order $V\|\boldsymbol{\lambda}^*\| + B_2$ where B_2 is also a constant determined by the problem parameters. Since $\boldsymbol{\lambda}^*$ is a constant vector independent of V, both algorithms are claimed to have $O(V^2)$ or $O(V)$ queue bounds. By Corollary 4.1, Algorithm 4.2 guarantees physical queues at each node are bounded by $4\|\boldsymbol{\lambda}^*\| + B_3$, where B_3 is constant given a problem. Thus, the constant queue bound guaranteed by Algorithm 4.2 is typically smaller than the $O(V^2)$ or $O(V)$ queue bounds from [ES06] and [Nee14] even for a small V. (A small V can yield a poor utility performance in the backpressure algorithms in [ES06, Nee10].)

Theorems 4.1 and Corollary 4.1 require $\alpha_n \geq \frac{1}{2}(d_n + 1), \forall n \in \mathcal{N}$ in Algorithm 4.2. The required value of each α_n is significantly smaller than $\alpha > \frac{1}{2}\left(\sqrt{|\mathcal{F}|} + \sqrt{2|\mathcal{L}|}\right)^2$ required by Algorithm 4.1 according to Lemma 4.2 and the general theory developed in Chapter 3.

4.4 Numerical Experiment

In this section, we consider a simple network with 6 nodes and 8 links and 2 sessions as described in Figure 4.1. This network has two sessions: session 1 from node 1 to node 6 has utility function $\log(x_1)$ and session 2 from node 3 to node 4 has utility function $1.5\log(x_2)$. The log utilities are widely used as metrics of proportional fairness in the network [KMT98]. The routing path of each session is arbitrary as long as data can be delivered from the source node to the destination node. For simplicity, assume that each link has capacity 1. The optimal source session rate to problem (4.1)-(4.6) is $x_1^* = 1.2$ and $x_2^* = 1.8$ and link session rates, i.e., static routing for each session, is drawn in Figure 4.2.

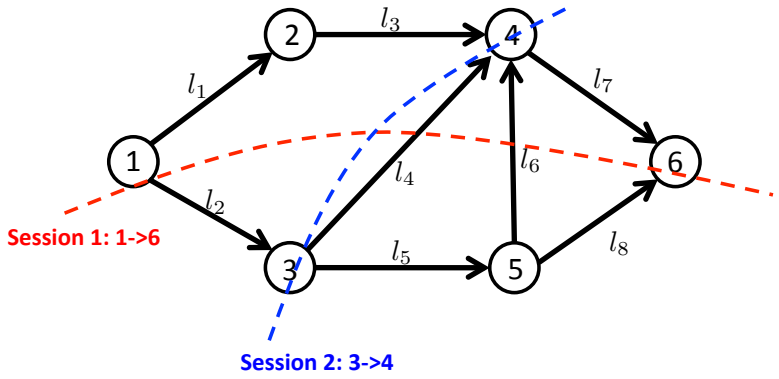

Figure 4.1: A simple network with 6 nodes, 8 links and 2 sessions.

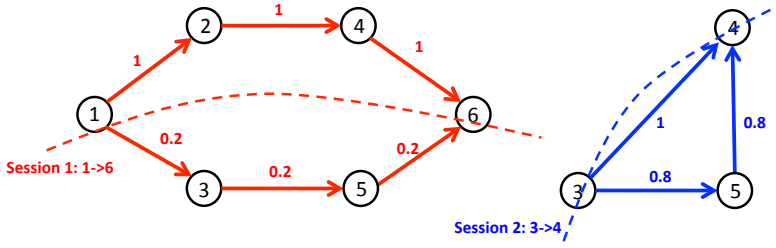

Figure 4.2: The optimal routing for the network in Figure 4.1.

To compare the convergence performance of Algorithm 4.2 and the backpressure algorithm in [Nee10] (with the best utility-delay tradeoff among all existing backpressure algorithms), we run both Algorithm 4.2 with $\alpha_n = \frac{1}{2}(d_n+1), \forall n \in \mathcal{N}$ and the backpressure algorithm in [Nee10] with $V = 500$ to plot Figure 4.3. It can be observed from Figure 4.3 that Algorithm 4.2 converges to the optimal source session rates faster than the backpressure algorithm in [Nee10]. The backpressure algorithm in [Nee10] with $V = 400$ takes around 2500 iterations to converges to source rates close to $(1.2, 1.8)$ while Algorithm 4.2 only takes around 800 iterations to converges to $(1.2, 1.8)$ (as shown in the zoom-in subfigure at the top right corner.) In fact, the backpressure algorithm in [Nee10] with $V = 500$ can not converge to the exact optimal source session rate $(1.2, 1.8)$ but can only converge to its neighborhood with a distance gap determined by the value of V. This is an effect from the fundamental $[O(\frac{1}{V}), O(V)]$ utility-delay tradeoff of the the backpressure algorithm in [Nee10]. In contrast, Algorithm 4.2 can eventually converge to the the exact optimal source session rate $(1.2, 1.8)$. A zoom-in subfigure at the bottom right corner in Figure 4.2 verifies this and shows that the source rate for Session 1 in Algorithm 4.2 converges to 1.2 while the source rate in the backpressure algorithm in [Nee10] with $V = 500$ oscillates around a point slightly

larger than 1.2.

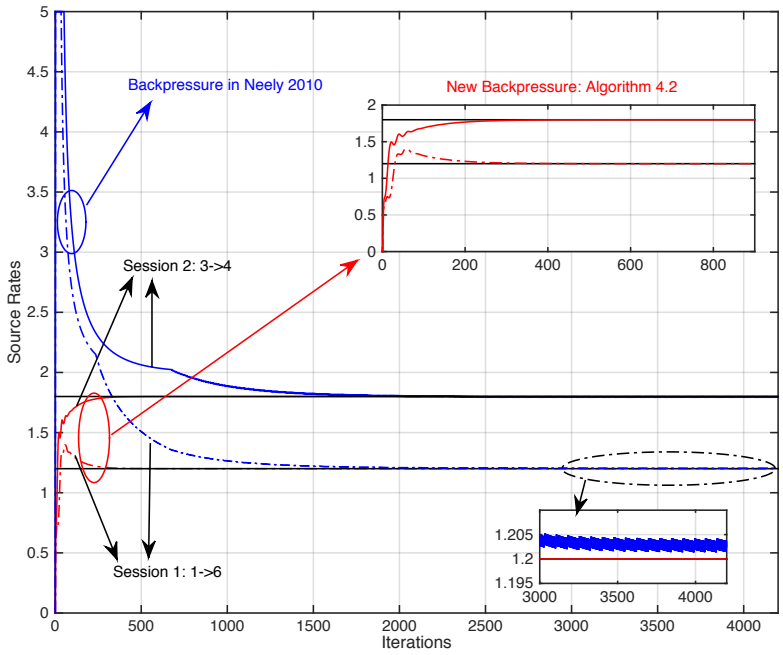

Figure 4.3: Convergence performance comparison between Algorithm 4.2 and the backpressure algorithm in [Nee10].

Corollary 4.1 shows that Algorithm 4.2 guarantees each actual queue in the network is bounded by constant $4\|\boldsymbol{\lambda}^*\| + 2\sqrt{2\xi}\|\mathbf{y}^*\| + \sum_{l\in\mathcal{O}(n)} C_l$. Recall that the backpressure algorithm in [Nee10] can guarantee the actual queues in the network are bounded by a constant of order $V\|\boldsymbol{\lambda}^*\|$. Figure 4.4 plots the sum of **actual queue length** at each node for Algorithm 4.2 and the backpressure algorithm in [Nee10] with $V = 10, 100$ and 500. (Recall a larger V in the backpressure algorithm in [Nee10] yields a smaller utility gap but a larger queue length.) It can be observed that Algorithm 4.2 has the smallest **actual queue length** (see the zoom-in subfigure) and the actual queue length of the backpressure algorithm in [Nee10] scales linearly with respect to V.

4.5 Chapter Summary

This chapter develops new backpressure algorithms for joint rate control and routing in multi-hop data networks. The new backpressure algorithms can achieve vanishing utility optimality gaps and finite queue lengths. This improves the state-of-art $[O(\epsilon), O(1/\epsilon^2)]$ or $[O(\epsilon), O(1/\epsilon)]$

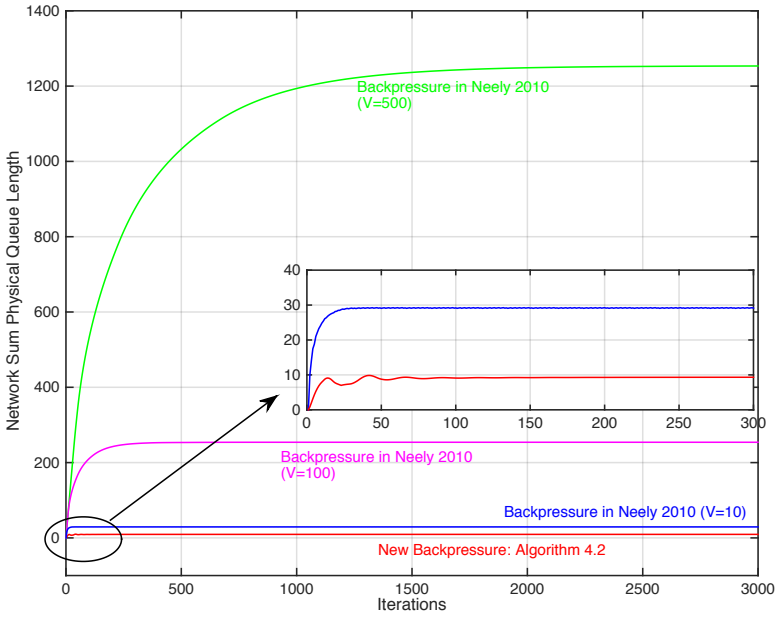

Figure 4.4: Actual queue length comparison between Algorithm 4.2 and the backpressure algorithm in [Nee10].

utility-delay tradeoff attained by existing backpressure algorithms [ES06, NMR05, LS04, LSXS15].

4.6 Supplement to this Chapter

4.6.1 Multi-Path Network Utility Maximization with Predetermined Paths

Consider the multi-path network utility maximization in [LS06] where each session has multiple given paths. Let x_f be the total source rate of each session $f \in \mathcal{F}$. Let $\mathcal{P}_f$ be the set of paths for session f. The link session rate $\mu_l^{(f)}$ becomes a vector $\boldsymbol{\mu}_l^{(f)} = [\mu_l^{(f,j)}]_{j\in\mathcal{P}_f}$. (Note that multiple paths for the same session are allowed to overlap.) Define $\mathcal{S}_l^{(f)}$ as the set of paths for session f that are allowed to use link l. Note that $\mathcal{S}_l^{(f)}$ are determined by the given paths for each session. That is, if path j for session f uses link l, then $j \in \mathcal{S}_l^{(f)}$; if no given path for session f uses link l, then $\mathcal{S}_l^{(f)} = \emptyset$. The multi-path network utility maximization problem can

be formulated as follows:

$$
\begin{aligned}
&\max \sum_{f\in\mathcal{F}} U_f(x_f)\\
&\text{s.t. } x_f \mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)}\sum_{j\in\mathcal{P}_f} \mu_l^{(f,j)} \le \sum_{l\in\mathcal{O}(n)}\sum_{j\in\mathcal{P}_f} \mu_l^{(f,j)}, f\in\mathcal{F}, \forall n\in\mathcal{N}\setminus\{\mathrm{Dst}(f)\}\\
&\quad \sum_{f\in\mathcal{F}}\sum_{j\in\mathcal{P}_f} \mu_l^{(f,j)} \le C_l, \forall l\in\mathcal{L},\\
&\quad \mu_l^{(f,j)} \ge 0, \forall l\in\mathcal{L}, \forall f\in\mathcal{F}, \forall j\in\mathcal{S}_l^{(f)},\\
&\quad \mu_l^{(f,j)} = 0, \forall l\in\mathcal{L}, \forall f\in\mathcal{F}, \forall j\in\mathcal{P}_f\setminus\mathcal{S}_l^{(f)},\\
&\quad x_f\in \mathrm{dom}(U_f), \forall f\in\mathcal{F}
\end{aligned}
$$

The above formulation is in the form of the problem (4.1)-(4.6) except that the variable dimension is extended.

In this case, Algorithm 4.2 developed to solve the problem (4.1)-(4.6) can be adapted to solve the above multi-path network utility maximization problem by replacing $\mu_l^{(f)}$ with $\sum_{j\in\mathcal{P}_f}\mu_l^{(f,j)}$ in updates (4.9) and (4.13); and replacing the subproblem (4.16)-(4.19) with

$$
\begin{aligned}
&\max_{\mu_{(n,m)}^{(f)}} \sum_{f\in\mathcal{F}} \left[W_n^{(f)}(t) - W_m^{(f)}(t)\right] \sum_{j\in\mathcal{P}_f} \mu_{(n,m)}^{(f,j)} - \left(\alpha_n+\alpha_m\right) \sum_{f\in\mathcal{F}}\sum_{j\in\mathcal{P}_f} \left[\mu_{(n,m)}^{(f,j)} - \mu_{(n,m)}^{(f,j)}(t-1)\right]^2\\
&\text{s.t.} \quad \sum_{f\in\mathcal{F}}\sum_{j\in\mathcal{P}_f} \mu_{(n,m)}^{(f)} \le C_{(n,m)}\\
&\quad \mu_{(n,m)}^{(f,j)} \ge 0, \forall f\in\mathcal{F}, \forall j\in\mathcal{S}_{(m,n)}^{(f)}\\
&\quad \mu_{(n,m)}^{(f,j)} = 0, \forall f\in\mathcal{F}, \forall j\notin\mathcal{S}_{(m,n)}^{(f)}
\end{aligned}
$$

which again has the same structure as the subproblem (4.16)-(4.19) except that the variable dimension is extended.

4.6.2 An Example Illustrating the Possibly Large Gap Between Model (4.7) and Model (4.8)

Consider a network example shown in Figure 4.5. The network has $3k+1$ nodes where only node 0 is a destination; and $a_i, i\in\{1,2,\ldots,k\}$ and $b_i, i\in\{1,2,\ldots,k\}$ can have exogenous arrivals. Assume all link capacities are equal to 1; and the exogenous arrivals are periodic with

period 2k, as follows:

- Time slot 1: One packet arrives at node a_1.
- Time slot 2: One packet arrives at node a_2.
- $\cdots$
- Time slot k: One packet arrives at node a_k.
- Time slot $k+1$: One packet arrives at node b_1.
- Time slot $k+2$: One packet arrives at node b_2.
- $\cdots$
- Time slot $2k$: One packet arrives at node b_k.

Under dynamics (4.7), each packet arrives on its own slot and traverses all links of its path to exit on the same slot it arrived. The queue backlog in each node is 0 for all time.

Under dynamics (4.8), the first packet arrives at time slot 1 to node a_1. This packet visits node a_2 at time slot 2, when the second packet also arrives at a_2. One of these packets is delivered to node a_3 at time slot 3, and another packet also arrives to node 3. The nodes $\{1, \ldots, k\}$ do not have any exogenous arrivals and act only to delay the delivery of all packets from the ai nodes. It follows that the link from node k to node 0 will send exactly one packet over each slots $t \in \{2k+1, 2k+2, \ldots, 2k+k\}$. Similarly, the link from b_k to 0 sends exactly one packet to node 0 over each of these same slots. Thus, node 0 receives 2 packets on each slot $t \in \{2k+1, 2k+2, \ldots, 2k+k\}$, but can only output 1 packet per slot. The queue backlog in this node grows linearly and reaches $k+1$ at time $2k+k$. Thus, the backlog in node 0 can be arbitrarily large when k is large. This example demonstrates that, even when there is only one destination, the deviation between virtual queues under dynamics (4.7) and actual queues under the dynamic (4.8) can be arbitrarily large, even when the in-degree and out-degree of 1 and an in-degree of at most 2.

4.6.3 Proof of Lemma 4.4

Note that the problem (4.20)-(4.22) satisfies the Slater condition. So the optimal solution to the problem (4.20)-(4.22) is characterized by KKT conditions [BV04]. Introducing a Lagrange

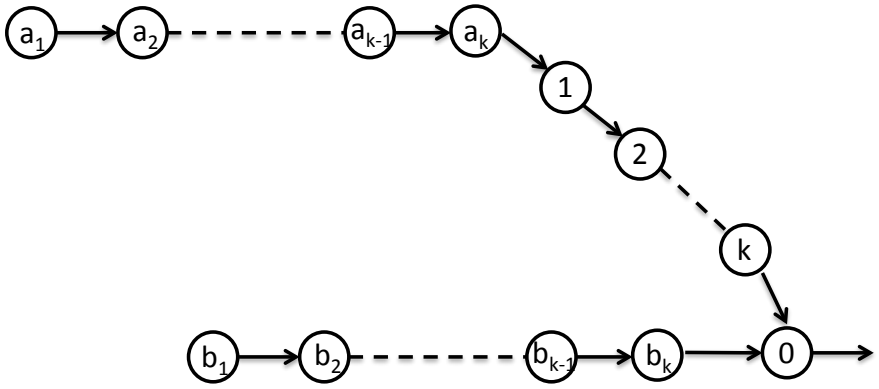

Figure 4.5: An example illustrating the possibly large gap between the queue model (4.7) and the queue model (4.8)

multiplier $\theta \in \mathbb{R}_+$ for the inequality constraint $\sum_{k=1}^K z_k \leq b$ and a Lagrange multiplier vector $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_K]^\mathsf{T} \in \mathbb{R}_+^K$ for the inequality constraints $z_k \geq 0, k \in \{1, 2, \ldots, K\}$. Let $\mathbf{z}^* = [z_1^*, \ldots, z_K^*]^\mathsf{T}$ and $(\theta^*, \boldsymbol{\nu}^*)$ be any primal and dual pair with the zero duality gap. By the KKT conditions, we have $z_k^* - a_k + \theta^* - \nu_k^* = 0, \forall k \in \{1, 2, \ldots, K\}; \sum_{k=1}^K z_k^* \leq b; \theta^* \geq 0; \theta^* \big(\sum_{k=1}^K z_k^* - b \big) = 0; z_k^* \geq 0, \forall k \in \{1, 2, \ldots, K\}; \nu_k^* \geq 0, \forall k \in \{1, 2, \ldots, K\}; \nu_k^* z_k^* = 0, \forall k \in \{1, 2, \ldots, K\}$.

Eliminating $\nu_k^*, \forall k \in \{1, 2, \ldots, K\}$ in all equations yields $\theta^* \geq a_k - z_k^*, k \in \{1, 2, \ldots, K\}$; $\sum_{k=1}^K z_k^* \leq b$; $\theta^* \geq 0$; $\theta^* \big(\sum_{k=1}^K z_k^* - b \big) = 0$; $z_k^* \geq 0, \forall k \in \{1, 2, \ldots, K\}$; $(z_k^* - a_k + \theta^*) z_k^* = 0, \forall k \in \{1, 2, \ldots, K\}$.

For all $k \in \{1, 2, \ldots, K\}$, we consider $\theta^* < a_k$ and $\theta^* \geq a_k$ separately:

1. If $\theta^* < a_k$, then $\theta^* \geq a_k - z_k^*$ holds only when $z_k^* > 0$, which by $(z_k^* - a_k + \theta^*) z_k^* = 0$ implies that $z_k^* = a_k - \theta^*$.

2. If $\theta^* \geq a_k$, then $z_k^* > 0$ is impossible, because $z_k^* > 0$ implies that $z_k^* - a_k + \theta^* > 0$, which together with $z_k^* > 0$ contradicts the slackness condition $(z_k^* - a_k + \theta^*) z_k^* = 0$. Thus, if $\theta^* \geq a_k$, we must have $z_k^* = 0$.

Summarizing both cases, we have $z_k^* = \max\{0, a_k - \theta^*\}, \forall k \in \{1, 2, \ldots, K\}$, where θ^* is chosen such that $\sum_{k=1}^K z_k^* \leq b$, $\theta^* \geq 0$ and $\theta^* \big(\sum_{k=1}^K z_k^* - b \big) = 0$.

To find such θ^*, we first check if $\theta^* = 0$. If $\theta^* = 0$ is true, the slackness condition $\theta^* \big(\sum_{k=1}^K z_k^* - b \big)$ is guaranteed to hold and we need to further require $\sum_{k=1}^K z_k^* = \sum_{k=1}^K \max\{0, a_k\} \leq b$. Thus $\theta^* = 0$ if and only if $\sum_{k=1}^K \max\{0, a_k\} \leq b$. Thus, Algorithm 4.3 check if $\sum_{k=1}^K \max\{0, a_k\} \leq b$ holds at the first step and if this is true, then we conclude $\theta^* = 0$ and we are done!

Otherwise, we know $\theta^* > 0$. By the slackness condition $\theta^*\big(\sum_{k=1}^K z_k^* - b\big) = 0$, we must have $\sum_{k=1}^K z_k^* = \sum_{k=1}^K \max\{0, a_k - \theta^*\} = b$. To find $\theta^* > 0$ such that $\sum_{k=1}^K \max\{0, a_k - \theta^*\} = b$, we can apply a bisection search by noting that all z_k^* are decreasing with respect to θ^*.

Another algorithm of finding θ^* is inspired by the observation that if $a_j \geq a_i, \forall i, j \in \{1, 2, \ldots, K\}$, then $z_j^* \geq z_i^*$. Thus, we first sort all a_k in a decreasing order, say π is the permutation such that $a_{\pi(1)} \geq a_{\pi(2)} \geq \cdots \geq a_{\pi(K)}$; and then sequentially check if $k \in \{1, 2, \ldots, K\}$ is the index such that $a_{\pi(k)} - \theta^* \geq 0$ and $a_{\pi(k+1)} - \theta^* < 0$. To check this, we first assume k is indeed such an index and solve the equation $\sum_{j=1}^k (a_{\pi(j)} - \theta^*) = b$ to obtain θ^*; (Note that in Algorithm 4.3, to avoid recalculating the partial sum $\sum_{j=1}^k a_{\pi(j)}$ for each k, we introduce the parameter $S_k = \sum_{j=1}^k a_{\pi(j)}$ and update S_k incrementally. By doing this, the complexity of each iteration in the loop is only $O(1)$.) then verify the assumption by checking if $\theta^* \geq 0$, $a_{\pi(k)} - \theta^* \geq 0$ and $a_{\pi(k+1)} - \theta^* \leq 0$. The algorithm is described in Algorithm 4.3 and has complexity $O(K \log(K))$. The overall complexity is dominated by the step of sorting all a_k.

4.6.4 Proof of Lemma 4.6

The objective function (4.30) can be rewritten as

$$
\begin{aligned}
& f(\mathbf{y}) - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \big[W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f)}) + \alpha_n \|\mathbf{y}_n^{(f)} - \mathbf{y}_n^{(f)}(t-1)\|^2\big] - \sum_{\substack{f\in\mathcal{F},\\ n=\mathrm{Dst}(f)}} \alpha_n \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2 \\
&\overset{(a)}{=} \sum_{f\in\mathcal{F}} U_f(x_f) - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} W_n^{(f)}(t)\big[x_f \mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)} - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}\big] \\
&\quad - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \alpha_n \big[[x_f - x_f(t-1)]^2 \mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2 + \sum_{l\in\mathcal{O}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2\big] \\
&\quad - \sum_{\substack{f\in\mathcal{F},\\ n=\mathrm{Dst}(f)}} \alpha_n \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2 \\
&\overset{(b)}{=} \sum_{f\in\mathcal{F}} \big[U_f(x_f) - W_{\mathrm{Src}(f)}^{(f)}(t) x_f - \alpha_{\mathrm{Src}(f)}[x_f - x_f(t-1)]^2\big] + \sum_{(n,m)\in\mathcal{L}} \sum_{f\in\mathcal{F}} [W_n^{(f)}(t) - W_m^{(f)}(t)] \mu_{(n,m)}^{(f)} \\
&\quad - \sum_{(n,m)\in\mathcal{L}} (\alpha_n + \alpha_m) \sum_{f\in\mathcal{F}} [\mu_{(n,m)}^{(f)} - \mu_{(n,m)}^{(f)}(t-1)]^2
\end{aligned} \tag{4.33}
$$

where (a) follows from the fact that $g_n^{(f)}(\mathbf{y}_n^{(f)}) = x_f \mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} \mu_l^{(f)} - \sum_{l\in\mathcal{O}(n)} \mu_l^{(f)}$ and $\|\mathbf{y}_n^{(f)} - \mathbf{y}_n^{(f)}(t-1)\|^2 = [x_f - x_f(t-1)]^2 \mathbf{1}_{\{n=\mathrm{Src}(f)\}} + \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2 + \sum_{l\in\mathcal{O}(n)} [\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2$; and (b) follows by collecting each linear term $\mu_l^{(f)}$ and each quadratic term $[\mu_l^{(f)} - \mu_l^{(f)}(t-1)]^2$. Note that each link session rate $\mu_l^{(f)}$ appears twice with opposite signs in the

summation term $\sum_{f\in\mathcal{F},n\in\mathcal{N}\setminus\{\mathrm{Dst}(f)\}} W_n^{(f)}(t)\big[x_f\mathbf{1}_{\{n=\mathrm{Src}(f)\}}+\sum_{l\in\mathcal{I}(n)}\mu_l^{(f)}-\sum_{l\in\mathcal{O}(n)}\mu_l^{(f)}\big]$ unless link l flows into $\mathrm{Dst}(f)$ and recall that $W_{\mathrm{Dst}(f)}^{(f)}=0,\forall f\in\mathcal{F}$. The quadratic terms are collected in a similar way. Note that the term $\sum_{f\in\mathcal{F},n=\mathrm{Dst}(f)}\alpha_n\sum_{l\in\mathcal{I}(n)}[\mu_l^{(f)}-\mu_l^{(f)}(t-1)]^2$ introduced to the objective function (4.30) is necessary to guarantee each quadratic term $[\mu_{(m,n)}^{(f)}-\mu_{(m,n)}^{(f)}(t-1)]^2$ with the same link index (n,m) but different flow indices $f\in\mathcal{F}$ to have the same coefficient $\alpha_n+\alpha_m$ in (4.33).

Note that the equation (4.33) is now separable for each scalar x_f and vector $[\mu_{(n,m)}^{(f)}]_{f\in\mathcal{F}}$. Thus, the problem (4.30)-(4.31) can be decomposed into independent smaller optimization problems in the form of the problem (4.14)-(4.15) with respect to each scalar x_f, and in the form of the problem (4.16)-(4.19) with respect to each vector $[\mu_{(n,m)}^{(f)}]_{f\in\mathcal{F}}$.

4.6.5 Proof of Lemma 4.7

Note that $W_n^{(f)}(t)$ appears as a known constant in (4.14). Since $U_f(x_f)$ is concave and $W_n^{(f)}(t)x_f$ is linear, it follows that (4.14) is strongly concave with respect to x_f with modulus $2\alpha_n$. Since $x_f(t)$ is chosen to solve (4.14)-(4.15), by Corollary 1.3, $\forall f\in\mathcal{F}$, we have

$$\begin{aligned}&\underbrace{U_f(x_f(t))-W_{\mathrm{Src}(f)}^{(f)}(t)x_f(t)-\alpha_n[x_f(t)-x_f(t-1)]^2}_{\text{(4.34)-I}}\\ \geq&\underbrace{U_f(x_f^*)-W_{\mathrm{Src}(f)}^{(f)}(t)x_f^*-\alpha_n[x_f^*-x_f(t-1)]^2+\alpha_n[x_f^*-x_f(t)]^2}_{\text{(4.34)-II}}.\end{aligned}\tag{4.34}$$

Similarly, we know (4.16) is strongly concave with respect to vector $[\mu_{(n,m)}^{f}]_{f\in\mathcal{F}}$ with modulus $2(\alpha_n+\alpha_m)$. By Corollary 1.3, $\forall(n,m)\in\mathcal{O}(n)$, we have

$$\begin{aligned}&\underbrace{\sum_{f\in\mathcal{F}}[W_n^{(f)}(t)-W_m^{(f)}(t)]\mu_{(n,m)}^{(f)}(t)-(\alpha_n+\alpha_m)\sum_{f\in\mathcal{F}}[\mu_{(n,m)}^{(f)}(t)-\mu_{(n,m)}^{(f)}(t-1)]^2}_{\text{(4.35)-I}}\\ \geq&\underbrace{\sum_{f\in\mathcal{F}}[W_n^{(f)}(t)-W_m^{(f)}(t)]\mu_{(n,m)}^{(f),*}-(\alpha_n+\alpha_m)\sum_{f\in\mathcal{F}}[\mu_{(n,m)}^{(f),*}-\mu_{(n,m)}^{(f)}(t-1)]^2+(\alpha_n+\alpha_m)\sum_{f\in\mathcal{F}}[\mu_{(n,m)}^{(f),*}-\mu_{(n,m)}^{(f)}(t)]^2}_{\text{(4.35)-II}}.\end{aligned}\tag{4.35}$$

Recall that each column vector $\mathbf{y}_n^{(f)}$ defined in (4.23) is composed by control actions that appear in each constraint (4.2); the column vector $\mathbf{y}=[x_f;\mu_l^{(f)}]_{f\in\mathcal{F},l\in\mathcal{L}}$ is the collection of all

control actions; and $f(\mathbf{y}) = \sum_{f\in\mathcal{F}} U_f(x_f)$. Summing the term (4.34)-I over all $f \in \mathcal{F}$ and the term (4.35)-I over all $(n,m) \in \mathcal{L}$ and using an argument similar to the proof of Lemma 4.6 (Recall that $\mathbf{y}(t)$ is jointly chosen to minimize (4.30) by Lemma 4.6.) yields

$$\begin{aligned}
&\sum_{f\in\mathcal{F}} (4.34)\text{-I} + \sum_{(n,m)\in\mathcal{N}} (4.35)\text{-I} \\
=&f(\mathbf{y}(t)) - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus \mathrm{Dst}(f)}} \left[W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \alpha_n \|\mathbf{y}_n^{(f)}(t) - \mathbf{y}_n^{(f)}(t-1)\|^2\right] \\
&- \sum_{\substack{f\in\mathcal{F},\\ n=\mathrm{Dst}(f)}} \alpha_n \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)}(t) - \mu_l^{(f)}(t-1)]^2. \qquad (4.36)
\end{aligned}$$

Recall that $\Phi(t) = \sum_{f\in\mathcal{F}, n\in\mathcal{N}} \left[\alpha_n \mathbf{1}_{\{n\neq \mathrm{Dst}(f)\}} \|\mathbf{y}_n^{(f),*} - \mathbf{y}_n^{(f)}(t)\|^2 + \alpha_n \mathbf{1}_{\{n=\mathrm{Dst}(f)\}} \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f),*} - \mu_l^{(f)}(t)]^2\right]$. Summing the term (4.34)-II over all $f \in \mathcal{F}$ and the term (4.35)-II over all $(n,m) \in \mathcal{L}$ yields

$$\sum_{f\in\mathcal{F}} (4.34)\text{-II} + \sum_{(n,m)\in\mathcal{N}} (4.35)\text{-II} = f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus \mathrm{Dst}(f)}} W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f),*}), \qquad (4.37)$$

Combining (4.34)-(4.37) and rearranging terms yields

$$\begin{aligned}
&f(\mathbf{y}(t)) \\
\geq& f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) - \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus \mathrm{Dst}(f)}} W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f),*}) + \sum_{\substack{f\in\mathcal{F},\\ n=\mathrm{Dst}(f)}} \alpha_n \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)}(t) - \mu_l^{(f)}(t-1)]^2 \\
&+ \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus \mathrm{Dst}(f)}} \left[W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \alpha_n \|\mathbf{y}_n^{(f)}(t) - \mathbf{y}_n^{(f)}(t-1)\|^2\right] \\
\overset{(a)}{\geq}& f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus \mathrm{Dst}(f)}} \left[W_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \alpha_n \|\mathbf{y}_n^{(f)}(t) - \mathbf{y}_n^{(f)}(t-1)\|^2\right] \\
\overset{(b)}{=}& f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) \\
&+ \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus \mathrm{Dst}(f)}} \left[Q_n^{(f)}(t) g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1)) g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \alpha_n \|\mathbf{y}_n^{(f)}(t) - \mathbf{y}_n^{(f)}(t-1)\|^2\right] \qquad (4.38)
\end{aligned}$$

where (a) follows because $g_n^{(f)}(\mathbf{y}_n^{(f),*}) = 0$, $\forall f \in \mathcal{F}, \forall n \in \mathcal{N} \setminus \mathrm{Dst}(f)$, and

$$\sum_{f\in\mathcal{F}, n=\mathrm{Dst}(f)} \alpha_n \sum_{l\in\mathcal{I}(n)} [\mu_l^{(f)}(t) - \mu_l^{(f)}(t-1)]^2 \geq 0;$$

and (b) follows because $W_n^{(f)}(t) = Q_n^{(f)}(t) + g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1))$.

Recall that $u_1u_2 = \frac{1}{2}u_1^2 + \frac{1}{2}u_2^2 - \frac{1}{2}(u_1 - u_2)^2$ for any $u_1, u_2 \in \mathbb{R}$. Thus, for all $f \in \mathcal{F}, n \in \mathcal{N} \setminus \mathrm{Dst}(f)$, we have

$$\begin{aligned}
&g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1))g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) \\
=&\frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1))]^2 + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2 - \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1)) - g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2. \qquad (4.39)
\end{aligned}$$

Substituting (4.39) into (4.38) yields

$$\begin{aligned}
&f(\mathbf{y}(t)) \\
\geq& f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \Big[Q_n^{(f)}(t)g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1))]^2 \\
&+ \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2 - \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1)) - g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2 + \alpha_n\|\mathbf{y}_n^{(f)}(t) - \mathbf{y}_n^{(f)}(t-1)\|^2\Big] \\
\overset{(a)}{\geq}& f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \Big[Q_n^{(f)}(t)g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1))]^2 \\
&+ \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2 + \big(\alpha_n - \frac{1}{2}\beta_n^2\big)\|\mathbf{y}_n^{(f)}(t) - \mathbf{y}_n^{(f)}(t-1)\|^2\Big] \\
\overset{(b)}{\geq}& f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \sum_{\substack{f\in\mathcal{F},\\ n\in\mathcal{N}\setminus\mathrm{Dst}(f)}} \big[Q_n^{(f)}(t)g_n^{(f)}(\mathbf{y}_n^{(f)}(t)) + \frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t))]^2\big] \qquad (4.40)
\end{aligned}$$

where (a) follows from the Fact 4.2, i.e., each $g_n^{(f)}(\cdot)$ is Lipschitz continuous with modulus β_n and (b) follows because $\alpha_n \geq \frac{1}{2}(d_n + 1)$, $\beta_n \leq \sqrt{d_n + 1}$ and $\frac{1}{2}[g_n^{(f)}(\mathbf{y}_n^{(f)}(t-1))]^2 \geq 0$.

Substituting (4.28) into (4.40) yields

$$f(\mathbf{y}(t)) \geq f(\mathbf{y}^*) + \Phi(t) - \Phi(t-1) + \Delta(t).$$

Chapter 5

Online Convex Optimization with Stochastic Constraints

Online convex optimization (OCO) is a multi-round learning process with arbitrarily-varying convex loss functions where the decision maker has to choose decision $x(t) \in \mathcal{X}$ before observing the corresponding convex loss function $f^t(\cdot)$. For a fixed time horizon T, define the *regret* of a learning algorithm with respect to the best fixed decision in hindsight (with full knowledge of all loss functions) as

$$\text{regret}(T) = \sum_{t=1}^{T} f^t(\mathbf{x}(t)) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f^t(\mathbf{x}).$$

The best fixed decision $\mathbf{x}^* = \text{argmin}_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f^t(\mathbf{x})$ typically cannot be implemented. That is because it would need to be determined before the start of the first round, and this would require knowledge of the future $f^t(\cdot)$ functions for all $t \in \{1, 2, \ldots, T\}$. However, to avoid being embarrassed by the situation where our performance is significantly exceeded by a stubborn decision maker guessing $\mathbf{x}^*$ correctly by luck, a desired learning algorithm should have a small regret. Specifically, we desire a learning algorithm for which regret(T) grows sub-linearly with respect to T, i.e., the difference of average loss tends to zero as T goes to infinity when comparing the dynamic learning algorithm and a lucky stubborn decision maker. The setting of OCO is introduced in a series of works [CBLW96, KW97, Gor99, Zin03] and is formalized in [Zin03]. OCO has gained considerable amount of research interest recently with various applications such as online regression, prediction with expert advice, online ranking, online shortest paths, and portfolio selection. See [SS11, Haz16] for more applications and background.

In [Zin03], Zinkevich shows that $O(\sqrt{T})$ regret can be achieved by using an online gradient

descent (OGD) update given by

$$\mathbf{x}(t+1) = \mathcal{P}_{\mathcal{X}}\big[\mathbf{x}(t) - \gamma \nabla f^t(\mathbf{x}(t))\big] \tag{5.1}$$

where γ is the step size, also known as the learning rate, $\nabla f^t(\cdot)$ is a subgradient of $f^t(\cdot)$ and $\mathcal{P}_{\mathcal{X}}[\cdot]$ is the projection onto set $\mathcal{X}$. Hazan et al. in [HAK07] show that better regret is possible under the assumption that each loss function is strongly convex but $O(\sqrt{T})$ is the best possible if no additional assumption is imposed.

Zinkevich's OGD in (5.1) requires the full knowledge of set $\mathcal{X}$ and low complexity of the projection $\mathcal{P}_{\mathcal{X}}[\cdot]$. However, in practice, the constraint set $\mathcal{X}$, which is often described by many functional inequality constraints, can be time varying and may not be fully disclosed to the decision maker. In [MTY09], Mannor et al. extend OCO by considering time-varying constraint functions $g^t(\mathbf{x})$ which can arbitrarily vary and are only disclosed to us after each $\mathbf{x}(t)$ is chosen. In this setting, Mannor et al. in [MTY09] explore the possibility of designing learning algorithms such that regret grows sub-linearly and $\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} g^t(\mathbf{x}(t)) \leq 0$, i.e., the (cumulative) constraint violation $\sum_{t=1}^{T} g^t(\mathbf{x}(t))$ also grows sub-linearly. Unfortunately, Mannor et al. in [MTY09] prove that this is impossible even when both $f^t(\cdot)$ and $g^t(\cdot)$ are simple linear functions.

Given the impossibility results shown by Mannor et al. in [MTY09], this chapter considers OCO where constraint functions $g^t(\mathbf{x})$ are not arbitrarily varying but independently and identically distributed (i.i.d.) generated from an unknown probability model (and functions $f^t(\mathbf{x})$ are still arbitrarily varying and possibly non-i.i.d.). Specifically, this chapter considers *online convex optimization (OCO) with stochastic constraint* $\mathcal{X} = \{\mathbf{x} \in \mathcal{X}_0 : \mathbb{E}_\omega[g_k(\mathbf{x};\omega)] \leq 0, k \in \{1,2,\ldots,m\}\}$ where $\mathcal{X}_0$ is a known fixed set; the expressions of stochastic constraints $\mathbb{E}_\omega[g_k(\mathbf{x};\omega)]$ (involving expectations with respect to ω from an unknown distribution) are unknown; and subscripts $k \in \{1,2,\ldots,m\}$ indicate the possibility of multiple functional constraints. In OCO with stochastic constraints, the decision maker receives loss function $f^t(\mathbf{x})$ and i.i.d. constraint function realizations $g_k^t(\mathbf{x}) \overset{\Delta}{=} g_k(\mathbf{x};\omega(t))$ at each round t. However, the expressions of $g_k^t(\cdot)$ and $f^t(\cdot)$ are disclosed to the decision maker only after decision $\mathbf{x}(t) \in \mathcal{X}_0$ is chosen. This setting arises naturally when decisions are restricted by stochastic environments or deterministic environments with noisy observations. For example, if we consider online routing (with link capacity constraints) in wireless networks [MTY09], each link capacity is not a fixed constant (as in wireline

networks) but an i.i.d. random variable since wireless channels are stochastically time-varying by nature [TV05]. OCO with stochastic constraints also covers important special cases such as OCO with long term constraints [MJY12, CGP15, JHA16], stochastic constrained convex optimization [MYJ13] and deterministic constrained convex optimization [Nes04].

Let $\mathbf{x}^* = \text{argmin}_{\{\mathbf{x}\in\mathcal{X}_0:\mathbb{E}[g_k(\mathbf{x};\omega)]\leq 0,\forall k\in\{1,2,\ldots,m\}\}} \sum_{t=1}^{T} f^t(\mathbf{x})$ be the best fixed decision in hindsight (knowing all loss functions $f^t(\mathbf{x})$ and the distribution of stochastic constraint functions $g_k(\mathbf{x};\omega)$). Thus, $\mathbf{x}^*$ minimizes the T-round cumulative loss and satisfies all stochastic constraints in expectation, which also implies $\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} g_k^t(\mathbf{x}^*) \leq 0$ almost surely by the strong law of large numbers. Our goal is to develop dynamic learning algorithms that guarantee both regret $\sum_{t=1}^{T} f^t(\mathbf{x}(t)) - \sum_{t=1}^{T} f^t(\mathbf{x}^*)$ and constraint violations $\sum_{t=1}^{T} g_k^t(\mathbf{x}(t))$ grow sub-linearly.

Note that Zinkevich's algorithm in (5.1) is not applicable to OCO with stochastic constraints since $\mathcal{X}$ is unknown and it can happen that $\mathcal{X}(t) = \{\mathbf{x} \in \mathcal{X}_0 : g_k(\mathbf{x};\omega(t)) \leq 0, \forall k \in \{1,2,\ldots,m\}\} = \emptyset$ for certain realizations $\omega(t)$, so that projections $\mathcal{P}_{\mathcal{X}}[\cdot]$ or $\mathcal{P}_{\mathcal{X}(t)}[\cdot]$ required in (5.1) are not even well-defined.

Our Contributions

This chapter solves online convex optimization with stochastic constraints. In particular, we propose a new learning algorithm that is proven to achieve $O(\sqrt{T})$ expected regret and constraint violations and $O(\sqrt{T}\log(T))$ high probability regret and constraint violations. The results in this chapter are originally developed in our paper [YNW17]. The proposed new algorithm also improves upon state-of-the-art results in the following special cases:

- *OCO with long term constraints:* This is a special case where each $g_k^t(\mathbf{x}) \equiv g_k(\mathbf{x})$ is known and does not depend on time. Note that $\mathcal{X} = \{\mathbf{x} \in \mathcal{X}_0 : g_k(\mathbf{x}) \leq 0, \forall k \in \{1,2,\ldots,m\}\}$ can be complicated while $\mathcal{X}_0$ might be a simple hypercube. To avoid high complexity involved in the projection onto $\mathcal{X}$ as in Zinkevich's algorithm, work in [MJY12, CGP15, JHA16] develops low complexity algorithms that use projections onto a simpler set $\mathcal{X}_0$ by allowing $g_k(\mathbf{x}(t)) > 0$ for certain rounds but ensuring $\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq 0$. The best existing performance is $O(T^{\max\{\beta,1-\beta\}})$ regret and $O(T^{1-\beta/2})$ constraint violations where $\beta \in (0,1)$ is an algorithm parameter [JHA16]. This gives $O(\sqrt{T})$ regret with worse $O(T^{3/4})$ constraint violations or $O(\sqrt{T})$ constraint violations with worse $O(T)$ regret. In contrast, our algorithm, which only uses projections onto $\mathcal{X}_0$ as shown in Lemma 5.1, can achieve

$O(\sqrt{T})$ regret and $O(\sqrt{T})$ constraint violations simultaneously. In Chapter 6, we focus on OCO with long term constraints and further develop a different algorithm that can only solve "OCO with long term constraints" but can achieve $O(\sqrt{T})$ regret and $O(1)$ constraint violations.

- *Stochastic constrained convex optimization:* This is a special case where each $f^t(\mathbf{x})$ is i.i.d. generated from an unknown distribution. This problem has many applications in operations research and machine learning such as Neyman-Pearson classification and risk-mean portfolio. The work [MYJ13] develops a (batch) offline algorithm that produces a solution with high probability performance guarantees only after sampling the problems for sufficiently many times. That is, during the process of sampling, there is no performance guarantee. The work [LZ16] proposes a stochastic approximation based (batch) offline algorithm for stochastic convex optimization with one single stochastic functional inequality constraint. In contrast, our algorithm is an online algorithm with online performance guarantees and can deal with an arbitrary number of stochastic constraints.

- *Deterministic constrained convex optimization:* This is a special case where each $f^t(\mathbf{x}) \equiv f(\mathbf{x})$ and $g_k^t(\mathbf{x}) \equiv g_k(\mathbf{x})$ are known and do not depend on time. In this case, the goal is to develop a fast algorithm that converges to a good solution (with a small error) with a few number of iterations; and our algorithm with $O(\sqrt{T})$ regret and constraint violations is equivalent to an iterative numerical algorithm with an $O(1/\sqrt{T})$ convergence rate. Our algorithm is subgradient based and does not require the smoothness or differentiability of the convex program. Indeed, our algorithm when used to solve general (possibly non-smooth) deterministic constrained convex programs is a third new Lagrangian method developed in this thesis. Recall that in Chapter 3, we have developed two other Lagrangian methods that can only solve deterministic constrained convex optimization but can achieve a faster $O(1/T)$ convergence rate. The algorithm developed in this chapter is a primal-dual type one since its primal update follows a projected gradient dynamic, has an $O(1/\sqrt{T})$ convergence rate, and does not require any knowledge of the optimal Lagrange multiplier vector. Recall that the primal-dual subgradient method Algorithm 1.2 has the same $O(1/\sqrt{T})$ convergence rate but requires an upper bound of optimal Lagrange multipliers, which is usually unknown in practice.

.

5.1 Problem Statement and New Algorithm

Let $\mathcal{X}_0$ be a known fixed compact convex set. Let $f^t(\mathbf{x})$ be a sequence of arbitrarily-varying convex functions. Let $g_k(\mathbf{x};\omega(t)), k \in \{1,2,\ldots,m\}$ be sequences of functions that are i.i.d. realizations of stochastic constraint functions $\tilde{g}_k(\mathbf{x}) \triangleq \mathbb{E}_\omega[g_k(\mathbf{x};\omega)]$ with random variable $\omega \in \Omega$ from an unknown distribution. That is, $\omega(t)$ are i.i.d. samples of ω. Assume that each $f^t(\cdot)$ is independent of all $\omega(\tau)$ with $\tau \geq t+1$ so that we are unable to predict future constraint functions based on the knowledge of the current loss function. For each $\omega \in \Omega$, we assume $g_k(\mathbf{x};\omega)$ are convex with respect to $\mathbf{x} \in \mathcal{X}_0$. At the beginning of each round t, neither the loss function $f^t(\mathbf{x})$ nor the constraint function realizations $g_k(\mathbf{x};\omega(t))$ are known to the decision maker. However, the decision maker still needs to make a decision $\mathbf{x}(t) \in \mathcal{X}_0$ for round t; and after that $f^t(\mathbf{x})$ and $g_k(\mathbf{x},\omega(t))$ are disclosed to the decision maker at the end of round t.

For convenience, we often suppress the dependence of each $g_k(\mathbf{x};\omega(t))$ on $\omega(t)$ and write $g_k^t(\mathbf{x}) = g_k(\mathbf{x};\omega(t))$. Recall $\tilde{g}_k(\mathbf{x}) = \mathbb{E}_\omega[g_k(\mathbf{x};\omega)]$ where the expectation is with respect to ω. Define $\mathcal{X} = \{\mathbf{x} \in \mathcal{X}_0 : \tilde{g}_k(\mathbf{x}) = \mathbb{E}[g_k(\mathbf{x};\omega)] \leq 0, \forall k \in \{1,2,\ldots,m\}\}$. We further define the stacked vector of multiple functions $g_1^t(\mathbf{x}),\ldots,g_m^t(\mathbf{x})$ as $\mathbf{g}^t(\mathbf{x}) = [g_1^t(\mathbf{x}),\ldots,g_m^t(\mathbf{x})]^\mathsf{T}$ and define $\tilde{\mathbf{g}}(\mathbf{x}) = [\mathbb{E}_\omega[g_1(\mathbf{x};\omega)],\ldots,\mathbb{E}_\omega[g_m(\mathbf{x};\omega)]]^\mathsf{T}$. We use $\|\cdot\|$ to denote the Euclidean norm for a vector. Throughout this chapter, we have the following assumptions:

Assumption 5.1 (Basic Assumptions)**.**

- *Loss functions $f^t(\mathbf{x})$ and constraint functions $g_k(\mathbf{x};\omega)$ have bounded subgradients on $\mathcal{X}_0$. That is, there exists $D_1 > 0$ and $D_2 > 0$ such that $\|\nabla f^t(\mathbf{x})\| \leq D_1$ for all $\mathbf{x} \in \mathcal{X}_0$ and all $t \in \{0,1,\ldots\}$ and $\|\nabla g_k(\mathbf{x};\omega)\| \leq D_2$ for all $\mathbf{x} \in \mathcal{X}_0$, all $\omega \in \Omega$ and all $k \in \{1,2,\ldots,m\}$.*
- *There exists constant $G > 0$ such that $\|\mathbf{g}(\mathbf{x};\omega)\| \leq G$ for all $\mathbf{x} \in \mathcal{X}_0$ and all $\omega \in \Omega$.*
- *There exists constant $R > 0$ such that $\|\mathbf{x}-\mathbf{y}\| \leq R$ for all $\mathbf{x},\mathbf{y} \in \mathcal{X}_0$.*

Assumption 5.2 (Interior Point Assumption)**.** *There exists $\epsilon > 0$ and $\hat{\mathbf{x}} \in \mathcal{X}_0$ such that $\tilde{g}_k(\hat{\mathbf{x}}) = \mathbb{E}_\omega[g_k(\hat{\mathbf{x}};\omega)] \leq -\epsilon$ for all $k \in \{1,2,\ldots,m\}$.*

5.1.1 New Algorithm

Now consider the following algorithm described in Algorithm 5.1. This algorithm chooses $\mathbf{x}(t+1)$ as the decision for round $t+1$ based on $f^t(\cdot)$ and $\mathbf{g}^t(\cdot)$ without requiring $f^{t+1}(\cdot)$ or $\mathbf{g}^{t+1}(\cdot)$.

Algorithm 5.1 New Algorithm for Online Convex Optimization with Stochastic Constraints

Let $V > 0$ and $\alpha > 0$ be constant algorithm parameters. Choose $\mathbf{x}(1) \in \mathcal{X}_0$ arbitrarily and let $Q_k(1) = 0, \forall k \in \{1, 2, \ldots, m\}$. At the end of each round $t \in \{1, 2, \ldots\}$, observe $f^t(\cdot)$ and $\mathbf{g}^t(\cdot)$ and do the following:

- Choose $\mathbf{x}(t+1)$ that solves

$$\min_{\mathbf{x} \in \mathcal{X}_0} \left\{ V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + \sum_{k=1}^{m} Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + \alpha \|\mathbf{x} - \mathbf{x}(t)\|^2 \right\} \quad (5.2)$$

  as the decision for the next round $t+1$, where $\nabla f^t(\mathbf{x}(t))$ is a subgradient of $f^t(\mathbf{x})$ at point $\mathbf{x} = \mathbf{x}(t)$ and $\nabla g_k^t(\mathbf{x}(t))$ is a subgradient of $g_k^t(\mathbf{x})$ at point $\mathbf{x} = \mathbf{x}(t)$.

- Update each virtual queue $Q_k(t+1), \forall k \in \{1, 2, \ldots, m\}$ via

$$Q_k(t+1) = \max\left\{ Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)], 0 \right\}. \quad (5.3)$$

The next lemma summarizes that $\mathbf{x}(t+1)$ update in (5.2) can be implemented via a simple projection onto $\mathcal{X}_0$.

Lemma 5.1. *The* $\mathbf{x}(t+1)$ *update in* (5.2) *is given by*

$$\mathbf{x}(t+1) = \mathcal{P}_{\mathcal{X}_0}\left[\mathbf{x}(t) - \frac{1}{2\alpha}\mathbf{d}(t)\right]$$

where $\mathbf{d}(t) = V\nabla f^t(\mathbf{x}(t)) + \sum_{k=1}^{m} Q_k(t)\nabla g_k^t(\mathbf{x}(t))$ *and* $\mathcal{P}_{\mathcal{X}_0}[\cdot]$ *is the projection onto convex set* $\mathcal{X}_0$.

Proof. The projection by definition is

$$\min_{\mathbf{x} \in \mathcal{X}_0} \left\| \mathbf{x} - [\mathbf{x}(t) - \frac{1}{2\alpha}\mathbf{d}(t)] \right\|^2$$

and is equivalent to (5.2) since multiplying the expression to be minimized by $\alpha > 0$ does not change the minimizer. □

5.1.2 Intuitions of Algorithm 5.1

Note that if there are no stochastic constraints $g_k^t(\mathbf{x})$, i.e., $\mathcal{X} = \mathcal{X}_0$, then Algorithm 5.1 has $Q_k(t) \equiv 0, \forall t$ and becomes Zinkevich's algorithm with $\gamma = \frac{V}{2\alpha}$ in (5.1) since

$$\begin{aligned}\mathbf{x}(t+1) &\overset{(a)}{=} \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}_0} \big\{ \underbrace{V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}-\mathbf{x}(t)] + \alpha\|\mathbf{x}-\mathbf{x}(t)\|^2}_{\text{penalty}} \big\} \\ &\overset{(b)}{=} \mathcal{P}_{\mathcal{X}_0}\big[\mathbf{x}(t) - \frac{V}{2\alpha}\nabla f^t(\mathbf{x}(t))\big] \end{aligned} \tag{5.4}$$

where (a) follows from (5.2); and (b) follows from Lemma 5.1 by noting that $\mathbf{d}(t) = V\nabla f^t(\mathbf{x}(t))$. Call the term marked by an underbrace in (5.4) the *penalty*. Thus, Zinkevich's algorithm is to minimize the *penalty* term and is a special case of Algorithm 5.1 used to solve OCO over $\mathcal{X}_0$.

Let $\mathbf{Q}(t) = \big[Q_1(t), \ldots, Q_m(t)\big]^{\mathsf{T}}$ be the vector of virtual queue backlogs. Let $L(t) = \frac{1}{2}\|\mathbf{Q}(t)\|^2$ be a *Lyapunov function* and define *Lyapunov drift*

$$\Delta(t) = L(t+1) - L(t) = \frac{1}{2}[\|\mathbf{Q}(t+1)\|^2 - \|\mathbf{Q}(t)\|^2]. \tag{5.5}$$

The intuition behind Algorithm 5.1 is to choose $\mathbf{x}(t+1)$ to minimize an upper bound of the expression

$$\underbrace{\Delta(t)}_{\text{drift}} + \underbrace{V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}-\mathbf{x}(t)] + \alpha\|\mathbf{x}-\mathbf{x}(t)\|^2}_{\text{penalty}} \tag{5.6}$$

The intention to minimize penalty is natural since Zinkevich's algorithm (for OCO without stochastic constraints) minimizes penalty, while the intention to minimize drift is motivated by observing that $g_k^t(\mathbf{x}(t))$ is accumulated into queue $Q_k(t+1)$ introduced in (5.3) such that we intend to have small queue backlogs. The drift $\Delta(t)$ can be complicated and is in general non-convex. The next lemma provides a simple upper bound on $\Delta(t)$ and follows directly from (5.3).

Lemma 5.2. *At each round $t \in \{1, 2, \ldots\}$, Algorithm 5.1 guarantees*

$$\Delta(t) \le \sum_{k=1}^{m} Q_k(t)\big[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1)-\mathbf{x}(t)]\big] + \frac{1}{2}(G + \sqrt{m}D_2R)^2, \tag{5.7}$$

where m is the number of constraint functions; and D_2, G and R are constants defined in Assumption 5.1.

Proof. Recall that for any $b \in \mathbb{R}$, if $a = \max\{b, 0\}$ then $a^2 \leq b^2$. Fix $k \in \{1, 2, \ldots, m\}$. The virtual queue update equation $Q_k(t+1) = \max\left\{Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)], 0\right\}$ implies that

$$\begin{aligned}
\frac{1}{2}[Q_k(t+1)]^2 \leq & \frac{1}{2}\left[Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]\right]^2 \\
= & \frac{1}{2}[Q_k(t)]^2 + Q_k(t)\left[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]\right] \\
& + \frac{1}{2}\left[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]\right]^2 \\
\overset{(a)}{=} & \frac{1}{2}[Q_k(t)]^2 + Q_k(t)\left[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]\right] + \frac{1}{2}[h_k]^2, \qquad (5.8)
\end{aligned}$$

where (a) follows by defining $h_k = g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]$.

Define $\mathbf{s} = [s_1, \ldots, s_m]^{\mathsf{T}}$, where $s_k = [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)], \forall k \in \{1, 2, \ldots, m\}$; and $\mathbf{h} = [h_1, \ldots, h_m]^{\mathsf{T}} = \mathbf{g}^t(\mathbf{x}(t)) + \mathbf{s}$. Then,

$$\|\mathbf{h}\| \overset{(a)}{\leq} \|\mathbf{g}^t(\mathbf{x}(t))\| + \|\mathbf{s}\| \overset{(b)}{\leq} G + \sqrt{\sum_{k=1}^{m} D_2^2 R^2} = G + \sqrt{m} D_2 R, \qquad (5.9)$$

where (a) follows from the triangle inequality; and (b) follows from the definition of Euclidean norm, the Cauchy-Schwartz inequality and Assumption 5.1.

Summing (5.8) over $k \in \{1, 2, \ldots, m\}$ yields

$$\begin{aligned}
& \frac{1}{2}\|\mathbf{Q}(t+1)\|^2 \\
\leq & \frac{1}{2}\|\mathbf{Q}(t)\|^2 + \sum_{k=1}^{m} Q_k(t)\left[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]\right] + \frac{1}{2}\|\mathbf{h}\|^2 \\
\overset{(a)}{\leq} & \frac{1}{2}\|\mathbf{Q}(t)\|^2 + \sum_{k=1}^{m} Q_k(t)\left[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)]\right] + \frac{1}{2}(G + \sqrt{m} D_2 R)^2,
\end{aligned}$$

where (a) follows from (5.9). Rearranging the terms yields the desired result. □

Note that at the end of round t, $\sum_{k=1}^{m} Q_k(t) g_k^t(\mathbf{x}(t)) + \frac{1}{2}(G + \sqrt{m} D_2 R)^2$ is a given constant that is not affected by decision $\mathbf{x}(t+1)$. The algorithm decision in (5.2) is now transparent: $\mathbf{x}(t+1)$ is chosen to minimize the drift-plus-penalty expression (5.6), where $\Delta(t)$ is approximated by the

bound in (5.7).

5.1.3 Preliminary Analysis and More Intuitions of Algorithm 5.1

The next lemma relates constraint violations and virtual queue values and follows directly from (5.3).

Lemma 5.3. *For any $T \geq 1$, Algorithm 5.1 guarantees*

$$\sum_{t=1}^{T} g_k^t(\mathbf{x}(t)) \leq \|\mathbf{Q}(T+1)\| + D_2 \sum_{t=1}^{T} \|\mathbf{x}(t+1) - \mathbf{x}(t)\|, \forall k \in \{1, 2, \ldots, m\},$$

where D_2 is the constant defined in Assumption 5.1.

Proof. Fix $k \in \{1, 2, \ldots, m\}$ and $T \geq 1$. For any $t \in \{0, 1, \ldots\}$, (5.3) in Algorithm 5.1 gives:

$$\begin{aligned}
Q_k(t+1) &= \max\{Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)], 0\} \\
&\geq Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] \\
&\overset{(a)}{\geq} Q_k(t) + g_k^t(\mathbf{x}(t)) - \|\nabla g_k^t(\mathbf{x}(t))\|\|\mathbf{x}(t+1) - \mathbf{x}(t)\| \\
&\overset{(b)}{\geq} Q_k(t) + g_k^t(\mathbf{x}(t)) - D_2\|\mathbf{x}(t+1) - \mathbf{x}(t)\|,
\end{aligned}$$

where (a) follows from the Cauchy-Schwartz inequality and (b) follows from Assumption 5.1. Rearranging terms yields

$$g_k^t(\mathbf{x}(t)) \leq Q_k(t+1) - Q_k(t) + D_2\|\mathbf{x}(t+1) - \mathbf{x}(t)\|.$$

Summing over $t \in \{1, \ldots, T\}$ yields

$$\begin{aligned}
\sum_{t=1}^{T} g_k^t(\mathbf{x}(t)) &\leq Q_k(T+1) - Q_k(1) + D_2 \sum_{t=1}^{T} \|\mathbf{x}(t+1) - \mathbf{x}(t)\| \\
&\overset{(a)}{=} Q_k(T+1) + D_2 \sum_{t=1}^{T} \|\mathbf{x}(t+1) - \mathbf{x}(t)\| \\
&\leq \|\mathbf{Q}(T+1)\| + D_2 \sum_{t=1}^{T} \|\mathbf{x}(t+1) - \mathbf{x}(t)\|.
\end{aligned}$$

where (a) follows from the fact $Q_k(1) = 0$. □

Note that the expression involved in minimization (5.2) in Algorithm 5.1 is strongly convex with respect to $\mathbf{x}$ with modulus 2α and $\mathbf{x}(t+1)$ is chosen to minimize it. Thus, the next lemma follows from Corollary 1.2.

Lemma 5.4. *Let $\mathbf{z} \in \mathcal{X}_0$ be arbitrary. For all $t \geq 1$, Algorithm 5.1 guarantees*

$$\begin{aligned}
&V[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1)-\mathbf{x}(t)] + \sum_{k=1}^m Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1)-\mathbf{x}(t)] + \alpha\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2 \\
\leq &V[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{z}-\mathbf{x}(t)] + \sum_{k=1}^m Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{z}-\mathbf{x}(t)] + \alpha\|\mathbf{z}-\mathbf{x}(t)\|^2 - \alpha\|\mathbf{z}-\mathbf{x}(t+1)\|^2.
\end{aligned}$$

The next corollary follows by taking $\mathbf{z} = \mathbf{x}(t)$ in Lemma 5.4.

Corollary 5.1. *For all $t \geq 1$, Algorithm 5.1 guarantees*

$$\|\mathbf{x}(t+1)-\mathbf{x}(t)\| \leq \frac{VD_1}{2\alpha} + \frac{\sqrt{m}D_2}{2\alpha}\|\mathbf{Q}(t)\|.$$

Proof. Fix $t \geq 1$. Note that $\mathbf{x}(t) \in \mathcal{X}_0$. Taking $\mathbf{z} = \mathbf{x}(t)$ in Lemma 5.4 yields

$$\begin{aligned}
&V[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1)-\mathbf{x}(t)] + \sum_{k=1}^m Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1)-\mathbf{x}(t)] + \alpha\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2 \\
\leq &-\alpha\|\mathbf{x}(t)-\mathbf{x}(t+1)\|^2.
\end{aligned}$$

Rearranging terms and cancelling common terms yields

$$\begin{aligned}
&2\alpha\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2 \\
\leq &-V[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1)-\mathbf{x}(t)] - \sum_{k=1}^m Q_k(t)\big[[\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1)-\mathbf{x}(t)]\big] \\
\overset{(a)}{\leq} &V\|\nabla f^t(\mathbf{x}(t))\|\|\mathbf{x}(t+1)-\mathbf{x}(t)\| + \|\mathbf{Q}(t)\|\sqrt{\sum_{k=1}^m \|\nabla g_k^t(\mathbf{x}(t))\|^2\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2} \\
\overset{(b)}{\leq} &VD_1\|\mathbf{x}(t+1)-\mathbf{x}(t)\| + \sqrt{m}D_2\|\mathbf{Q}(t)\|\|\mathbf{x}(t+1)-\mathbf{x}(t)\|
\end{aligned}$$

where (a) follows by the Cauchy-Schwarz inequality (note that the second term on the right side applies the Cauchy-Schwarz inequality twice); and (b) follows from Assumption 5.1.

Thus, we have

$$\|\mathbf{x}(t+1)-\mathbf{x}(t)\| \leq \frac{VD_1}{2\alpha}+\frac{\sqrt{m}D_2}{2\alpha}\|\mathbf{Q}(t)\|.$$

□

The next corollary follows directly from Lemma 5.3 and Corollary 5.1 and shows that constraint violations are ultimately bounded by sequence $\|\mathbf{Q}(t)\|, t \in \{1,2,\ldots,T+1\}$.

Corollary 5.2. *For any $T \geq 1$, Algorithm 5.1 guarantees*

$$\sum_{t=1}^{T} g_k^t(\mathbf{x}(t)) \leq \|\mathbf{Q}(T+1)\| + \frac{VTD_1D_2}{2\alpha} + \frac{\sqrt{m}D_2^2}{2\alpha}\sum_{t=1}^{T}\|\mathbf{Q}(t)\|, \forall k \in \{1,2,\ldots,m\}$$

where D_1 and D_2 are constants defined in Assumption 5.1.

This corollary further justifies why Algorithm 5.1 intends to minimize drift $\Delta(t)$. As illustrated in the next section, controlled drift can often lead to boundedness of a stochastic process. Thus, the intuition of minimizing drift $\Delta(t)$ is to yield small $\|\mathbf{Q}(t)\|$ bounds.

5.2 Expected Performance Analysis of Algorithm 5.1

This section shows that if we choose $V=\sqrt{T}$ and $\alpha = T$ in Algorithm 5.1, then both expected regret and expected constraint violations are $O(\sqrt{T})$.

5.2.1 A Drift Lemma for Stochastic Processes

Let $\{Z(t), t \geq 0\}$ be a discrete time stochastic process adapted[1] to a filtration $\{\mathcal{F}(t), t \geq 0\}$. For example, $Z(t)$ can be a random walk, a Markov chain or a martingale. The drift analysis is the method of deducing properties, e.g., recurrence, ergodicity, or boundedness, about $Z(t)$ from its drift $\mathbb{E}[Z(t+1)-Z(t)|\mathcal{F}(t)]$. See [Doo53, Haj82] for more discussions or applications on drift analysis. This chapter proposes a new drift analysis lemma for stochastic processes as follows:

Lemma 5.5. *Let $\{Z(t), t \geq 0\}$ be a discrete time stochastic process adapted to a filtration $\{\mathcal{F}(t), t \geq 0\}$ with $Z(0)=0$ and $\mathcal{F}(0)=\{\emptyset,\Omega\}$. Suppose there exists an integer $t_0 > 0$, real*

[1]Random variable Y is said to be adapted to σ-algebra $\mathcal{F}$ if Y is $\mathcal{F}$-measurable. In this case, we often write $Y \in \mathcal{F}$. Similarly, random process $\{Z(t)\}$ is adapted to filtration $\{\mathcal{F}(t)\}$ if $Z(t) \in \mathcal{F}(t), \forall t$. See e.g. [Dur10].

constants $\theta > 0$, $\delta_{\max} > 0$ *and* $0 < \zeta \leq \delta_{\max}$ *such that*

$$|Z(t+1) - Z(t)| \leq \delta_{\max}, \tag{5.10}$$

$$\mathbb{E}[Z(t+t_0) - Z(t)|\mathcal{F}(t)] \leq \begin{cases} t_0\delta_{\max}, & \text{if } Z(t) < \theta \\ -t_0\zeta, & \text{if } Z(t) \geq \theta \end{cases}. \tag{5.11}$$

hold for all $t \in \{1, 2, \ldots\}$. *Then, the following holds*

1. $\mathbb{E}[Z(t)] \leq \theta + t_0\delta_{\max} + t_0\frac{4\delta_{\max}^2}{\zeta}\log\left(\frac{8\delta_{\max}^2}{\zeta^2}\right), \forall t \in \{1, 2, \ldots\}$.
2. *For any constant* $0 < \mu < 1$, *we have* $Pr(Z(t) \geq z) \leq \mu, \forall t \in \{1, 2, \ldots\}$ *where* $z = \theta + t_0\delta_{\max} + t_0\frac{4\delta_{\max}^2}{\zeta}\log\left(\frac{8\delta_{\max}^2}{\zeta^2}\right) + t_0\frac{4\delta_{\max}^2}{\zeta}\log(\frac{1}{\mu})$.

Proof. See Section 5.5.1. □

The above lemma provides both expected and high probability bounds for stochastic processes based on a drift condition. It will be used to establish upper bounds of virtual queues $\|\mathbf{Q}(t)\|$, which further leads to expected and high probability constraint performance bounds of our algorithm. For a given stochastic process $Z(t)$, it is possible to show the drift condition (5.11) holds for multiple t_0 with different ζ and θ. In fact, we will show in Lemma 5.7 that $\|\mathbf{Q}(t)\|$ yielded by Algorithm 5.1 satisfies (5.11) for any integer $t_0 > 0$ by selecting ζ and θ according to t_0. One-step drift conditions, corresponding to the special case $t_0 = 1$ of Lemma 5.5, have been previously considered in [Haj82, Nee15]. However, Lemma 5.5 (with general $t_0 > 0$) allows us to choose the best t_0 in performance analysis such that sublinear regret and constraint violation bounds are possible.

5.2.2 Expected Constraint Violation Analysis

Define filtration $\{\mathcal{W}(t), t \geq 0\}$ with $\mathcal{W}(0) = \{\emptyset, \Omega\}$ and $\mathcal{W}(t) = \sigma(\omega(1), \ldots, \omega(t))$ being the σ-algebra generated by random samples $\{\omega(1), \ldots, \omega(t)\}$ up to round t. From the update rule in Algorithm 5.1, we observe that $\mathbf{x}(t+1)$ is a deterministic function of $f^t(\cdot), \mathbf{g}(\cdot; \omega(t))$ and $\mathbf{Q}(t)$ where $\mathbf{Q}(t)$ is further a deterministic function of $\mathbf{Q}(t-1), \mathbf{g}(\cdot; \omega(t-1))$, $\mathbf{x}(t)$ and $\mathbf{x}(t-1)$. By inductions, it is easy to show that $\sigma(\mathbf{x}(t)) \subseteq \mathcal{W}(t-1)$ and $\sigma(\mathbf{Q}(t)) \subseteq \mathcal{W}(t-1)$ for all $t \geq 1$ where $\sigma(Y)$ denotes the σ-algebra generated by random variable Y. For fixed $t \geq 1$, since $\mathbf{Q}(t)$

is fully determined by $\omega(\tau), \tau \in \{1, 2, \ldots, t-1\}$ and $\omega(t)$ are i.i.d., we know $\mathbf{g}^t(\mathbf{x})$ is independent of $\mathbf{Q}(t)$. This is formally summarized in the next lemma.

Lemma 5.6. *If $\mathbf{x}^* \in \mathcal{X}_0$ satisfies $\tilde{\mathbf{g}}(\mathbf{x}^*) = \mathbb{E}_\omega[\mathbf{g}(\mathbf{x}^*; \omega)] \leq \mathbf{0}$, then Algorithm 5.1 guarantees:*

$$\mathbb{E}[Q_k(t) g_k^t(\mathbf{x}^*)] \leq 0, \forall k \in \{1, 2, \ldots, m\}, \forall t \geq 1. \tag{5.12}$$

Proof. Fix $k \in \{1, 2, \ldots, m\}$ and $t \geq 1$. Since $g_k^t(\mathbf{x}^*) = g_k(\mathbf{x}^*; \omega(t))$ is independent of $Q_k(t)$, which is determined by $\{\omega(1), \ldots, \omega(t-1)\}$, it follows that

$$\mathbb{E}[Q_k(t) g_k^t(\mathbf{x}^*)] = \mathbb{E}[Q_k(t)] \mathbb{E}[g_k^t(\mathbf{x}^*)] \overset{(a)}{\leq} 0$$

where (a) follows from the fact that $\mathbb{E}[g_k^t(\mathbf{x}^*)] \leq 0$ and $Q_k(t) \geq 0$. □

To establish a bound on constraint violations, by Corollary 5.2, it suffices to derive upper bounds for $\|\mathbf{Q}(t)\|$. In this subsection, we derive upper bounds for $\|\mathbf{Q}(t)\|$ by applying the new drift lemma (Lemma 5.5) developed at the beginning of this section. The next lemma shows that random process $Z(t) = \|\mathbf{Q}(t)\|$ satisfies the conditions in Lemma 5.5.

Lemma 5.7. *Let $t_0 > 0$ be an arbitrary integer. At each round $t \in \{1, 2, \ldots, \}$ in Algorithm 5.1, the following holds*

$$\big|\|\mathbf{Q}(t+1)\| - \|\mathbf{Q}(t)\|\big| \leq G + \sqrt{m} D_2 R, \quad \textit{and}$$

$$\mathbb{E}[\|\mathbf{Q}(t+t_0)\| - \|\mathbf{Q}(t)\| \big| \mathcal{W}(t-1)] \leq \begin{cases} t_0(G + \sqrt{m} D_2 R), & \textit{if } \|\mathbf{Q}(t)\| < \theta \\ -t_0 \frac{\epsilon}{2}, & \textit{if } \|\mathbf{Q}(t)\| \geq \theta \end{cases},$$

where $\theta = \frac{\epsilon}{2} t_0 + (G + \sqrt{m} D_2 R) t_0 + \frac{2\alpha R^2}{t_0 \epsilon} + \frac{2V D_1 R + (G + \sqrt{m} D_2 R)^2}{\epsilon}$, m is the number of constraint functions; D_1, D_2, G and R are constants defined in Assumption 5.1; and ϵ is the constant defined in Assumption 5.2. (Note that $\epsilon < G$ by the definition of G.)

Proof. See Section 5.5.2. □

Lemma 5.7 allows us to apply Lemma 5.5 to random process $Z(t) = \|\mathbf{Q}(t)\|$ and obtain $\mathbb{E}[\|\mathbf{Q}(t)\|] = O(\sqrt{T}), \forall t$ by taking $t_0 = \lceil \sqrt{T} \rceil$, $V = \sqrt{T}$ and $\alpha = T$, where $\lceil \sqrt{T} \rceil$ represents the smallest integer no less than $\sqrt{T}$. By Corollary 5.2, this further implies the expected constraint violation bound $\mathbb{E}[\sum_{t=1}^T g_k(\mathbf{x}(t))] \leq O(\sqrt{T})$ as summarized in the next theorem.

Theorem 5.1 (Expected Constraint Violation Bound)**.** *If $V = \sqrt{T}$ and $\alpha = T$ in Algorithm 5.1, then for all $T \geq 1$, we have*

$$\mathbb{E}[\sum_{t=1}^{T} g_k^t(\mathbf{x}(t))] \leq O(\sqrt{T}), \forall k \in \{1, 2, \ldots, m\}. \tag{5.13}$$

where the expectation is taken with respect to all $\omega(t)$.

Proof. Define random process $Z(t)$ with $Z(0) = 0$ and $Z(t) = \|\mathbf{Q}(t)\|, t \geq 1$ and filtration $\mathcal{F}(t)$ with $\mathcal{F}(0) = \{\emptyset, \Omega\}$ and $\mathcal{F}(t) = \mathcal{W}(t-1), t \geq 1$. Note that $Z(t)$ is adapted to $\mathcal{F}(t)$. By Lemma 5.7, $Z(t)$ satisfies the conditions in Lemma 5.5 with $\delta_{\max} = G + \sqrt{m}D_2R$, $\zeta = \frac{\epsilon}{2}$ and $\theta = \frac{\epsilon}{2}t_0 + (G + \sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R + (G+\sqrt{m}D_2R)^2}{\epsilon}$. Thus, by part 1 of Lemma 5.5, for all $t \in \{1, 2, \ldots\}$, we have $\mathbb{E}[\|\mathbf{Q}(t)\|] \leq \frac{\epsilon}{2}t_0 + 2(G+\sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R+(G+\sqrt{m}D_2R)^2}{\epsilon} + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log\big(\frac{32(G+\sqrt{m}D_2R)^2}{\epsilon^2}\big)$. Taking $t_0 = \lceil\sqrt{T}\rceil$, $V = \sqrt{T}$ and $\alpha = T$, we have $\mathbb{E}[\|\mathbf{Q}(t)\|] \leq O(\sqrt{T})$ for all $t \in \{1, 2, \ldots\}$.

Fix $T \geq 1$. By Corollary 5.2 (with $V = \sqrt{T}$ and $\alpha = T$) , we have $\sum_{t=1}^{T} g_k^t(\mathbf{x}(t)) \leq \|\mathbf{Q}(T+1)\| + \frac{\sqrt{T}D_1D_2}{2} + \frac{\sqrt{m}D_2^2}{2T}\sum_{t=1}^{T}\|\mathbf{Q}(t)\|, \forall k \in \{1, 2, \ldots, m\}$. Taking expectations on both sides and substituting $\mathbb{E}[\|\mathbf{Q}(t)\|] = O(\sqrt{T}), \forall t$ into it yields $\mathbb{E}[\sum_{t=1}^{T} g_k^t(\mathbf{x}(t))] \leq O(\sqrt{T})$. □

5.2.3 Expected Regret Analysis

The next lemma refines Lemma 5.4 and is useful to analyze the regret.

Lemma 5.8. *Let $\mathbf{z} \in \mathcal{X}_0$ be arbitrary. For all $T \geq 1$, Algorithm 5.1 guarantees*

$$\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{z}) + \underbrace{\frac{\alpha}{V}R^2 + \frac{VD_1^2}{4\alpha}T + \frac{1}{2}(G+\sqrt{m}D_2R)^2\frac{T}{V}}_{(I)} + \underbrace{\frac{1}{V}\sum_{t=1}^{T}\big[\sum_{k=1}^{m} Q_k(t)g_k^t(\mathbf{z})\big]}_{(II)} \tag{5.14}$$

where m is the number of constraint functions; and D_1, D_2, G and R are constants defined in Assumption 5.1.

Proof. See Section 5.5.3. □

Note that if we take $V = \sqrt{T}$ and $\alpha = T$, then term (I) in (5.14) is $O(\sqrt{T})$. Recall that the expectation of term (II) in (5.14) with $\mathbf{z} = \mathbf{x}^*$ is non-positive by Lemma 5.6. The expected regret

bound of Algorithm 5.1 follows by taking expectations on both sides of (5.14) and is summarized in the next theorem.

Theorem 5.2 (Expected Regret Bound)**.** *Let $\mathbf{x}^* \in \mathcal{X}_0$ be any fixed solution that satisfies $\tilde{\mathbf{g}}(\mathbf{x}^*) \leq \mathbf{0}$, e.g., $\mathbf{x}^* = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\{\sum_{t=1}^{T} f^t(\mathbf{x})\}$. If $V = \sqrt{T}$ and $\alpha = T$ in Algorithm 5.1, then for all $T \geq 1$,*

$$\mathbb{E}[\sum_{t=1}^{T} f^t(\mathbf{x}(t))] \leq \mathbb{E}[\sum_{t=1}^{T} f^t(\mathbf{x}^*)] + O(\sqrt{T}).$$

where the expectation is taken with respect to all $\omega(t)$.

Proof. Fix $T \geq 1$. Taking $\mathbf{z} = \mathbf{x}^*$ in Lemma 5.8 yields $\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{x}^*) + \frac{\alpha}{V}R^2 + \frac{VD_1^2}{4\alpha}T + \frac{1}{2}(G + \sqrt{m}D_2R)^2\frac{T}{V} + \frac{1}{V}\sum_{t=1}^{T}\big[\sum_{k=1}^{m} Q_k(t)g_k^t(\mathbf{x}^*)\big]$. Taking expectations on both sides and using (5.12) yields $\sum_{t=1}^{T}\mathbb{E}[f^t(\mathbf{x}(t))] \leq \sum_{t=1}^{T}\mathbb{E}[f^t(\mathbf{x}^*)] + R^2\frac{\alpha}{V} + \frac{D_1^2}{4}\frac{V}{\alpha}T + \frac{1}{2}(G+\sqrt{m}D_2R)^2\frac{T}{V}$. Taking $V = \sqrt{T}$ and $\alpha = T$ yields $\sum_{t=1}^{T}\mathbb{E}[f^t(\mathbf{x}(t))] \leq \sum_{t=1}^{T}\mathbb{E}[f^t(\mathbf{x}^*)] + O(\sqrt{T})$. □

5.2.4 Special Case Performance Guarantees

Theorems 5.1 and 5.2 provide expected performance guarantees of Algorithm 5.1 for OCO with stochastic constraints. The results further imply the performance guarantees in the following special cases:

- **OCO with long term constraints**: In this case, $g_k(\mathbf{x};\omega(t)) \equiv g_k(\mathbf{x})$ and there is no randomness. Thus, the expectations in Theorems 5.1 and 5.2 disappear. For this problem, Algorithm 5.1 can achieve $O(\sqrt{T})$ (deterministic) regret and $O(\sqrt{T})$ (deterministic) constraint violations.

- **Stochastic constrained convex optimization**: Note that i.i.d. time-varying $f(\mathbf{x};\omega(t))$ is a special case of arbitrarily-varying $f^t(\mathbf{x})$ as considered in our OCO setting. Thus, Theorems 5.1 and 5.2 still hold when Algorithm 5.1 is applied to solve stochastic constrained convex optimization $\min_{\mathbf{x}}\{\mathbb{E}[f(\mathbf{x};\omega)] : \mathbb{E}[g_k(\mathbf{x};\omega)] \leq 0, \forall k \in \{1,2,\ldots,m\}, \mathbf{x} \in \mathcal{X}_0\}$ in an online fashion with i.i.d. realizations $\omega(t) \sim \omega$. Since Algorithm 5.1 chooses each $\mathbf{x}(t)$ without knowing $\omega(t)$, it follows that $\mathbf{x}(t)$ is independent of $\omega(t')$ for any $t' \geq t$ by the i.i.d. property of each $\omega(t)$. Fix $T > 0$, if we run Algorithm 5.1 for T slots and use

$\overline{\mathbf{x}}(T) = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}(t)$ as a fixed solution for any future slot $t' \geq T+1$, then

$$\begin{aligned}\mathbb{E}[f(\overline{\mathbf{x}}(T);\omega(t'))] &\overset{(a)}{\leq} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[f(\mathbf{x}(t);\omega(t'))] \\ &\overset{(b)}{=} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[f(\mathbf{x}(t);\omega(t))] \\ &\overset{(c)}{\leq} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[f(\mathbf{x}^*;\omega(t))] + O(\frac{1}{\sqrt{T}}) \\ &\overset{(d)}{=} \mathbb{E}[f(\mathbf{x}^*;\omega(t'))] + O(\frac{1}{\sqrt{T}})\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}[g_k(\overline{\mathbf{x}}(T);\omega(t'))] &\overset{(a)}{\leq} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[g_k(\overline{\mathbf{x}}(T);\omega(t'))] \\ &\overset{(b)}{=} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[g_k(\mathbf{x}(t);\omega(t))] \\ &\overset{(c)}{\leq} O(\frac{1}{\sqrt{T}}), \forall k \in \{1,2,\ldots,m\}\end{aligned}$$

where in both inequality chains (a) follows from Jensen's inequality and the fact that $\overline{\mathbf{x}}(T)$ is independent of $\omega(t')$; (b) follows because each $\mathbf{x}(t)$ is independent of both $\omega(t)$ and $\omega(t')$, and $\omega(t)$ and $\omega(t')$ are i.i.d. realizations of ω; (c) follows from Theorems 5.1 and 5.2 by dividing both sides by T and (d) follows because $\mathbb{E}[f(\mathbf{x}^*;\omega(t))] = \mathbb{E}[f(\mathbf{x}^*;\omega(t'))]$ for all $t \in \{1,\ldots,T\}$ by the i.i.d. property of each $\omega(t)$. Thus, if we use Algorithm 5.1 as a (batch) offline algorithm to solve stochastic constrained convex optimization, it has $O(1/\sqrt{T})$ convergence and ties with the algorithm developed in [LZ16], which is by design a (batch) offline algorithm and can only solve stochastic optimization with a single constraint function.

- **Deterministic constrained convex optimization**: Similarly to OCO with long term constraints, the expectations in Theorems 5.1 and 5.2 disappear in this case since $f^t(\mathbf{x}) \equiv f(\mathbf{x})$ and $g_k(\mathbf{x};\omega(t)) \equiv g_k(\mathbf{x})$. If we use $\overline{\mathbf{x}}(T) = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}(t)$ as the solution, then $f(\overline{\mathbf{x}}(T)) \leq f(\mathbf{x}^*) + O(\frac{1}{\sqrt{T}})$ and $g_k(\overline{\mathbf{x}}(T)) \leq O(\frac{1}{\sqrt{T}})$, which follows by dividing inequalities in Theorems 5.1 and 5.2 by T on both sides and applying Jensen's inequality. Thus, Algorithm 5.1 solves deterministic constrained convex optimization with $O(\frac{1}{\sqrt{T}})$ convergence.

5.3 High Probability Performance Analysis

This section shows that if we choose $V = \sqrt{T}$ and $\alpha = T$ in Algorithm 5.1, then for any $0 < \lambda < 1$, with probability at least $1 - \lambda$, regret is $O(\sqrt{T}\log(T)\log^{1.5}(\frac{1}{\lambda}))$ and constraint violations are $O\big(\sqrt{T}\log(T)\log(\frac{1}{\lambda})\big)$.

5.3.1 High Probability Constraint Violation Analysis

Similarly to the expected constraint violation analysis, we can use part 2 of the new drift lemma (Lemma 5.5) to obtain a high probability bound of $\|\mathbf{Q}(t)\|$, which together with Corollary 5.2 leads to a high probability constraint violation bound summarized in Theorem 5.3.

Theorem 5.3 (High Probability Constraint Violation Bound)**.** *Let $0 < \lambda < 1$ be arbitrary. If $V = \sqrt{T}$ and $\alpha = T$ in Algorithm 5.1, then for all $T \geq 1$ and all $k \in \{1, 2, \ldots, m\}$, we have*

$$Pr\Big(\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq O\big(\sqrt{T}\log(T)\log(\frac{1}{\lambda})\big)\Big) \geq 1 - \lambda.$$

Proof. See Section 5.5.4. □

5.3.2 High Probability Regret Analysis

To obtain a high probability regret bound from Lemma 5.8, it remains to derive a high probability bound of term (II) in (5.14) with $\mathbf{z} = \mathbf{x}^*$. The main challenge is that term (II) is a supermartingale with unbounded differences (due to the possibly unbounded virtual queues $Q_k(t)$). Most concentration inequalities, e.g., the Hoeffding-Azuma inequality, used in high probability performance analysis of online algorithms are restricted to martingales/supermartingales with bounded differences. See for example [CBL06, BDH$^+$08, MJY12]. The following lemma considers supermartingales with unbounded differences. Its proof uses the truncation method to construct an auxiliary well-behaved supermartingale. Similar proof techniques are previously used in [Vu02, TV15] to prove different concentration inequalities for supermartingales/martingales with unbounded differences. The truncation method is also previously used in [WYN15] to analyze the high probability sample path performance of the conventional drift-plus-penalty technique for opportunistic stochastic optimzation.

Lemma 5.9. *Let $\{Z(t), t \geq 0\}$ be a supermartingale adapted to a filtration $\{\mathcal{F}(t), t \geq 0\}$ with $Z(0) = 0$ and $\mathcal{F}(0) = \{\emptyset, \Omega\}$, i.e., $\mathbb{E}[Z(t+1)|\mathcal{F}(t)] \leq Z(t), \forall t \geq 0$. Suppose there exits a constant $c > 0$ such that $\{|Z(t+1) - Z(t)| > c\} \subseteq \{Y(t) > 0\}, \forall t \geq 0$, where $Y(t)$ is process with $Y(t)$ adapted to $\mathcal{F}(t)$ for all $t \geq 0$. Then, for all $z > 0$, we have*

$$Pr(Z(t) \geq z) \leq e^{-z^2/(2tc^2)} + \sum_{\tau=0}^{t-1} Pr(Y(\tau) > 0), \forall t \geq 1.$$

Note that if $\Pr(Y(t) > 0) = 0, \forall t \geq 0$, then $\Pr(\{|Z(t+1) - Z(t)| > c\}) = 0, \forall t \geq 0$ and $Z(t)$ is a supermartingale with differences bounded by c. In this case, Lemma 5.9 reduces to the conventional Hoeffding-Azuma inequality.

The next theorem summarizes the high probability regret performance of Algorithm 5.1 and follows from Lemmas 5.5-5.9 .

Theorem 5.4 (High Probability Regret Bound)**.** *Let $\mathbf{x}^* \in \mathcal{X}_0$ be any fixed solution that satisfies $\tilde{\mathbf{g}}(\mathbf{x}^*) \leq \mathbf{0}$, e.g., $\mathbf{x}^* = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\{\sum_{t=1}^{T} f^t(\mathbf{x})\}$. Let $0 < \lambda < 1$ be arbitrary. If $V = \sqrt{T}$ and $\alpha = T$ in Algorithm 5.1, then for all $T \geq 1$, we have*

$$Pr\Big(\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{x}^*) + O(\sqrt{T}\log(T)\log^{1.5}(\frac{1}{\lambda}))\Big) \geq 1 - \lambda.$$

Proof. See Section 5.5.6. □

5.4 Chapter Summary

This chapter studies OCO with stochastic constraints, where the objective function varies arbitrarily but the constraint functions are i.i.d. over time. A novel learning algorithm is developed that guarantees $O(\sqrt{T})$ expected regret and constraint violations and $O(\sqrt{T}\log(T))$ high probability regret and constraint violations.

5.5 Supplement to this Chapter

5.5.1 Proof of Lemma 5.5

In this proof, we first establish an upper bound of $\mathbb{E}[e^{rZ(t)}]$ for some constant $r > 0$. Part 1 of this lemma follows by applying Jensen's inequality since e^{rx} is convex with respect to x when $r > 0$. Part 2 of this lemma follows directly from Markov's inequality.

The following fact is useful in the proof.

Fact 5.1. *$e^x \leq 1 + x + 2x^2$ for any $|x| \leq 1$.*

Proof. By Taylor's expansion, we known for any $x \in \mathbb{R}$, there exists a point $\hat{x}$ in between 0 and x such that $e^x = 1 + x + e^{\hat{x}} \frac{x^2}{2}$. (Note that the value of $\hat{x}$ depends on x and if $x > 0$, then $\hat{x} \in (0, x)$; if $x < 0$, then $\hat{x} \in (x, 0)$; and if $x = 0$, then $\hat{x} = x$.) Since $|x| \leq 1$, we have $e^{\hat{x}} \leq e \leq 4$. Thus, $e^x \leq 1 + x + 2x^2$ for any $|x| \leq 1$. □

The next lemma provides an upper bound of $\mathbb{E}[e^{rZ(t)}]$ with constant $r = \frac{\zeta}{4t_0\delta_{\max}^2} < 1$.

Lemma 5.10. *Under the assumption of Lemma 5.5, we have*

$$\mathbb{E}[e^{rZ(t)}] \leq \frac{e^{rt_0\delta_{\max}}}{1-\rho} e^{r\theta}, \forall t \in \{0, 1, \ldots\},$$

where $r = \frac{\zeta}{4t_0\delta_{\max}^2}$, $\rho = 1 - \frac{\zeta^2}{8\delta_{\max}^2} = 1 - \frac{rt_0\zeta}{2}$.

Proof. Since $0 < \zeta < \delta_{\max}$, we have $0 < \rho < 1 < e^{r\delta_{\max}}$. Define $\eta(t) = Z(t + t_0) - Z(t)$. Note that $|\eta(t)| \leq t_0\delta_{\max}, \forall t \geq 0$ and $|r\eta(t)| \leq \frac{\zeta}{4t_0\delta_{\max}^2} t_0\delta_{\max} = \frac{\zeta}{4\delta_{\max}} \leq 1$. Then,

$$\begin{aligned} e^{rZ(t+t_0)} =& e^{rZ(t)} e^{r\eta(t)} && (5.15) \\ \overset{(a)}{\leq}& e^{rZ(t)}(1 + r\eta(t) + 2r^2 t_0^2 \delta_{\max}^2) \\ \overset{(b)}{=}& e^{rZ(t)}(1 + r\eta(t) + \frac{1}{2} rt_0\zeta), && (5.16) \end{aligned}$$

where (a) follows from Fact 5.1 by noting that $|r\eta(t)| \leq 1$ and $|\eta(t)| \leq t_0\delta_{\max}$; and (b) follows by substituting $r = \frac{\zeta}{4t_0\delta_{\max}^2}$ into a single r of the term $2r^2 t_0^2 \delta_{\max}^2$.

Next, consider the cases $Z(t) \geq \theta$ and $Z(t) < \theta$, separately.

- Case $Z(t) \geq \theta$: Taking conditional expectations on both sides of (5.16) yields:

$$\begin{aligned}\mathbb{E}[e^{rZ(t+t_0)}|Z(t)] \leq & \mathbb{E}[e^{rZ(t)}(1 + r\eta(t) + \frac{1}{2}rt_0\zeta)|Z(t)] \\ \overset{(a)}{\leq} & e^{rZ(t)}\big(1 - rt_0\zeta + \frac{1}{2}rt_0\zeta\big) \\ = & e^{rZ(t)}\big(1 - \frac{rt_0\zeta}{2}\big) \\ \overset{(b)}{=} & \rho e^{rZ(t)}.\end{aligned}$$

  where (a) follows from the fact that $\mathbb{E}[Z(t+t_0) - Z(t)|\mathcal{F}(t)] \leq -t_0\zeta$ when $Z(t) \geq \theta$; and (b) follows from the fact that $\rho = 1 - \frac{rt_0\zeta}{2}$.

- Case $Z(t) < \theta$: Taking conditional expectations on both sides of (5.15) yields:

$$\begin{aligned}\mathbb{E}[e^{rZ(t+t_0)}|Z(t)] = & \mathbb{E}[e^{rZ(t)}e^{r\eta(t)}|Z(t)] \\ = & e^{rZ(t)}\mathbb{E}[e^{r\eta(t)}|Z(t)] \\ \overset{(a)}{\leq} & e^{rt_0\delta_{\max}}e^{rZ(t)},\end{aligned}$$

  where (a) follows from the fact that $\eta(t) \leq t_0\delta_{\max}$.

Putting two cases together yields:

$$\begin{aligned}\mathbb{E}[e^{rZ(t+t_0)}] \overset{(a)}{=} & \Pr(Z(t) \geq \theta)\mathbb{E}[e^{rZ(t+t_0)}|Z(t) \geq \theta] + \Pr(Z(t) < \theta)\mathbb{E}[e^{rZ(t+t_0)}|Z(t) < \theta] \\ \overset{(b)}{\leq} & \rho\mathbb{E}[e^{rZ(t)}|Z(t) \geq \theta]\Pr(Z(t) \geq \theta) + e^{rt_0\delta_{\max}}\mathbb{E}[e^{rZ(t)}|Z(t) < \theta]\Pr(Z(t) < \theta) \\ \overset{(c)}{=} & \rho\mathbb{E}[e^{rZ(t)}] + (e^{rt_0\delta_{\max}} - \rho)\mathbb{E}[e^{rZ(t)}|Z(t) < \theta]\Pr(Z(t) < \theta) \\ \overset{(d)}{\leq} & \rho\mathbb{E}[e^{rZ(t)}] + (e^{rt_0\delta_{\max}} - \rho)e^{r\theta} \\ \leq & \rho\mathbb{E}[e^{rZ(t)}] + e^{rt_0\delta_{\max}}e^{r\theta}, \end{aligned} \tag{5.17}$$

where (a) follows by the definition of expectations; (b) follows from the results in the above two cases; (c) follows from the fact that $\mathbb{E}[e^{rZ(t)}] = \Pr(Z(t) \geq \theta)\mathbb{E}[e^{rZ(t)}|Z(t) \geq \theta] + \Pr(Z(t) < \theta)\mathbb{E}[e^{rZ(t)}|Z(t) < \theta]$; and (d) follow from the fact that $e^{rt_0\delta_{\max}} > \rho$.

Now, we prove $\mathbb{E}[e^{rZ(t)}] \leq \frac{e^{rt_0\delta_{\max}}}{1-\rho}e^{r\theta}, \forall t \geq 0$, by inductions.

We first consider the base case $t \in \{0, 1, \ldots, t_0\}$. Since $Z(t) \leq t\delta_{\max}, \forall t \geq 0$, it follows that

$\mathbb{E}[e^{rZ(t)}] \leq e^{rt\delta_{\max}} \leq e^{rt_0\delta_{\max}} \leq \frac{e^{rt_0\delta_{\max}}}{1-\rho} e^{r\theta}, \forall t \in \{0, 1, \ldots, t_0\}$, where the last inequality follows because $\frac{e^{r\theta}}{1-\rho} \geq 1$.

Now assume that $\mathbb{E}[e^{rZ(t)}] \leq \frac{e^{rt_0\delta_{\max}}}{1-\rho} e^{r\theta}$ for all $t \in \{0, 1, \ldots, \tau\}$ with some $\tau \geq t_0$ and consider iteration $t = \tau + 1$. By (5.17), we have

$$\begin{aligned}
\mathbb{E}[e^{rZ(\tau+1)}] \leq & \rho \mathbb{E}[e^{rZ(\tau+1-t_0)}] + e^{rt_0\delta_{\max}} e^{r\theta} \\
\overset{(a)}{\leq} & \rho \frac{e^{rt_0\delta_{\max}}}{1-\rho} e^{r\theta} + e^{rt_0\delta_{\max}} e^{r\theta} \\
= & \frac{e^{rt_0\delta_{\max}}}{1-\rho} e^{r\theta}
\end{aligned}$$

where (a) follows from the induction hypothesis by noting that $0 \leq \tau + 1 - t_0 \leq \tau$.

Thus, this lemma follows by inductions. □

By this lemma, for all $t \in \{0, 1, \ldots\}$, we have

$$\mathbb{E}[e^{rZ(t)}] \leq \frac{e^{rt_0\delta_{\max}}}{1-\rho} e^{r\theta}. \tag{5.18}$$

Proof of Part 1: Note that e^{rx} is convex with respect to x when $r > 0$. By Jensen's inequality,

$$e^{r\mathbb{E}[Z(t)]} \leq \mathbb{E}[e^{rZ(t)}] \overset{(a)}{\leq} \frac{e^{r(\theta+t_0\delta_{\max})}}{1-\rho}, \tag{5.19}$$

where (a) follows from (5.18).

Taking logarithm on both sides and dividing by r yields:

$$\begin{aligned}
\mathbb{E}[Z(t)] \leq & \theta + t_0\delta_{\max} + \frac{1}{r} \log\big(\frac{1}{1-\rho}\big) \\
\overset{(a)}{=} & \theta + t_0\delta_{\max} + t_0 \frac{4\delta_{\max}^2}{\zeta} \log\big(\frac{8\delta_{\max}^2}{\zeta^2}\big),
\end{aligned}$$

where (a) follows by recalling that $r = \frac{\zeta}{4t_0\delta_{\max}^2}$ and $\rho = 1 - \frac{\zeta^2}{8\delta_{\max}^2}$.

Proof of Part 2: Fix z. Note that

$$\begin{aligned}\Pr(Z(t) \geq z) =& \Pr(e^{rZ(t)} \geq e^{rz}) \\ \overset{(a)}{\leq}& \frac{\mathbb{E}[e^{rZ(t)}]}{e^{rz}} \\ \overset{(b)}{\leq}& e^{r(\theta - z + t_0\delta_{\max})} \frac{1}{1-\rho} \\ \overset{(c)}{=}& e^{\frac{\zeta}{4t_0\delta_{\max}^2}(\theta - z + t_0\delta_{\max})} \big(\frac{8\delta_{\max}^2}{\zeta^2}\big) \end{aligned} \tag{5.20}$$

where (a) follows from Markov's inequality; (b) follows from (5.18); and (c) follows by recalling that $r = \frac{\zeta}{4t_0\delta_{\max}^2}$ and $\rho = 1 - \frac{\zeta^2}{8\delta_{\max}^2}$.

Define $\mu = e^{\frac{\zeta}{4t_0\delta_{\max}^2}(\theta - z + t_0\delta_{\max})}\big(\frac{8\delta_{\max}^2}{\zeta^2}\big)$. It follows that if

$$z = \theta + t_0\delta_{\max} + t_0\frac{4\delta_{\max}^2}{\zeta}\log\big(\frac{8\delta_{\max}^2}{\zeta^2}\big) + t_0\frac{4\delta_{\max}^2}{\zeta}\log(\frac{1}{\mu}),$$

then we have $\Pr(Z(t) \geq z) \leq \mu$ by (5.20).

5.5.2 Proof of Lemma 5.7

The next lemma will be useful in our proof.

Lemma 5.11. *Let $\hat{\mathbf{x}} \in \mathcal{X}_0$ be a Slater point defined in Assumption 5.2, i.e, $\tilde{g}_k(\hat{\mathbf{x}}) = \mathbb{E}_\omega[g_k(\hat{\mathbf{x}};\omega)] \leq -\epsilon, \forall k \in \{1,2,\ldots,m\}$. Then*

$$\mathbb{E}[\sum_{k=1}^{m} Q_k(t_1)g_k^{t_1}(\hat{\mathbf{x}})|\mathcal{W}(t_2)] \leq -\epsilon\mathbb{E}[\|\mathbf{Q}(t_1)\||\mathcal{W}(t_2)], \quad \forall t_2 \leq t_1 - 1$$

where $\epsilon > 0$ is defined in Assumption 5.2.

Proof. To prove this lemma, we first show that

$$\mathbb{E}[Q_k(t_1)g_k^{t_1}(\hat{\mathbf{x}})|\mathcal{W}(t_2)] \leq -\epsilon\mathbb{E}[Q_k(t_1)|\mathcal{W}(t_2)], \forall k \in \{1,2,\ldots,m\}, \forall t_2 \leq t_1 - 1.$$

Fix $k \in \{1,2,\ldots,m\}$. Note that $\mathbf{Q}(t_1) \in \mathcal{W}(t_1 - 1)$ and $g_k^{t_1}(\hat{\mathbf{x}})$ is independent of $\mathcal{W}(t_1 - 1)$.

Further, if $t_2 \leq t_1 - 1$, then $\mathcal{W}(t_2) \subseteq \mathcal{W}(t_1 - 1)$. Thus, we have

$$\begin{aligned}\mathbb{E}[Q_k(t_1)g_k^{t_1}(\hat{\mathbf{x}})|\mathcal{W}(t_2)] &\overset{(a)}{=} \mathbb{E}\big[\mathbb{E}[Q_k(t_1)g_k^{t_1}(\hat{\mathbf{x}})|\mathcal{W}(t_1-1)]|\mathcal{W}(t_2)\big] \\ &\overset{(b)}{=} \mathbb{E}\big[Q_k(t_1)\mathbb{E}[g_k^{t_1}(\hat{\mathbf{x}})]|\mathcal{W}(t_2)\big] \\ &\overset{(c)}{=} \mathbb{E}[g_k^{t_1}(\hat{\mathbf{x}})]\mathbb{E}[Q_k(t_1)|\mathcal{W}(t_2)] \\ &\overset{(d)}{\leq} -\epsilon\mathbb{E}[Q_k(t_1)|\mathcal{W}(t_2)]\end{aligned}$$

where (a) follows from iterated expectations; (b) follows because $g_k^{t_1}(\hat{\mathbf{x}})$ is independent of $\mathcal{W}(t_1-1)$ and $Q_k(t_1) \in \mathcal{W}(t_1-1)$; (c) follows by extracting the constant $\mathbb{E}[g_k^{t_1}(\hat{\mathbf{x}})]$ and (d) follows from the assumption that $\hat{\mathbf{x}}$ is a Slater point, $g^t(\cdot)$ are i.i.d. across t and the fact that $Q_k(t) \geq 0$.

Now, summing over $m \in \{1, 2, \ldots, m\}$ yields

$$\begin{aligned}\mathbb{E}[\sum_{k=1}^{m} Q_k(t_1)g_k^{t_1}(\hat{\mathbf{x}})|\mathcal{W}(t_2)] &\leq -\epsilon\mathbb{E}[\sum_{k=1}^{m} Q_k(t_1)|\mathcal{W}(t_2)] \\ &\overset{(a)}{\leq} -\epsilon\mathbb{E}[\|\mathbf{Q}(t_1)\||\mathcal{W}(t_2)]\end{aligned}$$

where (a) follows from the basic fact that $\sum_{k=1}^{m} a_k \geq \sqrt{\sum_{k=1}^{m} a_k^2}$ when $a_k \geq 0, \forall k \in \{1, 2, \ldots, m\}$.

□

The bounded difference of $|\mathbf{Q}(t+1) - \mathbf{Q}(t)|$ follows directly from the virtual queue update equation (5.3) and is summarized in the next Lemma.

Lemma 5.12. *Let $\mathbf{Q}(t), t \in \{0, 1, \ldots\}$ be the sequence generated by Algorithm 5.1. Then,*

$$\|\mathbf{Q}(t)\| - G - \sqrt{m}D_2R \leq \|\mathbf{Q}(t+1)\| \leq \|\mathbf{Q}(t)\| + G, \forall t \geq 0.$$

Proof.

- **Proof of $\|\mathbf{Q}(t+1)\| \leq \|\mathbf{Q}(t)\| + G$:**

  Fix $t \geq 0$ and $k \in \{1, 2, \ldots, m\}$. The virtual queue update equation implies that

$$\begin{aligned}Q_k(t+1) &= \max\{Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)], 0\} \\ &\overset{(a)}{\leq} \max\{Q_k(t) + g_k^t(\mathbf{x}(t+1)), 0\},\end{aligned}$$

where (a) follows from the convexity of $g_k^t(\cdot)$.

Note that $Q_k(t+1) \geq 0$ and recall the fact that if $0 \leq a \leq \max\{b, 0\}$, then $a^2 \leq b^2$ for all $a, b \in \mathbb{R}$. Then, we have $[Q_k(t+1)]^2 \leq [Q_k(t) + g_k^t(\mathbf{x}(t+1))]^2$.

Summing over $k \in \{1, 2, \ldots, m\}$ yields

$$\|\mathbf{Q}(t+1)\|^2 \leq \|\mathbf{Q}(t) + \mathbf{g}^t(\mathbf{x}(t+1))\|^2.$$

Thus, $\|\mathbf{Q}(t+1)\| \leq \|\mathbf{Q}(t) + \mathbf{g}^t(\mathbf{x}(t+1))\| \leq \|\mathbf{Q}(t)\| + \|\mathbf{g}^t(\mathbf{x}(t+1))\| \leq \|\mathbf{Q}(t)\| + G$ where the last inequality follows from Assumption 5.1.

- **Proof of $\|\mathbf{Q}(t+1)\| \geq \|\mathbf{Q}(t)\| - G - \sqrt{m}D_2R$:**

  Since $Q_k(t) \geq 0$, it follows that $|Q_k(t+1) - Q_k(t)| \leq |g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)]|$. (This can be shown by considering $g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] \geq 0$ and $g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] < 0$ separately.) Thus, we have $\|\mathbf{Q}(t+1) - \mathbf{Q}(t)\| \leq G + \sqrt{m}D_2R$, which further implies $\|\mathbf{Q}(t+1)\| \geq \|\mathbf{Q}(t)\| - G - \sqrt{m}D_2R$ by the triangle inequality of norms.

□

Now, we are ready to present the main proof of Lemma 5.7. Note that Lemma 5.12 gives $\big|\|\mathbf{Q}(t+1)\| - \|\mathbf{Q}(t)\|\big| \leq G + \sqrt{m}D_2R$, which further implies that $\mathbb{E}[\|\mathbf{Q}(t+t_0)\| - \|\mathbf{Q}(t)\| | \mathbf{Q}(t)] \leq t_0(G + \sqrt{m}D_2R)$ when $\|\mathbf{Q}(t)\| < \theta$. It remains to prove $\mathbb{E}[\|\mathbf{Q}(t+1)\| - \|\mathbf{Q}(t)\| \big| \mathbf{Q}(t)] \leq -\frac{\epsilon}{2}t_0$ when $\|\mathbf{Q}(t)\| \geq \theta$. Note that $\|\mathbf{Q}(0)\| = 0 < \theta$.

Fix $t \geq 1$ and consider that $\|\mathbf{Q}(t)\| \geq \theta$. Let $\hat{\mathbf{x}} \in \mathcal{X}_0$ and $\epsilon > 0$ be defined in Assumption 5.2. Note that $\mathbb{E}[g_k^t(\hat{\mathbf{x}})] \leq -\epsilon, \forall k \in \{1, 2, \ldots, m\}, \forall t \in \{1, 2, \ldots\}$ since $\omega(t)$ are i.i.d. from the distribution of ω. Since $\hat{\mathbf{x}} \in \mathcal{X}_0$, by Lemma 5.4, for all $\tau \in \{t, t+1, \ldots, t+t_0-1\}$, we have

$$\begin{aligned}
&V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1) - \mathbf{x}(\tau)] + \sum_{k=1}^m Q_k(\tau)[\nabla g_k^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1) - \mathbf{x}(\tau)] + \alpha\|\mathbf{x}(\tau+1) - \mathbf{x}(\tau)\|^2 \\
\leq &V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(\tau)] + \sum_{k=1}^m Q_k(\tau)[\nabla g_k^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(\tau)] + \alpha[\|\hat{\mathbf{x}} - \mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}} - \mathbf{x}(\tau+1)\|^2].
\end{aligned}$$

Adding $\sum_{k=1}^m Q_k(\tau) g_k^\tau(\mathbf{x}(\tau))$ on both sides and noting that $g_k^\tau(\mathbf{x}(\tau)) + [\nabla g_k^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(\tau)] \leq$

$g_k^\tau(\hat{\mathbf{x}})$ by convexity yields

$$\begin{aligned}
&V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1)-\mathbf{x}(\tau)] + \sum_{k=1}^m Q_k(\tau)\big[g_k^\tau(\mathbf{x}(\tau)) + [\nabla g_k^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1)-\mathbf{x}(\tau)]\big] \\
&+\alpha\|\mathbf{x}(\tau+1)-\mathbf{x}(\tau)\|^2 \\
\le& V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\hat{\mathbf{x}}-\mathbf{x}(\tau)] + \sum_{k=1}^m Q_k(\tau) g_k^\tau(\hat{\mathbf{x}}) + \alpha[\|\hat{\mathbf{x}}-\mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\|^2].
\end{aligned}$$

Rearranging terms yields

$$\begin{aligned}
&\sum_{k=1}^m Q_k(t)\big[g_k^\tau(\mathbf{x}(t)) + [\nabla g_k^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1)-\mathbf{x}(\tau)]\big] \\
\le& V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\hat{\mathbf{x}}-\mathbf{x}(\tau)] - V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1)-\mathbf{x}(\tau)] \\
&+\alpha[\|\hat{\mathbf{x}}-\mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\|^2] - \alpha\|\mathbf{x}(\tau+1)-\mathbf{x}(\tau)\|^2 + \sum_{k=1}^m Q_k(t) g_k^\tau(\hat{\mathbf{x}}) \\
\le& V[\nabla f^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\hat{\mathbf{x}}-\mathbf{x}(\tau+1)] + \alpha[\|\hat{\mathbf{x}}-\mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\|^2] + \sum_{k=1}^m Q_k(\tau) g_k^\tau(\hat{\mathbf{x}}) \\
\overset{(a)}{\le}& V\|\nabla f^\tau(\mathbf{x}(\tau))\|\|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\| + \alpha[\|\hat{\mathbf{x}}-\mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\|^2] + \sum_{k=1}^m Q_k(\tau) g_k^\tau(\hat{\mathbf{x}}) \\
\overset{(b)}{\le}& VD_1R + \alpha[\|\hat{\mathbf{x}}-\mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\|^2] + \sum_{k=1}^m Q_k(\tau) g_k^\tau(\hat{\mathbf{x}}),
\end{aligned} \tag{5.21}$$

where (a) follows from the Cauchy-Schwarz inequality and (b) follows from Assumption 5.1.

By Lemma 5.2, for all $\tau \in \{t, t+1, \ldots, t+t_0-1\}$, we have

$$\begin{aligned}
\Delta(\tau) \le& \sum_{k=1}^m Q_k(\tau)\big[g_k^\tau(\mathbf{x}(\tau)) + [\nabla g_k^\tau(\mathbf{x}(\tau))]^\mathsf{T}[\mathbf{x}(\tau+1)-\mathbf{x}(\tau)]\big] + \frac{1}{2}(G+\sqrt{m}D_2R)^2 \\
\overset{(a)}{\le}& VD_1R + \frac{1}{2}(G+\sqrt{m}D_2R)^2 + \alpha[\|\hat{\mathbf{x}}-\mathbf{x}(\tau)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(\tau+1)\|^2] + \sum_{k=1}^m Q_k(\tau) g_k^\tau(\hat{\mathbf{x}}),
\end{aligned}$$

where (a) follows from (5.21).

Summing the above inequality over $\tau \in \{t, t+1, \ldots, t+t_0-1\}$, taking expectations conditional

on $\mathcal{W}(t-1)$ on both sides and recalling that $\Delta(\tau) = \frac{1}{2}\|\mathbf{Q}(\tau+1)\|^2 - \frac{1}{2}\|\mathbf{Q}(\tau)\|^2$ yields

$$
\begin{aligned}
&\mathbb{E}[\|\mathbf{Q}(t+t_0)\|^2 - \|\mathbf{Q}(t)\|^2\big|\mathcal{W}(t-1)] \\
\leq& 2VD_1Rt_0 + t_0[G+\sqrt{m}D_2R]^2 + 2\alpha\mathbb{E}[\|\hat{\mathbf{x}}-\mathbf{x}(t)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(t+t_0)\|^2|\mathcal{W}(t-1)] \\
&+ 2\sum_{\tau=t}^{t+t_0-1}\mathbb{E}[\sum_{k=1}^{m}Q_k(\tau)g_k^\tau(\hat{\mathbf{x}})|\mathcal{W}(t-1)] \\
\overset{(a)}{\leq}& 2VD_1Rt_0 + t_0(G+\sqrt{m}D_2R)^2 + 2\alpha R^2 - 2\epsilon\sum_{\tau=t}^{t+t_0-1}\mathbb{E}[\|\mathbf{Q}(\tau)\||\mathcal{W}(t-1)] \\
\overset{(b)}{\leq}& 2VD_1Rt_0 + t_0(G+\sqrt{m}D_2R)^2 + 2\alpha R^2 - 2\epsilon\sum_{\tau=0}^{t_0-1}\mathbb{E}[\|\mathbf{Q}(t)\| - \tau(G+\sqrt{m}D_2R)|\mathcal{W}(t-1)] \\
=& 2VD_1Rt_0 + t_0(G+\sqrt{m}D_2R)^2 + 2\alpha R^2 - 2\epsilon t_0\|\mathbf{Q}(t)\| + \epsilon t_0(t_0-1)(G+\sqrt{m}D_2R) \\
\leq& 2VD_1Rt_0 + t_0(G+\sqrt{m}D_2R)^2 + 2\alpha R^2 - 2\epsilon t_0\|\mathbf{Q}(t)\| + \epsilon t_0^2(G+\sqrt{m}D_2R)
\end{aligned}
$$

where (a) follows because $\|\hat{\mathbf{x}}-\mathbf{x}(t)\|^2 - \|\hat{\mathbf{x}}-\mathbf{x}(t+t_0)\|^2 \leq R^2$ by Assumption 5.1 and

$$\mathbb{E}[\sum_{k=1}^{m}Q_k(\tau)g_k^\tau(\hat{\mathbf{x}})|\mathcal{W}(t-1)] \leq -\epsilon\mathbb{E}[\|\mathbf{Q}(\tau)\||\mathcal{W}(t-1)], \forall\tau\in\{t,t+1,\ldots,t+t_0-1\}$$

by Lemma 5.11; (b) follows because $\|\mathbf{Q}(t+1)\| \geq \|\mathbf{Q}(t)\| - (G+\sqrt{m}D_2R), \forall t$ by Lemma 5.12.

This inequality can be rewritten as

$$
\begin{aligned}
&\mathbb{E}[\|\mathbf{Q}(t+t_0)\|^2\big|\mathcal{W}(t-1)] \\
\leq& \|\mathbf{Q}(t)\|^2 - 2\epsilon t_0\|\mathbf{Q}(t)\| + 2VD_1Rt_0 + 2\alpha R^2 + t_0(G+\sqrt{m}D_2R)^2 + \epsilon t_0^2(G+\sqrt{m}D_2R) \\
\overset{(a)}{\leq}& \|\mathbf{Q}(t)\|^2 - \epsilon t_0\|\mathbf{Q}(t)\| - \epsilon t_0[\frac{\epsilon}{2}t_0 + (G+\sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R + (G+\sqrt{m}D_2R)^2}{\epsilon}] \\
&+ 2VD_1Rt_0 + 2\alpha R^2 + t_0(G+\sqrt{m}D_2R)^2 + \epsilon t_0^2(G+\sqrt{m}D_2R) \\
=& \|\mathbf{Q}(t)\|^2 - \epsilon t_0\|\mathbf{Q}(t)\| - \frac{\epsilon^2 t_0^2}{2} \\
\leq& [\|\mathbf{Q}(t)\| - \frac{\epsilon}{2}t_0]^2,
\end{aligned}
$$

where (a) follows from the hypothesis that $\|\mathbf{Q}(t)\| \geq \theta = \frac{\epsilon}{2}t_0 + (G+\sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R+(G+\sqrt{m}D_2R)^2}{\epsilon}$.

Taking square root on both sides yields

$$\sqrt{\mathbb{E}[\|\mathbf{Q}(t+t_0)\|^2 \big| \mathcal{W}(t-1)]} \leq \|\mathbf{Q}(t)\| - \frac{\epsilon}{2} t_0.$$

By the concavity of function $\sqrt{x}$ and Jensen's inequality, we have

$$\mathbb{E}[\|\mathbf{Q}(t+t_0)\| \big| \mathcal{W}(t-1)] \leq \sqrt{\mathbb{E}[\|\mathbf{Q}(t+t_0)\|^2 | \mathcal{W}(t-1)]} \leq \|\mathbf{Q}(t)\| - \frac{\epsilon}{2} t_0.$$

5.5.3 Proof of Lemma 5.8

Fix $t \geq 1$. By Lemma 5.4, we have

$$\begin{aligned}
&V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] + \sum_{k=1}^{m} Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] + \alpha \|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
\leq &V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{z} - \mathbf{x}(t)] + \sum_{k=1}^{m} Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{z} - \mathbf{x}(t)] + \alpha[\|\mathbf{z} - \mathbf{x}(t)\|^2 - \|\mathbf{z} - \mathbf{x}(t+1)\|^2].
\end{aligned}$$

Adding constant $Vf^t(\mathbf{x}(t)) + \sum_{k=1}^{m} Q_k(t) g_k^t(\mathbf{x}(t))$ on both sides; and noting that $f^t(\mathbf{x}(t)) + [\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{z} - \mathbf{x}(t)] \leq f^t(\mathbf{z})$ and $g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{z} - \mathbf{x}(t)] \leq g_k^t(\mathbf{z})$ by convexity yields

$$\begin{aligned}
&Vf^t(\mathbf{x}(t)) + V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] + \sum_{k=1}^{m} Q_k(t) \big[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] \big] \\
&\quad + \alpha \|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
\leq &Vf^t(\mathbf{z}) + \sum_{k=1}^{m} Q_k(t) g_k^t(\mathbf{z}) + \alpha[\|\mathbf{z} - \mathbf{x}(t)\|^2 - \|\mathbf{z} - \mathbf{x}(t+1)\|^2]. \qquad (5.22)
\end{aligned}$$

By Lemma 5.2, we have

$$\Delta(t) \leq \sum_{k=1}^{m} Q_k(t) \big[g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] \big] + \frac{1}{2}[G + \sqrt{m} D_2 R]^2. \qquad (5.23)$$

Summing (5.22) and (5.23), cancelling common terms and rearranging terms yields

$$\begin{aligned}
Vf^t(\mathbf{x}(t)) \leq &Vf^t(\mathbf{z}) - \Delta(t) + \sum_{k=1}^{m} Q_k(t) g_k^t(\mathbf{z}) + \alpha[\|\mathbf{z} - \mathbf{x}(t)\|^2 - \|\mathbf{z} - \mathbf{x}(t+1)\|^2] \\
&- V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] - \alpha \|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 + \frac{1}{2}[G + \sqrt{m} D_2 R]^2 \qquad (5.24)
\end{aligned}$$

Note that

$$
\begin{aligned}
&- V[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1)-\mathbf{x}(t)] - \alpha\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2 \\
\overset{(a)}{\leq} & V\|\nabla f^t(\mathbf{x}(t))\|\|\mathbf{x}(t+1)-\mathbf{x}(t)\| - \alpha\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2 \\
\overset{(b)}{\leq} & VD_1\|\mathbf{x}(t+1)-\mathbf{x}(t)\| - \alpha\|\mathbf{x}(t+1)-\mathbf{x}(t)\|^2 \\
= & -\alpha\big[\|\mathbf{x}(t+1)-\mathbf{x}(t)\| - \frac{VD_1}{2\alpha}\big]^2 + \frac{V^2D_1^2}{4\alpha} \\
\leq & \frac{V^2D_1^2}{4\alpha} \qquad (5.25)
\end{aligned}
$$

where (a) follows from the Cauchy-Schwarz inequality; and (b) follows from Assumption 5.1.

Substituting (5.25) into (5.24) yields

$$
\begin{aligned}
Vf^t(\mathbf{x}(t)) \leq & Vf^t(\mathbf{z}) - \Delta(t) + \sum_{k=1}^m Q_k(t)g_k^t(\mathbf{z}) + \alpha[\|\mathbf{z}-\mathbf{x}(t)\|^2 - \|\mathbf{z}-\mathbf{x}(t+1)\|^2] + \frac{V^2D_1^2}{4\alpha} \\
& + \frac{1}{2}[G+\sqrt{m}D_2R]^2.
\end{aligned}
$$

Summing over $t \in \{1,2,\ldots,T\}$ yields

$$
\begin{aligned}
V\sum_{t=1}^T f^t(\mathbf{x}(t)) \leq & V\sum_{t=1}^T f^t(\mathbf{z}) - \sum_{t=1}^T \Delta(t) + \alpha\sum_{t=1}^T[\|\mathbf{z}-\mathbf{x}(t)\|^2 - \|\mathbf{z}-\mathbf{x}(t+1)\|^2] + \frac{V^2D_1^2}{4\alpha}T \\
& + \frac{1}{2}(G+\sqrt{m}D_2R)^2T + \sum_{t=1}^T\big[\sum_{k=1}^m Q_k(t)g_k^t(\mathbf{z})\big] \\
\overset{(a)}{=} & V\sum_{t=1}^T f^t(\mathbf{z}) + L(1) - L(T+1) + \alpha\|\mathbf{z}-\mathbf{x}(1)\|^2 - \alpha\|\mathbf{z}-\mathbf{x}(T+1)\|^2 + \frac{V^2D_1^2}{4\alpha}T \\
& + \frac{1}{2}(G+\sqrt{m}D_2R)^2T + \sum_{t=1}^T\big[\sum_{k=1}^m Q_k(t)g_k^t(\mathbf{z})\big] \\
\overset{(b)}{\leq} & V\sum_{t=1}^T f^t(\mathbf{z}) + \alpha R^2 + \frac{V^2D_1^2}{4\alpha}T + \frac{1}{2}(G+\sqrt{m}D_2R)^2T + \sum_{t=1}^T\big[\sum_{k=1}^m Q_k(t)g_k^t(\mathbf{z})\big].
\end{aligned}
$$

where (a) follows by recalling that $\Delta(t) = L(t+1) - L(t)$; and (b) follows because $\|\mathbf{z}-\mathbf{x}(1)\| \leq R$ by Assumption 5.1, $L(1) = \frac{1}{2}\|\mathbf{Q}(1)\|^2 = 0$ and $L(T+1) = \frac{1}{2}\|\mathbf{Q}(T+1)\|^2 \geq 0$.

Dividing both sides by V yields the desired result.

5.5.4 Proof of Theorem 5.3

Define random process $Z(t) = \|\mathbf{Q}(t)\|, \forall t \in \{1, 2, \ldots\}$. By Lemma 5.7, $Z(t)$ satisfies the conditions in Lemma 5.5 with $\delta_{\max} = G + \sqrt{m}D_2R$, $\zeta = \frac{\epsilon}{2}$ and

$$\theta = \frac{\epsilon}{2}t_0 + (G + \sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R + (G + \sqrt{m}D_2R)^2}{\epsilon}.$$

Fix $T \geq 1$ and $0 < \lambda < 1$. Taking $\mu = \lambda/(T+1)$ in part 2 of Lemma 5.5 yields

$$\Pr(\|\mathbf{Q}(t)\| \geq \gamma) \leq \frac{\lambda}{T+1}, \forall t \in \{1, 2, \ldots, T+1\},$$

where $\gamma = \frac{\epsilon}{2}t_0 + 2(G+\sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R+(G+\sqrt{m}D_2R)^2}{\epsilon} + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log\big(\frac{32(G+\sqrt{m}D_2R)^2}{\epsilon^2}\big) + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log(\frac{T+1}{\lambda})$.

By the union bound of probability, we have

$$\Pr(\|\mathbf{Q}(t)\| \geq \gamma \text{ for some } t \in \{1, 2, \ldots, T+1\}) \leq \lambda.$$

This implies

$$\Pr(\|\mathbf{Q}(t)\| \leq \gamma \text{ for all } t \in \{1, 2, \ldots, T+1\}) \geq 1 - \lambda. \tag{5.26}$$

Taking $t_0 = \lceil\sqrt{T}\rceil$, $V = \sqrt{T}$ and $\alpha = T$ yields

$$\gamma = O(\sqrt{T}\log(T)) + O(\sqrt{T}\log(\frac{1}{\lambda})) = O(\sqrt{T}\log(T)\log(\frac{1}{\lambda})) \tag{5.27}$$

Recall that by Corollary 5.2 (with $V = \sqrt{T}$ and $\alpha = T$), for all $k \in \{1, 2, \ldots, m\}$, we have

$$\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq \|\mathbf{Q}(T+1)\| + \frac{\sqrt{T}D_1D_2}{2} + \frac{\sqrt{m}D_2^2}{2T}\sum_{t=1}^{T}\|\mathbf{Q}(t)\|. \tag{5.28}$$

It follows from (5.26)-(5.28) that

$$\Pr\big(\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq O(\sqrt{T}\log(T)\log(\frac{1}{\lambda}))\big) \geq 1 - \lambda.$$

5.5.5 Proof of Lemma 5.9

Intuitively, the second term on the right side in the lemma bounds the probability that $|Z(\tau+1)-Z(\tau)| > c$ for any $\tau \in \{0,1,\ldots,t-1\}$, while the first term on the right side comes from the conventional Hoeffding-Azuma inequality. However, it is unclear whether or not $Z(t)$ is still a supermartigale conditional on the event that $|Z(\tau+1)-Z(\tau)| \leq c$ for any $\tau \in \{0,1,\ldots,t-1\}$.That's why it is important to have $\{|Z(t+1)-Z(t)| > c\} \subseteq \{Y(t) > 0\}$ and $Y(t) \in \mathcal{F}(t)$, which means the boundedness of $|Z(t+1)-Z(t)|$ can be inferred from another random variable $Y(t)$ that belongs to $\mathcal{F}(t)$. The proof of Lemma 5.9 uses the truncation method to construct an auxiliary supermargingale.

Recall the definition of stoping time given as follows:

Definition 5.1 ([Dur10]). *Let $\{\emptyset,\Omega\} = \mathcal{F}(0) \subseteq \mathcal{F}(1) \subseteq \mathcal{F}(2)\cdots$ be a filtration. A discrete random variable T is a stoping time (also known as an option time) if for any integer $t < \infty$,*

$$\{T = t\} \in \mathcal{F}(t),$$

i.e. the event that the stopping time occurs at time t is contained in the information up to time t.

The next theorem summarizes that a supermartingale truncated at a stoping time is still a supermartingale.

Theorem 5.5. *(Theorem 5.2.6 in [Dur10]) If random variable T is a stopping time and $Z(t)$ is a supermartingale, then $Z(t \wedge T)$ is also a supermartingale, where $a \wedge b \triangleq \min\{a,b\}$.*

To prove this lemma, we first construct a new supermartingale by truncating the original supermartingale at a carefully chosen stopping time such that the new supermartingale has bounded differences.

Define integer random variable $T = \inf\{t \geq 0 : Y(t) > 0\}$. That is, T is the first time t when $Y(t) > 0$ happens. Now, we show that T is a stoping time and if we define $\widetilde{Z}(t) = Z(t \wedge T)$, then $\{\widetilde{Z}(t) \neq Z(t)\} \subseteq \bigcup_{\tau=0}^{t-1}\{Y(\tau) > 0\}, \forall t \geq 1$ and $\widetilde{Z}(t)$ is a supermartingale with differences bounded by c .

1. **To show T is a stoping time:** Note that $\{T = 0\} = \{Y(0) > 0\} \in \mathcal{F}(0)$. Fix integer

$t' > 0$, we have

$$\begin{aligned}\{T = t'\} =& \big\{ \inf\{t \geq 0 : Y(t) > 0\} = t' \big\} \\ =& \big\{ \bigcap_{\tau=0}^{t'-1} \{|Y(\tau) \leq 0\} \big\} \bigcap \{Y(t') > 0\} \\ \overset{(a)}{\in}& \mathcal{F}(t')\end{aligned}$$

where (a) follows because $\{Y(\tau) \leq 0\} \in \mathcal{F}(\tau) \subseteq \mathcal{F}(t')$ for all $\tau \in \{0, 1, \ldots, t'-1\}$ and $\{Y(t') > 0\} \in \mathcal{F}(t')$. It follows that T is a stoping time.

2. **To show** $\{\widetilde{Z}(t) \neq Z(t)\} \subseteq \bigcup_{\tau=0}^{t-1} \{Y(\tau) > 0\}, \forall t \geq 1$**:** Fix $t = t' > 1$. Note that

$$\begin{aligned}\{\widetilde{Z}(t') \neq Z(t')\} \overset{(a)}{\subseteq} \{T < t'\} &= \big\{ \inf\{t > 0 : Y(t) > 0\} < t' \big\} \\ &\subseteq \bigcup_{\tau=0}^{t'-1} \{Y(\tau) > 0\}\end{aligned}$$

where (a) follows by noting that if $T \geq t'$ then $\widetilde{Z}(t') = Z(t' \wedge T) = Z(t')$.

3. **To show** $\widetilde{Z}(t)$ **is a supermartingale with differences bounded by** c**:** Since random variable T is proven to be a stoping time, $\widetilde{Z}(t) = Z(t \wedge T)$ is a supermartingale by Theorem 5.5. It remains to show $|\widetilde{Z}(t+1) - \widetilde{Z}(t)| \leq c, \forall t \geq 0$. Fix integer $t = t' \geq 0$. Note that

$$\begin{aligned}&|\widetilde{Z}(t'+1) - \widetilde{Z}(t')| \\ =&|Z(T \wedge (t'+1)) - Z(T \wedge t')| \\ =&|\mathbf{1}_{\{T \geq t'+1\}}[Z(T \wedge (t'+1)) - Z(T \wedge t')] + \mathbf{1}_{\{T \leq t'\}}[Z(T \wedge (t'+1)) - Z(T \wedge t')]| \\ =&|\mathbf{1}_{\{T \geq t'+1\}}[Z(t'+1) - Z(t')] + \mathbf{1}_{\{T \leq t'\}}[Z(T) - Z(T)]| \\ =&\mathbf{1}_{\{T \geq t'+1\}}|Z(t'+1) - Z(t')|\end{aligned}$$

Now consider $T \leq t'$ and $T \geq t'+1$ separately.

- In the case when $T \leq t'$, it is straightforward that $|\widetilde{Z}(t'+1) - \widetilde{Z}(t')| = \mathbf{1}_{\{T \geq t'+1\}}|Z(t'+1) - Z(t')| = 0 \leq c$.
- Consider the case when $T \geq t'+1$. By the definition of T, we know that $\{T \geq t'+1\} = \big\{ \inf\{t \geq 0 : Y(t) > 0\} \geq t'+1 \big\} \subseteq \bigcap_{\tau=0}^{t'} \{Y(\tau) \leq 0\} \subseteq \bigcap_{\tau=0}^{t'} \{|Z(\tau+1) - Z(\tau)| \leq$

$c\}$, where the last inclusion follows from the fact that $\{|Z(\tau+1)-Z(\tau)|>c\}\subseteq\{Y(\tau)>0\}$. That is, when $T\geq t'+1$, we must have $|Z(\tau+1)-Z(\tau)|\leq c$ for all $\tau\in\{1,\ldots,t'\}$, which further implies that $|Z(t'+1)-Z(t')|\leq c$. Thus, when $T\geq t'+1$, $|\widetilde{Z}(t'+1)-\widetilde{Z}(t')|=\mathbf{1}_{\{T\geq t'+1\}}|Z(t'+1)-Z(t')|\leq c$.

Combining two cases together proves $|\widetilde{Z}(t'+1)-\widetilde{Z}(t')|\leq c$.

Since $\widetilde{Z}(t)$ is a supermartingale with bounded differences c and $\widetilde{Z}(0)=Z(0)=0$, by the conventional Hoeffding-Azuma inequality, for any $z>0$, we have

$$\Pr(\widetilde{Z}(t)\geq z)\leq e^{-z^2/(2tc^2)} \tag{5.29}$$

Finally, we have

$$\begin{aligned}\Pr(Z(t)\geq z)=&\Pr(\widetilde{Z}(t)=Z(t),Z(t)\geq z)+\Pr(\widetilde{Z}(t)\neq Z(t),Z(t)\geq z)\\ \leq&\Pr(\widetilde{Z}(t)\geq z)+\Pr(\widetilde{Z}(t)\neq Z(t))\\ \overset{(a)}{\leq}&e^{-z^2/(2tc^2)}+\Pr(\bigcup_{\tau=0}^{t-1}Y(\tau)>0)\\ \overset{(b)}{\leq}&e^{-z^2/(2tc^2)}+\sum_{\tau=0}^{t-1}p(\tau)\end{aligned}$$

where (a) follows from equation (5.29) and the second bullet in the above; and (b) follows from the union bound and the hypothesis that $\Pr(Y(\tau)>0)\leq p(\tau),\forall\tau$.

5.5.6 Proof of Theorem 5.4

Define $Z(0)=0$ and $Z(t)=\sum_{\tau=1}^{t}\sum_{k=1}^{m}Q_k(\tau)g_k^\tau(\mathbf{x}^*)$. Recall $\mathcal{W}(0)=\{\emptyset,\Omega\}$ and $\mathcal{W}(t)=\sigma(\omega(1),\ldots,\omega(t)),\forall t\geq 1$. The next lemma shows that for any $c>0$, $Z(t)$ satisfies Lemma 5.9 with $\mathcal{F}(t)=\mathcal{W}(t)$ and $Y(t)=\|\mathbf{Q}(t+1)\|-\frac{c}{G}$.

Lemma 5.13. *Let $\mathbf{x}^*\in\mathcal{X}_0$ be any solution satisfying $\tilde{\mathbf{g}}(\mathbf{x}^*)\leq\mathbf{0}$, e.g., $\mathbf{x}^*=\operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\{\sum_{t=1}^{T}f^t(\mathbf{x})\}$. Let $c>0$ be an arbitrary constant. Under Algorithm 5.1, if we define $Z(0)=0$ and*

$$Z(t)=\sum_{\tau=1}^{t}\sum_{k=1}^{m}Q_k(\tau)g_k^\tau(\mathbf{x}^*),\forall t\geq 1,$$

then $\{Z(t), t \geq 0\}$ *is a supermartingale adapted to filtration* $\{\mathcal{W}(t), t \geq 0\}$ *such that*

$$\{|Z(t+1) - Z(t)| > c\} \subseteq \{Y(t) > 0\}, \forall t \geq 0$$

where $Y(t) = \|\mathbf{Q}(t+1)\| - \frac{c}{G}$ *is a random variable adapted to* $\mathcal{W}(t)$*. (Note that* G *is a constant defined in Assumption 5.1.)*

Proof. It is easy to say $\{Z(t), t \geq 0\}$ is adapted $\{\mathcal{W}(t), t \geq 0\}$. It remains to show $\{Z(t), t \geq 0\}$ is a supermartingale. Note that $Z(t+1) = Z(t) + \sum_{k=1}^{m} Q_k(t+1) g_k^{t+1}(\mathbf{x}^*)$ and

$$\begin{aligned}
\mathbb{E}[Z(t+1)|\mathcal{W}(t)] =& \mathbb{E}[Z(t) + \sum_{k=1}^{m} Q_k(t+1) g_k^{t+1}(\mathbf{x}^*)|\mathcal{W}(t)] \\
\overset{(a)}{=}& Z(t) + \sum_{k=1}^{m} Q_k(t+1) \mathbb{E}[g_k^{t+1}(\mathbf{x}^*)] \\
\overset{(b)}{\leq}& Z(t)
\end{aligned}$$

where (a) follows from the fact that $Z(t) \in \mathcal{W}(t)$, $\mathbf{Q}(t+1) \in \mathcal{W}(t)$ and $\mathbf{g}^{t+1}(\mathbf{x}^*)$ is independent of $\mathcal{W}(t)$; and (b) follows from $\mathbb{E}[g_k^{t+1}(\mathbf{x}^*)] = \tilde{g}_k(\mathbf{x}^*) \leq 0$ which further follows from $\omega(t)$ are i.i.d. samples. Thus, $\{Z(t), t \geq 0\}$ is a supermartingale.

We further note that

$$|Z(t+1) - Z(t)| = |\sum_{k=1}^{m} Q_k(t+1) g_k^{t+1}(\mathbf{x}^*)| \overset{(a)}{\leq} \|\mathbf{Q}(t+1)\| G$$

where (a) follows from the Cauchy-Schwarz inequality and the assumption that $\|\mathbf{g}^t(\mathbf{x}^*)\| \leq G$.

This implies that if $|Z(t+1) - Z(t)| > c$, then $\|\mathbf{Q}(t)\| > \frac{c}{G}$. Thus, $\{|Z(t+1) - Z(t)| > c\} \subseteq \{\|\mathbf{Q}(t+1)\| > \frac{c}{G}\}$. Since $\mathbf{Q}(t+1)$ is adapted to $\mathcal{W}(t)$, it follows that $Y(t) = \|\mathbf{Q}(t+1)\| - \frac{c}{G}$ is a random variable adapted to $\mathcal{W}(t)$. □

By Lemma 5.13, $Z(t)$ satisfies Lemma 5.9. Fix $T \geq 1$, Lemma 5.9 implies that

$$\Pr(\sum_{t=1}^{T} \sum_{k=1}^{m} Q_k(t) g_k^t(\mathbf{x}^*) \geq \gamma) \leq \underbrace{e^{-\gamma^2/(2Tc^2)}}_{\text{(I)}} + \underbrace{\sum_{t=0}^{T-1} \Pr(\|\mathbf{Q}(t+1)\| > \frac{c}{G})}_{\text{(II)}} \tag{5.30}$$

Fix $0 < \lambda < 1$. In the following, we shall choose γ and c such that both term (I) and term

(II) in (5.30) are no larger than $\frac{\lambda}{2}$.

Recall that by Lemma 5.7, random process $\widetilde{Z}(t) = \|\mathbf{Q}(t)\|$ satisfies the conditions in Lemma 5.5 with $\delta_{\max} = G + \sqrt{m}D_2R$, $\zeta = \frac{\epsilon}{2}$ and

$$\theta = \frac{\epsilon}{2}t_0 + (G + \sqrt{m}D_2R)t_0 + \frac{2\alpha R^2}{t_0\epsilon} + \frac{2VD_1R + (G + \sqrt{m}D_2R)^2}{\epsilon}.$$

To guarantee term (II) is no lareger than $\frac{\lambda}{2}$, it suffices to choose c such that

$$\Pr(\|\mathbf{Q}(t)\| > \frac{c}{G}) \leq \frac{\lambda}{2T}, \forall t \in \{1, 2, \ldots, T\}$$

By part 2 of Lemma 5.5 (with $\mu = \frac{\lambda}{2T}$), the above inequality holds if we choose $c = t_0\frac{\epsilon}{2}G + 2t_0(G + \sqrt{m}D_2R)G + \frac{2\alpha R^2}{t_0\epsilon}G + \frac{2VD_1R+(G+\sqrt{m}D_2R)^2}{\epsilon}G + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log\big(\frac{32(G+\sqrt{m}D_2R)^2}{\epsilon^2}\big)G + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log(\frac{2T}{\lambda})G$ where $t_0 > 0$ is an arbitrary integer.

Once c is chosen, we further need to choose γ such that term (I) in (5.30) is $\frac{\lambda}{2}$. It follows that if $\gamma = \sqrt{2T}\log^{0.5}(\frac{2}{\lambda})c = \sqrt{2T}\log^{0.5}(\frac{2}{\lambda})\big[\frac{\epsilon}{2}t_0G + 2t_0(G+\sqrt{m}D_2R)G + \frac{2\alpha R^2}{t_0\epsilon}G + \frac{2VD_1R+(G+\sqrt{m}D_2R)^2}{\epsilon}G + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log\big(\frac{32(G+\sqrt{m}D_2R)^2}{\epsilon^2}\big)G + t_0\frac{8(G+\sqrt{m}D_2R)^2}{\epsilon}\log(\frac{2T}{\lambda})G\big]$, then the term (I) is equal to $\frac{\lambda}{2}$.

Thus, we have

$$\Pr(\sum_{t=1}^{T}\sum_{k=1}^{m} Q_k(t)g_k^t(\mathbf{x}^*) \geq \gamma) \leq \lambda,$$

which further implies,

$$\Pr(\sum_{t=1}^{T}\sum_{k=1}^{m} Q_k(t)g_k^t(\mathbf{x}^*) \leq \gamma) \geq 1 - \lambda. \tag{5.31}$$

Note that if we take $t_0 = \lceil\sqrt{T}\rceil$, $V = \sqrt{T}$ and $\alpha = T$, then $\gamma = O\big(T\log(T)\log^{0.5}(\frac{1}{\lambda})\big) + O\big(T\log^{1.5}(\frac{1}{\lambda})\big) = O\big(T\log(T)\log^{1.5}(\frac{1}{\lambda})\big)$.

By Lemma 5.8 (with $\mathbf{z} = \mathbf{x}^*$, $V = \sqrt{T}$ and $\alpha = T$), we have

$$\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{x}^*) + \sqrt{T}R^2 + \frac{D_1^2}{4}\sqrt{T} + \frac{1}{2}[G + \sqrt{m}D_2R]^2\sqrt{T} + \frac{1}{\sqrt{T}}\sum_{t=1}^{T}\big[\sum_{k=1}^{m} Q_k(t)g_k^t(\mathbf{x}^*)\big] \tag{5.32}$$

Substituting (5.31) into (5.32) yields

$$\Pr\Big(\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{x}^*) + O\big(\sqrt{T}\log(T)\log^{1.5}(\frac{1}{\lambda})\big)\Big) \geq 1-\lambda.$$

Chapter 6

Online Convex Optimization with Long Term Constraints

This chapter focuses on "online convex optimization with long term constraints", which is a special case problem of online convex optimization with stochastic constraints considered in Chapter 5 such that $g_k^t(\mathbf{x}) \equiv g_k(\mathbf{x})$ are perfectly known and do not depend on time. Ideally, this problem can be solved by Zinkevich's online gradient descent given by

$$\mathbf{x}(t+1) = \mathcal{P}_{\mathcal{X}}\left[\mathbf{x}(t) - \gamma \nabla f^t(\mathbf{x}(t))\right], \tag{6.1}$$

where $\mathcal{P}_{\mathcal{X}}[\cdot]$ represents the projection onto the convex set $\mathcal{X} = \{\mathbf{x} \in \mathcal{X}_0 : g_k(\mathbf{x}) \leq 0, k \in \{1, 2, \ldots, m\}\}$ and γ is the step size, also known as the learning rate.

In the case when $\mathcal{X}$ is a simple set, such as when there are no $g_k(\mathbf{x})$ are missing and $\mathcal{X}_0$ is a multidimensional box, the projection $\mathcal{P}_{\mathcal{X}}[\cdot]$ often has a closed form solution or enjoys low complexity. However, if $\mathcal{X}$ is complicated, e.g., functional inequality constraints $g_k(\mathbf{x}) \leq 0$ are complicated, then the equation (6.1) requires us to solve the following convex program:

$$\min \quad \|\mathbf{x} - [\mathbf{x}(t) - \gamma \nabla f^t(\mathbf{x}(t))]\|^2 \tag{6.2}$$

$$\text{s.t.} \quad g_k(\mathbf{x}) \leq 0, \forall k \in \{1, 2, \ldots, m\} \tag{6.3}$$

$$\mathbf{x} \in \mathcal{X}_0 \in \mathbb{R}^n \tag{6.4}$$

which can yield heavy computation and/or storage burden at each round. For instance, the interior point method (or other Newton-type methods) is an iterative algorithm and takes a number of iterations to approach the solution to the above convex program. The computation and memory space complexity at each iteration is between $O(n^2)$ and $O(n^3)$, where n is the

dimension of $\mathbf{x}$.

To circumvent the computational challenge of the projection operator, online convex optimization with long term constraints is first considered in [MJY12]. In online convex optimization with long term constraints, complicated functional constraints $g_k(\mathbf{x}) \leq 0$ are relaxed to be soft long term constraints. That is, we do not require $\mathbf{x}(t) \in \mathcal{X}_0$ to satisfy $g_k(\mathbf{x}(t)) \leq 0$ at each round, but only require that $\sum_{t=1}^{T} g_k(\mathbf{x}(t))$, called constraint violations, grows sub-linearly. [MJY12] proposes two algorithms such that one achieves $O(\sqrt{T})$ regret and $O(T^{3/4})$ constraint violations; and the other achieves $O(T^{2/3})$ for both regret and constraint violations when set $\mathcal{X}$ can be represented by linear constraints. [JHA16] recently extends the algorithm of [MJY12] to achieve $O(T^{\max\{\beta,1-\beta\}})$ regret and $O(T^{1-\beta/2})$ constraint violations where $\beta \in (0,1)$ is a user-defined tradeoff parameter. By choosing $\beta = 1/2$ or $\beta = 2/3$, the $[O(\sqrt{T}), O(T^{3/4})]$ or $[O(T^{2/3}), O(T^{2/3})]$ regret and constraint violations of [MJY12] are recovered. It is easy to observe that the best regret or constraint violations in [JHA16] are $O(\sqrt{T})$ under different β values. However, the algorithm of [JHA16] can not achieve $O(\sqrt{T})$ regret and $O(\sqrt{T})$ constraint violations simultaneously.

As discussed in Chapter 5, Algorithm 5.1 developed in Chapter 5 can solve online convex optimization with long term constraints and achieves $O(\sqrt{T})$ regret and $O(\sqrt{T})$ constraint violations. This chapter proposes a new algorithm that can achieve $O(\sqrt{T})$ regret and finite constraint violations that do not grow with T; and hence yields improved performance in comparison to prior works [MJY12, JHA16] and our own Algorithm 5.1. This new algorithm is also closely related to the new Lagrangian dual methods with $O(1/t)$ convergence developed in Chapter 3 for deterministic constrained convex programs. The results in this chapter are originally developed in our technical report [YN16b].

Many engineering problems can be directly formulated as online convex optimization with long term constraints. For example, problems with energy or monetary constraints often define these in terms of long term time averages rather than instantaneous constraints. In general, we assume that instantaneous constraints are incorporated into the set $\mathcal{X}_0$; and long term constraints are represented via functional constraints $g_k(\mathbf{x})$. Two example problems are given as follows. More examples can be found in [MJY12] and [JHA16].

- In the application of online display advertising [GT11a, GMPV09], the publisher needs to iteratively allocate "impressions" to advertisers to optimize some online concave utilities for each advertiser. The utility is typically unknown when the decision is made but can be

inferred later by observing user click behaviors under the given allocations. Since each advertiser usually specifies a certain budget for a period, the "impressions" should be allocated to maximize advertisers' long term utilities subject to long term budget constraints.

- In the application of network routing in a neutral or adversarial environment, the decision maker needs to iteratively make routing decisions to maximize network utilities. Furthermore, link quality can vary after each routing decision is made. The routing decisions should satisfy the long term flow conservation constraint at each intermediate node so that queues do not overflow.

6.1 Problem Statement and New Algorithm

This section introduces the problem of online convex optimization with long term constraints and presents our new algorithm.

6.1.1 Problem Statement

Let $\mathcal{X}_0$ be a compact convex set and $g_k(\mathbf{x}), k \in \{1, 2, \ldots, m\}$ be continuous convex functions. Denote the stacked vector of multiple functions $g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})$ as $\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), \ldots, g_m(\mathbf{x})]^{\mathsf{T}}$. Define $\mathcal{X} = \{\mathbf{x} \in \mathcal{X}_0 : g_k(\mathbf{x}) \leq 0, i \in \{1, 2, \ldots, m\}\}$. Let $f^t(\mathbf{x})$ be a sequence of continuous convex loss functions which are determined by nature (or by an adversary) such that $f^t(\mathbf{x})$ is unknown to the decision maker until the end of round t. For any sequence $\mathbf{x}(t)$ yielded by an online algorithm, define $\sum_{t=1}^{T} f^t(\mathbf{x}(t)) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f^t(\mathbf{x})$ as the regret and $\sum_{t=1}^{T} g_k(\mathbf{x}(t)), k \in \{1, 2, \ldots, m\}$ as the constraint violations. The goal of online convex optimization with long term constraints is to choose $\mathbf{x}(t) \in \mathcal{X}_0$ for each round t such that both the regret and the constraint violations grow sub-linearly with respect to T. Throughout this chapter, we consider online convex optimization with long term constraints satisfying the following assumptions:

Assumption 6.1 (Basic Assumptions)**.**

- *The loss functions have bounded gradients on $\mathcal{X}_0$. That is, there exists $D > 0$ such that $\|\nabla f^t(\mathbf{x})\| \leq D$ for all $\mathbf{x} \in \mathcal{X}_0$ and all t.*

- *There exists a constant β such that $\|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})\| \leq \beta\|\mathbf{x} - \mathbf{y}\|$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}_0$, i.e., $\mathbf{g}(\mathbf{x})$ is Lipschitz continuous with modulus β.*

- *There exists a constant G such that $\|\mathbf{g}(\mathbf{x})\| \leq G$ for all $\mathbf{x} \in \mathcal{X}_0$.*

- *There exists a constant R such that $\|\mathbf{x} - \mathbf{y}\| \leq R$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}_0$.*

Note that the existence of G follows directly from the compactness of set $\mathcal{X}_0$ and the continuity of $\mathbf{g}(\mathbf{x})$. The existence of R follows directly from the compactness of set $\mathcal{X}_0$.

Assumption 6.2 (Interior Point Assumption)**.** *There exists $\epsilon > 0$ and $\hat{\mathbf{x}} \in \mathcal{X}_0$ such that $g_k(\hat{\mathbf{x}}) \leq -\epsilon$ for all $k \in \{1, 2, \ldots, m\}$.*

6.1.2 New Algorithm

Define $\tilde{\mathbf{g}}(\mathbf{x}) = c\mathbf{g}(\mathbf{x})$ where $c > 0$ is an algorithm parameter. Note that each $\tilde{g}_k(\mathbf{x})$ is still a convex function and $\tilde{\mathbf{g}}(\mathbf{x}) \leq \mathbf{0}$ if and only if $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$. The next lemma trivially follows.

Lemma 6.1. *If online convex optimization with long term constraints satisfies Assumptions 6.1 and 6.2, then*

- *$\|\tilde{\mathbf{g}}(\mathbf{x}) - \tilde{\mathbf{g}}(\mathbf{y})\| \leq c\beta\|\mathbf{x} - \mathbf{y}\|$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}_0$.*

- *$\|\tilde{\mathbf{g}}(\mathbf{x})\| \leq cG$ for all $\mathbf{x} \in \mathcal{X}_0$.*

- *$\tilde{g}_k(\hat{\mathbf{x}}) \leq -c\epsilon$ for all $k \in \{1, 2, \ldots, m\}$ where $\hat{\mathbf{x}}$ is defined in Assumption 6.2.*

Now consider the following algorithm described in Algorithm 6.1. This algorithm chooses $\mathbf{x}(t+1)$ as the decision for round $t+1$ based on $f^t(\cdot)$ without knowing the cost function $f^{t+1}(\cdot)$. The remainder of this chapter shows that if the parameters c and α are chosen to satisfy $c = T^{1/4}$ and $\alpha = \frac{1}{2}(\beta^2+1)\sqrt{T}$, then Algorithm 6.1 achieves an $O(\sqrt{T})$ regret bound with finite constraint violations.

This algorithm introduces a virtual queue vector for constraint functions. The update equation of this virtual queue vector is similar to Algorithms 3.1 and 3.2 developed in Chapter 3 for deterministic convex programs (with a fixed and known objective function) and the main difference is that scaled versions of the constraint functions, rather than the original constraint functions, are used in the virtual queue update equation. In fact, scaling the constraint functions with a factor $c = T^{1/4}$ is the key to achieve finite constraint violations and Algorithm 6.1 can only achieve $O(\sqrt{T})$ constraint violations without the scaling factor c. The update for $\mathbf{x}(t+1)$ is different from the primal update in Algorithm 3.1 but is closely related to the primal update in

Algorithm 6.1 New Algorithm for Online Convex Optimization with Long Term Constraints

Let $c, \alpha > 0$ be constant parameters. Define function $\tilde{\mathbf{g}}(\mathbf{x}) = c\mathbf{g}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}_0$. Choose any $\mathbf{x}(0) \in \mathcal{X}_0$. Initialize $Q_k(0) = \max\{-\tilde{g}_k(\mathbf{x}(0)), 0\}, \forall k \in \{1, 2, \ldots, m\}$. At the end of each round $t \in \{0, 1, 2, 3, \ldots\}$, observe $f^t(\cdot)$ and do the following:

- Choose $\mathbf{x}(t+1)$ that solves

$$\min_{\mathbf{x} \in \mathcal{X}_0} \left\{ [\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}} [\mathbf{x} - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}} \tilde{\mathbf{g}}(\mathbf{x}) + \alpha \|\mathbf{x} - \mathbf{x}(t)\|^2 \right\}$$

  as the decision for the next round $t+1$, where $\nabla f^t(\mathbf{x}(t))$ is the gradient of $f^t(\mathbf{x})$ at point $\mathbf{x} = \mathbf{x}(t)$.

- Update virtual queue vector $\mathbf{Q}(t+1)$ via

$$Q_k(t+1) = \max\left\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\right\}, \forall k \in \{1, 2, \ldots, m\}.$$

Algorithm 3.2. In fact, if $\mathbf{g}(\mathbf{x})$ are linear functions, Lemma 6.2 shows that the update of $\mathbf{x}(t)$ in Algorithm 6.1 can also be implemented by a convex projection involving the subgradient of that scaled constraint functions.

Because of the $\|\mathbf{x} - \mathbf{x}(t)\|^2$ term, the choice of $\mathbf{x}(t+1)$ in Algorithm 6.1 involves minimization of a strongly convex function. If the constraint functions $\mathbf{g}(\mathbf{x})$ are separable (or equivalently, $\tilde{\mathbf{g}}(\mathbf{x})$ are separable) with respect to components or blocks of $\mathbf{x}$, e.g., $\mathbf{g}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$, then the primal updates for $\mathbf{x}(t+1)$ can be decomposed into several smaller independent subproblems, each of which only involves a component or block of $\mathbf{x}(t+1)$. The next lemma further shows that the update of $\mathbf{x}(t+1)$ follows a simple gradient update in the case when $\mathbf{g}(\mathbf{x})$ is linear.

Lemma 6.2. *If $\mathbf{g}(\mathbf{x})$ is linear, then the update of $\mathbf{x}(t+1)$ at each round in Algorithm 6.1 is given by*

$$\mathbf{x}(t+1) = \mathcal{P}_{\mathcal{X}_0}\left[\mathbf{x}(t) - \frac{1}{2\alpha}\mathbf{d}(t)\right]$$

where $\mathbf{d}(t) = \nabla f^t(\mathbf{x}(t)) + \sum_{k=1}^m [Q_k(t) + \tilde{g}_k(\mathbf{x}(t))] \nabla \tilde{g}_k(\mathbf{x}(t))$.

Proof. Fix $t \geq \{0, 1, \ldots\}$. Note that $\mathbf{d}(t)$ is a constant vector in the update of $\mathbf{x}(t+1)$. The projection operator can be interpreted as an optimization problem as follows:

$$
\begin{aligned}
& \mathbf{x}(t+1) = \mathcal{P}_{\mathcal{X}_0}\left[\mathbf{x}(t) - \frac{1}{2\alpha}\mathbf{d}(t)\right] \\
\overset{(a)}{\Leftrightarrow} \quad & \mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}_0}\left[\left\|\mathbf{x} - [\mathbf{x}(t) - \frac{1}{2\alpha}\mathbf{d}(t)]\right\|^2\right] \\
\Leftrightarrow \quad & \mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}_0}\left[\|\mathbf{x} - \mathbf{x}(t)\|^2 + \frac{1}{\alpha}\mathbf{d}^{\mathsf{T}}(t)[\mathbf{x} - \mathbf{x}(t)] + \frac{1}{4\alpha^2}\|\mathbf{d}(t)\|^2\right] \\
\overset{(b)}{\Leftrightarrow} \quad & \mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}_0}\left[\sum_{k=1}^{m}[Q_k(t+1) + \tilde{g}_k(\mathbf{x}(t))]\tilde{g}_k(\mathbf{x}(t)) + \mathbf{d}^{\mathsf{T}}(t)[\mathbf{x} - \mathbf{x}(t)] + \alpha\|\mathbf{x} - \mathbf{x}(t)\|^2\right] \\
\overset{(c)}{\Leftrightarrow} \quad & \mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}_0}\Big[[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + \sum_{k=1}^{m}[Q_k(t) + \tilde{g}_k(\mathbf{x}(t))]\tilde{g}_k(\mathbf{x}(t)) \\
& \qquad\qquad + \sum_{k=1}^{m}[Q_k(t) + \tilde{g}_k(\mathbf{x}(t))][\nabla\tilde{g}_k(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + \alpha\|\mathbf{x} - \mathbf{x}(t)\|^2\Big] \\
\overset{(d)}{\Leftrightarrow} \quad & \mathbf{x}(t+1) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}_0}\left[[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}) + \alpha\|\mathbf{x} - \mathbf{x}(t)\|^2\right]
\end{aligned}
$$

where (a) follows from the definition of the projection onto a convex set; (b) follows from the fact the minimizing solution does not change when we remove constant term $\frac{1}{4\alpha^2}\|\mathbf{d}(t)\|^2$, multiply positive constant α and add constant term $\sum_{k=1}^{m}[Q_k(t)+\tilde{g}_k(\mathbf{x}(t))]\tilde{g}_k(\mathbf{x}(t))$ in the objective function; (c) follows from the definition of $\mathbf{d}(t)$; and (d) follows from the identity $[\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}) = \sum_{k=1}^{m}[Q_k(t) + \tilde{g}_k(\mathbf{x}(t))]\tilde{g}_k(\mathbf{x}(t)) + \sum_{k=1}^{m}[Q_k(t) + \tilde{g}_k(\mathbf{x}(t))]\nabla\tilde{g}_k(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)]$ for any $\mathbf{x} \in \mathbb{R}^n$, which further follows from the linearity of $\tilde{\mathbf{g}}(\mathbf{x})$. □

6.2 Regret and Constraint Violation Analysis

This section analyzes the regret and constraint violations of Algorithm 6.1 for online convex optimization with long term constraints under Assumptions 6.1-6.2.

6.2.1 Properties of the Virtual Queues and the Drift

Lemma 6.3. *In Algorithm 6.1, we have*

1. *At each round $t \in \{0, 1, 2, \ldots\}$, $Q_k(t) \geq 0$ for all $k \in \{1, 2, \ldots, m\}$.*
2. *At each round $t \in \{0, 1, 2, \ldots\}$, $Q_k(t) + \tilde{g}_k(\mathbf{x}(t)) \geq 0$ for all $k \in \{1, 2 \ldots, m\}$.*
3. *At round $t = 0$, $\|\mathbf{Q}(0)\|^2 \leq \|\tilde{\mathbf{g}}(\mathbf{x}(0))\|^2$. At each round $t \in \{1, 2, \ldots\}$, $\|\mathbf{Q}(t)\|^2 \geq \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2$.*

4. *At each round* $t \in \{0, 1, 2, \ldots\}$, $\|\mathbf{Q}(t+1)\| \leq \|\mathbf{Q}(t)\| + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|$.

Proof. The proof of the first 3 parts is essentially identical to the proof of Lemma 3.1.

1. Fix $k \in \{1, 2, \ldots, m\}$. The proof is by induction. Note that $Q_k(0) \geq 0$ by initialization. Assume $Q_k(t) \geq 0$ for some $t \in \{0, 1, 2, \ldots\}$. We now prove $Q_k(t+1) \geq 0$. If $\tilde{g}_k(\mathbf{x}(t+1)) \geq 0$, the virtual queue update equation of Algorithm 6.1 gives:

$$Q_k(t+1) = \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \geq Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1)) \geq 0.$$

   On the other hand, if $\tilde{g}_k(\mathbf{x}(t+1)) < 0$, then $Q_k(t+1) = \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \geq -\tilde{g}_k(\mathbf{x}(t+1)) > 0$. Thus, in both cases we have $Q_k(t+1) \geq 0$.

2. Fix $k \in \{1, 2, \ldots, m\}$. Note that $Q_k(0) + \tilde{g}_k(\mathbf{x}(0)) \geq 0$ by the initialization rule of $Q_k(0)$. Fix $t \in \{0, 1, \ldots\}$. By the virtual queue update equation, we have $Q_k(t+1) = \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \geq -\tilde{g}_k(\mathbf{x}(t+1))$, which implies that $Q_k(t+1) + \tilde{g}_k(\mathbf{x}(t+1)) \geq 0$.

3. Note that $\|\mathbf{Q}(0)\|^2 \leq \|\tilde{\mathbf{g}}(\mathbf{x}(0))\|^2$ follows by the initialization rule of $Q_k(0)$. Fix $t \in \{0, 1, \ldots\}$ and $k \in \{1, 2, \ldots, m\}$. If $\tilde{g}_k(\mathbf{x}(t+1)) \geq 0$, then

$$\begin{aligned} Q_k(t+1) &= \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \\ &\geq Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1)) \\ &\overset{(a)}{\geq} \tilde{g}_k(\mathbf{x}(t+1)) = |\tilde{g}_k(\mathbf{x}(t+1))|, \end{aligned}$$

   where (a) follows from part 1. On the other hand, if $\tilde{g}_k(\mathbf{x}(t+1)) < 0$, then $Q_k(t+1) = \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \geq -\tilde{g}_k(\mathbf{x}(t+1)) = |\tilde{g}_k(\mathbf{x}(t+1))|$. Thus, in both cases, we have $Q_k(t+1) \geq |\tilde{g}_k(\mathbf{x}(t+1))|$. Squaring both sides and summing over $k \in \{1, 2, \ldots, m\}$ yields $\|\mathbf{Q}(t+1)\|^2 \geq \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2$. This holds for all $t \in \{0, 1, \ldots\}$. Thus, we have $\|\mathbf{Q}(t)\|^2 \geq \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2$ for all $t \in \{1, 2, \ldots\}$.

4. Fix $t \in \{0, 1, \ldots\}$. Define vector $\mathbf{h} = [h_1, \ldots, h_m]^{\mathsf{T}}$ by $h_k = |\tilde{g}_k(\mathbf{x}(t+1))|, \forall k \in \{1, 2, \ldots, m\}$. Note that $\|\mathbf{h}\| = \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|$. For any $k \in \{1, 2, \ldots, m\}$, by the virtual update equation

we have

$$\begin{aligned} Q_k(t+1) &= \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \\ &\leq |Q_k(t)| + |\tilde{g}_k(\mathbf{x}(t+1))| \\ &= Q_k(t) + h_k. \end{aligned}$$

Squaring both sides and summing over $k \in \{1, 2, \ldots, m\}$ yields $\|\mathbf{Q}(t+1)\|^2 \leq \|\mathbf{Q}(t) + \mathbf{h}\|^2$, which is equivalent to $\|\mathbf{Q}(t+1)\| \leq \|\mathbf{Q}(t) + \mathbf{h}\|$. Finally, by the triangle inequality $\|\mathbf{Q}(t) + \mathbf{h}\| \leq \|\mathbf{Q}(t)\| + \|\mathbf{h}\|$ and recalling that $\|\mathbf{h}\| = \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|$, we have $\|\mathbf{Q}(t+1)\| \leq \|\mathbf{Q}(t)\| + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|$.

□

Lemma 6.4. *Let $\mathbf{Q}(t), t \in \{0, 1, \ldots\}$ be the sequence generated by Algorithm 6.1. For any $T \geq 1$, we have*

$$\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq \frac{1}{c} Q_k(T), \forall k \in \{1, 2, \ldots, m\}.$$

Proof. Fix $k \in \{1, 2, \ldots, m\}$ and $T \geq 1$. For any $t \in \{0, 1, \ldots, T-1\}$, the update rule of Algorithm 6.1 gives:

$$Q_k(t+1) = \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\} \geq Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1)).$$

Hence, $\tilde{g}_k(\mathbf{x}(t+1)) \leq Q_k(t+1) - Q_k(t)$. Summing over $t \in \{0, \ldots, T-1\}$ yields

$$\sum_{t=1}^{T} \tilde{g}_k(\mathbf{x}(t)) = \sum_{t=0}^{T-1} \tilde{g}_k(\mathbf{x}(t+1)) \leq Q_k(T) - Q_k(0) \overset{(a)}{\leq} Q_k(T)$$

where (a) follows from the fact $Q_k(0) \geq 0$, i.e., part 1 in Lemma 6.3. This lemma follows by recalling that $\tilde{g}_k(\mathbf{x}) = c g_k(\mathbf{x})$. □

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

$$\Delta(t) = L(t+1) - L(t) = \frac{1}{2}[\|\mathbf{Q}(t+1)\|^2 - \|\mathbf{Q}(t)\|^2]. \tag{6.5}$$

Lemma 6.5. *At each round $t \in \{0, 1, 2, \ldots\}$ in Algorithm 6.1, an upper bound of the Lyapunov drift is given by*

$$\Delta(t) \leq [\mathbf{Q}(t)]^{\mathsf{T}} \tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2. \tag{6.6}$$

Proof. The virtual queue update equations $Q_k(t+1) = \max\{-\tilde{g}_k(\mathbf{x}(t+1)), Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1))\}, \forall k \in \{1, 2, \ldots, m\}$ can be rewritten as

$$Q_k(t+1) = Q_k(t) + h_k(\mathbf{x}(t+1)), \forall k \in \{1, 2, \ldots, m\}, \tag{6.7}$$

where

$$h_k(\mathbf{x}(t)) = \begin{cases} \tilde{g}_k(\mathbf{x}(t+1)), & \text{if } Q_k(t) + \tilde{g}_k(\mathbf{x}(t+1)) \geq -\tilde{g}_k(\mathbf{x}(t+1)) \\ -Q_k(t) - \tilde{g}_k(\mathbf{x}(t+1)), & \text{else} \end{cases} \quad \forall k.$$

Fix $k \in \{1, 2, \ldots, m\}$. Squaring both sides of (6.7) and dividing by factor 2 yield:

$$\begin{aligned}
&\frac{1}{2}[Q_k(t+1)]^2 \\
=&\frac{1}{2}[Q_k(t)]^2 + \frac{1}{2}[h_k(\mathbf{x}(t+1))]^2 + Q_k(t)h_k(\mathbf{x}(t+1)) \\
=&\frac{1}{2}[Q_k(t)]^2 + \frac{1}{2}[h_k(\mathbf{x}(t+1))]^2 + Q_k(t)\tilde{g}_k(\mathbf{x}(t+1)) + Q_k(t)[h_k(\mathbf{x}(t+1)) - \tilde{g}_k(\mathbf{x}(t+1))] \\
\overset{(a)}{=}&\frac{1}{2}[Q_k(t)]^2 + \frac{1}{2}[h_k(\mathbf{x}(t+1))]^2 + Q_k(t)\tilde{g}_k(\mathbf{x}(t+1)) \\
&- [h_k(\mathbf{x}(t+1)) + \tilde{g}_k(\mathbf{x}(t+1))][h_k(\mathbf{x}(t)) - \tilde{g}_k(\mathbf{x}(t+1))] \\
=&\frac{1}{2}[Q_k(t)]^2 - \frac{1}{2}[h_k(\mathbf{x}(t+1))]^2 + Q_k(t)\tilde{g}_k(\mathbf{x}(t+1)) + [\tilde{g}_k(\mathbf{x}(t+1))]^2 \\
\leq&\frac{1}{2}[Q_k(t)]^2 + Q_k(t)\tilde{g}_k(\mathbf{x}(t+1)) + [\tilde{g}_k(\mathbf{x}(t+1))]^2,
\end{aligned}$$

where (a) follows from the fact that $Q_k(t)[h_k(\mathbf{x}(t+1)) - \tilde{g}_k(\mathbf{x}(t+1))] = -[h_k(\mathbf{x}(t+1)) + \tilde{g}_k(\mathbf{x}(t+1))] \cdot [h_k(\mathbf{x}(t+1)) - \tilde{g}_k(\mathbf{x}(t+1))]$, which can be shown by considering $h_k(\mathbf{x}(t+1)) = \tilde{g}_k(\mathbf{x}(t+1))$ and $h_k(\mathbf{x}(t+1)) \neq \tilde{g}_k(\mathbf{x}(t+1))$. Summing over $k \in \{1, 2, \ldots, m\}$ yields

$$\frac{1}{2}\|\mathbf{Q}(t+1)\|^2 \leq \frac{1}{2}\|\mathbf{Q}(t)\|^2 + [\mathbf{Q}(t)]^{\mathsf{T}} \tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2.$$

Rearranging the terms yields the desired result. □

6.2.2 An Upper Bound of the Drift-Plus-Penalty Expression

Lemma 6.6. *Consider online convex optimization with long term constraints under Assumption 6.1. Let* $\mathbf{x}^* \in \mathcal{X}_0$ *be any fixed solution that satisfies* $\mathbf{g}(\mathbf{x}^*) \leq \mathbf{0}$*, e.g.,* $\mathbf{x}^* = \text{argmin}_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f^t(\mathbf{x})$*. Let* $c > 0$ *and* $\eta > 0$ *be arbitrary. If* $\alpha \geq \frac{1}{2}(c^2\beta^2 + \eta)$ *in Algorithm 6.1, then for all* $t \geq 1$*, we have*

$$\begin{aligned}&\Delta(t) + f^t(\mathbf{x}(t))\\ \leq& f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + \frac{1}{2}[\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 - \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2] + \frac{1}{2\eta}D^2\end{aligned}$$

where β *and* D *are constants defined in Assumption 6.1.*

Proof. Fix $t \geq 1$. Note that part 2 of Lemma 6.3 implies that $\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))$ is component-wise nonnegative. Hence, $[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x})$ is a convex function with respect to $\mathbf{x}$. Since $\alpha\|\mathbf{x} - \mathbf{x}(t)\|^2$ is strongly convex with respect to $\mathbf{x}$ with modulus 2α, it follows that

$$[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x} - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}) + \alpha\|\mathbf{x} - \mathbf{x}(t)\|^2$$

is strongly convex with respect to $\mathbf{x}$ with modulus 2α.

Since $\mathbf{x}(t+1)$ is chosen to minimize the above strongly convex function, by Corollary 1.2, we have

$$\begin{aligned}&[\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}(t+1) - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \alpha\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2\\ \leq& [\nabla f^t(\mathbf{x}(t))]^{\mathsf{T}}[\mathbf{x}^* - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(t+1)\|^2.\end{aligned}$$

Adding $f^t(\mathbf{x}(t))$ on both sides yields

$$\begin{aligned}
&f^t(\mathbf{x}(t)) + [\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \alpha\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
\leq & f^t(\mathbf{x}(t)) + [\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}^* - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(t)\|^2 \\
&- \alpha\|\mathbf{x}^* - \mathbf{x}(t+1)\|^2 \\
\overset{(a)}{\leq} & f^t(\mathbf{x}^*) + \underbrace{[\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}^*)}_{\leq 0} + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] \\
\overset{(b)}{\leq} & f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2],
\end{aligned}$$

where (a) follows from the convexity of function $f^t(\mathbf{x})$; and (b) follows by using the fact that $\tilde{g}_k(\mathbf{x}^*) \leq 0$ and $Q_k(t) + \tilde{g}_k(\mathbf{x}(t)) \geq 0$ (i.e., part 2 in Lemma 6.3) for all $k \in \{1, 2, \ldots, m\}$ to eliminate the term marked by an underbrace.

Rearranging terms yields

$$\begin{aligned}
&f^t(\mathbf{x}(t)) + [\mathbf{Q}(t)]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) \\
\leq & f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] - \alpha\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
&- [\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] - [\tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)).
\end{aligned} \tag{6.8}$$

For any $\eta > 0$, we have

$$\begin{aligned}
-[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] &\overset{(a)}{\leq} \|\nabla f^t(\mathbf{x}(t))\|\|\mathbf{x}(t+1) - \mathbf{x}(t)\| \\
&= [\frac{1}{\sqrt{\eta}}\|\nabla f^t(\mathbf{x}(t))\|][\sqrt{\eta}\|\mathbf{x}(t+1) - \mathbf{x}(t)\|] \\
&\overset{(b)}{\leq} \frac{1}{2\eta}\|\nabla f^t(\mathbf{x}(t))\|^2 + \frac{1}{2}\eta\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
&\overset{(c)}{\leq} \frac{1}{2\eta}D^2 + \frac{1}{2}\eta\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2,
\end{aligned} \tag{6.9}$$

where (a) follows from the Cauchy-Schwarz inequality; (b) follows from the basic inequality $ab \leq \frac{1}{2}(a^2 + b^2), \forall a, b \in \mathbb{R}$; and (c) follows from Assumption 6.1.

Recall that $\mathbf{u}_1^\mathsf{T}\mathbf{u}_2 = \frac{1}{2}[\|\mathbf{u}_1\|^2 + \|\mathbf{u}_2\|^2 - \|\mathbf{u}_1 - \mathbf{u}_2\|^2]$ for any $\mathbf{u}_1, \mathbf{u}_2 \in \mathbb{R}^m$. Thus, we have

$$[\tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) = \frac{1}{2}\big[\|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2 + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 - \|\tilde{\mathbf{g}}(\mathbf{x}(t+1)) - \tilde{\mathbf{g}}(\mathbf{x}(t))\|^2\big]. \tag{6.10}$$

Substituting (6.9) and (6.10) into (6.8) yields

$$
\begin{aligned}
&f^t(\mathbf{x}(t)) + [\mathbf{Q}(t)]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) \\
\leq& f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + (\frac{1}{2}\eta - \alpha)\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 + \frac{1}{2\eta}D^2 \\
&+ \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t+1)) - \tilde{\mathbf{g}}(\mathbf{x}(t))\|^2 - \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2 - \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 \\
\overset{(a)}{\leq}& f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + (\frac{1}{2}c^2\beta^2 + \frac{1}{2}\eta - \alpha)\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 + \frac{1}{2\eta}D^2 \\
&- \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2 - \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 \\
\overset{(b)}{\leq}& f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + \frac{1}{2\eta}D^2 - \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2 - \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2,
\end{aligned}
\tag{6.11}
$$

where (a) follows because $\|\tilde{\mathbf{g}}(\mathbf{x}(t+1)) - \tilde{\mathbf{g}}(\mathbf{x}(t))\| \leq c\beta\|\mathbf{x}(t+1) - \mathbf{x}(t)\|$, which further follows from Lemma 6.1; and (b) follows because $\alpha \geq \frac{1}{2}(c^2\beta^2 + \eta)$.

By Lemma 6.5, we have

$$
\Delta(t) \leq [\mathbf{Q}(t)]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2. \tag{6.12}
$$

Summing (6.11) and (6.12) together yields

$$
\begin{aligned}
&\Delta(t) + f^t(\mathbf{x}(t)) \\
\leq& f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + \frac{1}{2}[\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 - \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2] + \frac{1}{2\eta}D^2.
\end{aligned}
$$

□

6.2.3 Regret Analysis

Theorem 6.1. *Consider online convex optimization with long term constraints under Assumption 6.1. Let $\mathbf{x}^* \in \mathcal{X}_0$ be any fixed solution that satisfies $\mathbf{g}(\mathbf{x}^*) \leq \mathbf{0}$, e.g., $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\{\sum_{t=1}^{T} f^t(\mathbf{x})\}$.*

1. *Let $c > 0$ and $\eta > 0$ be arbitrary. If $\alpha \geq \frac{1}{2}(c^2\beta^2 + \eta)$ in Algorithm 6.1, then for all $T \geq 1$, we have*

$$
\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha R^2 + 2c^2G^2 + \frac{1}{2\eta}D^2T.
$$

2. If $c = T^{1/4}, \eta = \sqrt{T}$ and $\alpha = \frac{1}{2}(\beta^2 + 1)\sqrt{T}$ in Algorithm 6.1, then for all $T \geq 1$, we have

$$\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq \sum_{t=1}^{T} f^t(\mathbf{x}^*) + O(\sqrt{T}).$$

Proof. Fix $T \geq 1$. Since $\alpha \geq \frac{1}{2}(c^2\beta^2 + \eta)$, by Lemma 6.6, for all $t \in \{1, 2, \ldots, T\}$, we have

$$\begin{aligned}
&\Delta(t) + f^t(\mathbf{x}(t)) \\
\leq &f^t(\mathbf{x}^*) + \alpha[\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + \frac{1}{2}[\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 - \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2] + \frac{1}{2\eta}D^2.
\end{aligned}$$

Summing over $t \in \{1, 2, \ldots, T\}$ yields

$$\begin{aligned}
&\sum_{t=1}^{T} \Delta(t) + \sum_{t=1}^{T} f^t(\mathbf{x}(t)) \\
\leq &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha \sum_{t=1}^{T} [\|\mathbf{x}^* - \mathbf{x}(t)\|^2 - \|\mathbf{x}^* - \mathbf{x}(t+1)\|^2] + \frac{1}{2} \sum_{t=1}^{T} [\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 - \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|^2] \\
&+ \frac{1}{2\eta}D^2T.
\end{aligned}$$

Recalling that $\Delta(t) = L(t+1) - L(t)$ and simplifying summations yields

$$\begin{aligned}
&L(T+1) - L(1) + \sum_{t=1}^{T} f^t(\mathbf{x}(t)) \\
\leq &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(1)\|^2 - \alpha\|\mathbf{x}^* - \mathbf{x}(T+1)\|^2 + \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(T+1))\|^2 - \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(1))\|^2 + \frac{1}{2\eta}D^2T \\
\leq &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(1)\|^2 + \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(T+1))\|^2 + \frac{1}{2\eta}D^2T.
\end{aligned}$$

Rearranging terms yields

$$
\begin{aligned}
&\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \\
\leq &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(1)\|^2 + \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(T+1))\|^2 + L(1) - L(T+1) + \frac{1}{2\eta}D^2T \\
\overset{(a)}{=} &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(1)\|^2 + \frac{1}{2}\|\tilde{\mathbf{g}}(\mathbf{x}(T+1))\|^2 + \frac{1}{2}\|\mathbf{Q}(1)\|^2 - \frac{1}{2}\|\mathbf{Q}(T+1)\|^2 + \frac{1}{2\eta}D^2T \\
\overset{(b)}{\leq} &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha\|\mathbf{x}^* - \mathbf{x}(1)\|^2 + \frac{1}{2}\|\mathbf{Q}(1)\|^2 + \frac{1}{2\eta}D^2T \\
\overset{(c)}{\leq} &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \alpha R^2 + 2c^2G^2 + \frac{1}{2\eta}D^2T,
\end{aligned}
$$

where (a) follows form the definition that $L(1) = \frac{1}{2}\|\mathbf{Q}(1)\|^2$ and $L(T+1) = \frac{1}{2}\|\mathbf{Q}(T+1)\|^2$; (b) follows because $\|\mathbf{Q}(T+1)\|^2 \geq \|\tilde{\mathbf{g}}(\mathbf{x}(T+1))\|^2$ by part 3 in Lemma 6.3; and (c) follows because $\|\mathbf{x}^* - \mathbf{x}(1)\| \leq R$ by Assumption 6.1 and $\|\mathbf{Q}(1)\| \leq \|\mathbf{Q}(0)\| + \|\tilde{\mathbf{g}}(\mathbf{x}(1))\| \leq \|\tilde{\mathbf{g}}(\mathbf{x}(0))\| + \|\tilde{\mathbf{g}}(\mathbf{x}(1))\| \leq 2cG$, which further follows from part 3 and part 4 in Lemma 6.3 and Lemma 6.1.

Thus, the first part of this theorem follows. Note that if we let $c = T^{1/4}$ and $\eta = \sqrt{T}$, then $\alpha = \frac{1}{2}(\beta^2+1)\sqrt{T} \geq \frac{1}{2}(c^2\beta^2 + \eta)$. The second part of this theorem follows by substituting $c = T^{1/4}, \eta = \sqrt{T}$ and $\alpha = \frac{1}{2}(\beta^2+1)\sqrt{T}$ into the first part of this theorem. Thus, we have

$$
\begin{aligned}
\sum_{t=1}^{T} f^t(\mathbf{x}(t)) \leq &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + \frac{1}{2}(\beta^2+1)R^2\sqrt{T} + 2G^2\sqrt{T} + \frac{1}{2}D^2\sqrt{T} \\
= &\sum_{t=1}^{T} f^t(\mathbf{x}^*) + O(\sqrt{T}).
\end{aligned}
$$

□

6.2.4 An Upper Bound of the Virtual Queue Vector

It remains to establish a bound on constraint violations. Lemma 6.4 implies that this can be done by bounding $\|\mathbf{Q}(t)\|$.

Lemma 6.7. *Consider online convex optimization with long term constraints under Assumptions*

6.1-6.2. At each round $t \in \{1, 2, \dots, \}$ in Algorithm 6.1, if

$$\|\mathbf{Q}(t)\| > cG + \frac{\alpha R^2 + DR + 2c^2G^2}{c\epsilon}$$

where D, G and R are constants defined in Assumption 6.1 and ϵ is the constant defined in Assumption 6.2, then

$$\|\mathbf{Q}(t+1)\| < \|\mathbf{Q}(t)\|.$$

Proof. Let $\hat{\mathbf{x}} \in \mathcal{X}_0$ and $\epsilon > 0$ be defined in Assumption 6.2. Fix $t \geq 0$. Since $\mathbf{x}(t+1)$ is chosen to minimize $[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x} - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}) + \alpha\|\mathbf{x} - \mathbf{x}(t)\|^2$, we have

$$\begin{aligned}
&[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \alpha\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
\leq &[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(t)] + [\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\hat{\mathbf{x}}) + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2 \\
\overset{(a)}{\leq} &[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(t)] - c\epsilon\sum_{k=1}^{m}[Q_k(t) + \tilde{g}_k(\mathbf{x}(t))] + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2 \\
\overset{(b)}{\leq} &[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(t)] - c\epsilon\|\mathbf{Q}(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))\| + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2 \\
\overset{(c)}{\leq} &[\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(t)] - c\epsilon\big[\|\mathbf{Q}(t)\| - \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|\big] + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2
\end{aligned}$$

where (a) follows because $\tilde{g}_k(\hat{\mathbf{x}}) \leq -c\epsilon, \forall k \in \{1, 2, \dots, m\}$ by Lemma 6.1 and $Q_k(t) + \tilde{g}_k(\mathbf{x}(t)) \geq 0, \forall k \in \{1, 2, \dots, m\}$ by part 2 in Lemma 6.3; (b) follows from the basic inequality $\sum_{i=1}^m a_i \geq \sqrt{\sum_{i=1}^m a_i^2}$ for any nonnegative vector $\mathbf{a} \geq \mathbf{0}$; and (c) follows from the triangle inequality $\|\mathbf{x}-\mathbf{y}\| \geq \|\mathbf{x}\| - \|\mathbf{y}\|, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^m$.

Rearranging terms yields

$$\begin{aligned}
&[\mathbf{Q}(t)]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) \\
\leq &- c\epsilon\|\mathbf{Q}(t)\| + c\epsilon\|\tilde{\mathbf{g}}(\mathbf{x}(t))\| + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2 - \alpha\|\mathbf{x}(t+1) - \mathbf{x}(t)\|^2 \\
&+ [\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(t)] - [\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\mathbf{x}(t+1) - \mathbf{x}(t)] - [\tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) \\
\leq &- c\epsilon\|\mathbf{Q}(t)\| + c\epsilon\|\tilde{\mathbf{g}}(\mathbf{x}(t))\| + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2 + [\nabla f^t(\mathbf{x}(t))]^\mathsf{T}[\hat{\mathbf{x}} - \mathbf{x}(t+1)] - [\tilde{\mathbf{g}}(\mathbf{x}(t))]^\mathsf{T}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) \\
\overset{(a)}{\leq} &- c\epsilon\|\mathbf{Q}(t)\| + c\epsilon\|\tilde{\mathbf{g}}(\mathbf{x}(t))\| + \alpha\|\hat{\mathbf{x}} - \mathbf{x}(t)\|^2 + \|\nabla f^t(\mathbf{x}(t))\|\|\hat{\mathbf{x}} - \mathbf{x}(t+1)\| \\
&+ \|\tilde{\mathbf{g}}(\mathbf{x}(t))\|\|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\| \\
\overset{(b)}{\leq} &- c\epsilon\|\mathbf{Q}(t)\| + c^2\epsilon G + \alpha R^2 + DR + c^2G^2, \qquad (6.13)
\end{aligned}$$

where (a) follows from the Cauchy-Schwarz inequality and (b) follows from Assumption 6.1 and Lemma 6.1.

By Lemma 6.5, we have

$$\begin{aligned}\Delta(t) \leq & [\mathbf{Q}(t)]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) + \|\tilde{\mathbf{g}}(\mathbf{x}(t+1))\|^2 \\ \overset{(a)}{\leq} & [\mathbf{Q}(t)]^{\mathsf{T}}\tilde{\mathbf{g}}(\mathbf{x}(t+1)) + c^2G^2 \\ \overset{(b)}{\leq} & -c\epsilon\|\mathbf{Q}(t)\| + c^2\epsilon G + \alpha R^2 + DR + 2c^2G^2,\end{aligned}$$

where (a) follows from Lemma 6.1 and (b) follows from (6.13).

Thus, if $\|\mathbf{Q}(t)\| > cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$, then $\Delta(t) < 0$. That is, $\|\mathbf{Q}(t+1)\| < \|\mathbf{Q}(t)\|$. □

Corollary 6.1. *Consider online convex optimization with long term constraints under Assumptions 6.1-6.2. At each round $t \in \{1, 2, \dots, \}$ in Algorithm 6.1,*

$$\|\mathbf{Q}(t)\| \leq 2cG + \frac{\alpha R^2 + DR + 2c^2G^2}{c\epsilon},$$

where D, G and R are constants defined in Assumption 6.1 and $\epsilon > 0$ is defined in Assumption 6.2.

Proof. Note that $\|\mathbf{Q}(0)\| \overset{(a)}{\leq} \|\tilde{\mathbf{g}}(\mathbf{x}(0))\| \overset{(b)}{\leq} cG$ and $\|\mathbf{Q}(1)\| \overset{(a)}{\leq} \|\mathbf{Q}(0)\| + \|\tilde{\mathbf{g}}(\mathbf{x}(0))\| \overset{(b)}{\leq} 2cG \leq 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$, where (a) follows from Lemma 6.3 and (b) follows from Lemma 6.1. We need to show $\|\mathbf{Q}(t)\| \leq 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$ for all rounds $t \geq 2$. This can be proven by contradiction as follows:

Assume that $\|\mathbf{Q}(t)\| > 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$ happens at some round $t \geq 2$. Let τ be the first (smallest) round index at which this happens, i.e., $\|\mathbf{Q}(\tau)\| > 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$. Note that $\tau \geq 2$ since we know $\|\mathbf{Q}(1)\| \leq 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$. The definition of τ implies that $\|\mathbf{Q}(\tau-1)\| \leq 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$. Now consider the value of $\|\mathbf{Q}(\tau-1)\|$ in two cases.

- If $\|\mathbf{Q}(\tau-1)\| > cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$, then by Lemma 6.7, we must have $\|\mathbf{Q}(\tau)\| < \|\mathbf{Q}(\tau-1)\| \leq 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$. This contradicts the definition of τ.
- If $\|\mathbf{Q}(\tau-1)\| \leq cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$, then by part 4 in Lemma 6.3, we must have $\|\mathbf{Q}(\tau)\| \leq \|\mathbf{Q}(\tau-1)\| + \|\tilde{\mathbf{g}}(\mathbf{x}(\tau))\| \overset{(a)}{\leq} cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon} + cG = 2cG + \frac{\alpha R^2+DR+2c^2G^2}{c\epsilon}$, where (a) follows from Lemma 6.1. This also contradicts the definition of τ.

In both cases, we have a contradiction. Thus, $\|\mathbf{Q}(t)\| \leq 2cG + \frac{\alpha R^2 + DR + 2c^2G^2}{c\epsilon}$ for all round $t \geq 2$. □

6.2.5 Constraint Violation Analysis

Theorem 6.2. *Consider online convex optimization with long term constraints under Assumptions 6.1-6.2. Let D, β, G, R and ϵ be constants defined in Assumptions 6.1-6.2. The following are ensured by Algorithm 6.1:*

1. *For all $T \geq 1$, we have*

$$\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq 2G + \frac{\alpha R^2 + DR + 2c^2G^2}{c^2\epsilon}.$$

2. *If $c = T^{1/4}$ and $\alpha = \frac{1}{2}(\beta^2+1)\sqrt{T}$ in Algorithm 6.1, then for all $T \geq 1$, we have*

$$\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq 2G + \frac{\frac{1}{2}(\beta^2+1)R^2 + 2G^2 + DR}{\epsilon}, \forall k \in \{1, 2, \ldots, m\}.$$

Proof. Fix $T \geq 1$ and $k \in \{1, 2, \ldots, m\}$. By Lemma 6.4, we have

$$\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq \frac{1}{c} Q_k(T) \leq \frac{1}{c}\|Q_k(T)\| \overset{(a)}{\leq} \frac{2c}{c}G + \frac{\alpha R^2 + DR + 2c^2G^2}{c^2\epsilon},$$

where (a) follows from Corollary 6.1. Thus, the first part of this theorem follows.

The second part of this theorem follows by substituting $c = T^{1/4}$ and $\alpha = \frac{1}{2}(\beta^2+1)\sqrt{T}$ into the last inequality:

$$\begin{aligned}
\sum_{t=1}^{T} g_k(\mathbf{x}(t)) \leq & 2G + \frac{\frac{1}{2}(\beta^2+1)\sqrt{T}R^2 + DR + 2\sqrt{T}G^2}{\sqrt{T}\epsilon} \\
\leq & 2G + \frac{\frac{1}{2}(\beta^2+1)R^2 + 2G^2}{\epsilon} + \frac{DR}{\epsilon}T^{-1/2} \\
\leq & 2G + \frac{\frac{1}{2}(\beta^2+1)R^2 + 2G^2 + DR}{\epsilon}
\end{aligned}$$

□

6.2.6 Practical Implementations

The finite constraint violation bound proven in Theorem 6.2 is in terms of constants D, G, R and ϵ defined in Assumptions 6.1-6.2. However, the implementation of Algorithm 6.1 only requires the knowledge of β, which is known to us since the constraint function $\mathbf{g}(\mathbf{x})$ does not change. In contrast, the algorithms developed in [MJY12] and [JHA16] have parameters that must be chosen based on the knowledge of D, which is usually unknown and can be difficult to estimate in an online optimization scenario.

6.3 Extensions

This section extends the analysis in the previous section by considering intermediate and unknown time horizon T.

6.3.1 Intermediate Time Horizon T

Note that parts 1 of Theorems 6.1 and 6.2 hold for any T. For large T, choosing $c = T^{1/4}$ and $\alpha = \frac{1}{2}(\beta^2 + 1)\sqrt{T}$ yields the $O(\sqrt{T})$ regret bound and finite constraint violations as proven in parts 2 of both theorems. For intermediate T, the constant factor hidden in the $O(\sqrt{T})$ bound can be important and finite constraint violation bound can be relatively large. If parameters in Assumptions 6.1-6.2 are known, we can obtain the best regret and constraint violation bounds by choosing c and α as the solution to the following geometric program[1]:

$$
\begin{aligned}
\min_{\eta,c,\alpha,z} \quad & z \\
\text{s.t.} \quad & \alpha R^2 + 2c^2G^2 + \frac{1}{2\eta}D^2T \leq z, \\
& 2G + \frac{\alpha R^2 + DR + 2c^2G^2}{c^2\epsilon} \leq z, \\
& \frac{1}{2}(\beta^2 c^2 + \eta) \leq \alpha, \\
& \eta, c, \alpha, z > 0.
\end{aligned}
$$

[1] By dividing the first two constraints by z and dividing the third constraint by α on both sides, this geometric program can be written into the standard from of geometric programs. Geometric programs can be reformulated into convex programs and can be efficiently solved. See [BKVH07] for more discussions on geometric programs.

In certain applications, we can choose c and α to minimize the regret bound subject to the constraint violation guarantee by solving the following geometric program:

$$\begin{aligned} \min_{\eta,c,\alpha} \quad & \alpha R^2 + 2c^2G^2 + \frac{1}{2\eta}D^2T \\ \text{s.t.} \quad & 2G + \frac{\alpha R^2 + DR + 2c^2G^2}{c^2\epsilon} \leq z_0, \\ & \frac{1}{2}(\beta^2c^2 + \eta) \leq \alpha, \\ & \eta, c, \alpha > 0, \end{aligned}$$

where $z_0 > 0$ is a constant that specifies the maximum allowed constraint violation. Or alternatively, we can consider the problem of minimizing the constraint violation subject to the regret bound guarantee.

6.3.2 Unknown Time Horizon T

To achieve $O(\sqrt{T})$ regret and finite constraint violations, the parameters c and α in Algorithm 6.1 depend on the time horizon T. In the case when T is unknown, we can use the classical "doubling trick" to achieve $O(\sqrt{T})$ regret and $O(\log_2 T)$ constraint violations.

Suppose we have an online convex optimization algorithm $\mathcal{A}$ whose parameters depend on the time horizon. In the case when the time horizon T is unknown, the general doubling trick [CBL06, SS11] is described in Algorithm 6.2. It is known that the doubling trick can preserve the order of algorithm $\mathcal{A}$'s regret bound in the case when the time horizon T is unknown. The next theorem summarizes that by using the "doubling trick" for Algorithm 6.1 with unknown time horizon T, we can achieve $O(\sqrt{T})$ regret and $O(\log_2 T)$ constraint violations.

Algorithm 6.2 The Doubling Trick [CBL06, SS11]

- Let algorithm $\mathcal{A}$ be an algorithm whose parameters depend on the time horizon. Let $i = 1$.
- Repeat until we reach the end of the time horizon
  - Run algorithm $\mathcal{A}$ for 2^i rounds by using 2^i as the time horizon.
  - Let $i = i + 1$.

Theorem 6.3. *If the time horizon T is unknown, then applying Algorithm 6.1 with the "doubling trick" can yield $O(\sqrt{T})$ regret and $O(\log_2 T)$ constraint violations.*

Proof. Let T be the unknown time horizon. Define each iteration in the doubling trick as a period. Since the i-th period consists of 2^i rounds, we have in total $\lceil \log_2 T \rceil$ periods, where $\lceil x \rceil$ denote the smallest integer no less than x.

1. The proof of $O(\sqrt{T})$ regret is almost identical to the classical proof. By Theorem 6.1, there exists a constant C such that the regret in the i-th period is at most $C\sqrt{2^i}$. Thus, the total regret is at most

$$\begin{aligned}\sum_{i=1}^{\lceil \log_2 T \rceil} C\sqrt{2^i} =& C\frac{\sqrt{2}[1-\sqrt{2}^{\lceil \log_2 T \rceil}]}{1-\sqrt{2}} \\ =& \frac{\sqrt{2}C}{\sqrt{2}-1}[\sqrt{2}^{\lceil \log_2 T \rceil}-1] \\ \leq& \frac{\sqrt{2}C}{\sqrt{2}-1}\sqrt{2}^{1+\log_2 T} \\ \leq& \frac{2C}{\sqrt{2}-1}\sqrt{T}\end{aligned}$$

   Thus, the regret bound is $O(\sqrt{T})$ when using the "doubling trick".

2. The proof of $O(\log_2 T)$ constraint violations is simple. By Theorem 6.1, there exists a constant C such that the constraint violation in the i-th period is at most C. Since we have $\lceil \log_2 T \rceil$ periods, the total constraint violation is $C\lceil \log_2 T \rceil$.

□

6.4 Chapter Summary

This chapter considers online convex optimization with long term constraints, where functional constraints are only required to be satisfied in the long term. Prior algorithms in [MJY12] can achieve $O(\sqrt{T})$ regret and $O(T^{3/4})$ constraint violations for general problems and achieve $O(T^{2/3})$ bounds for both regret and constraint violations when the constraint set can be described by a finite number of linear constraints. A recent extension in [JHA16] can achieve $O(T^{\max\{\beta,1-\beta\}})$ regret and $O(T^{1-\beta/2})$ constraint violations where $\beta \in (0,1)$ in an algorithm parameter. Algorithm 5.1 developed in Chapter 5 can achieve $O(\sqrt{T})$ regret and $O(\sqrt{T})$ constraint violations. This chapter proposes a new algorithm that can achieve an $O(\sqrt{T})$ bound for regret and an $O(1)$ bound for constraint violations; and hence yields improved performance in

comparison to prior works [MJY12, JHA16] and our own Algorithm 5.1.

Chapter 7

Power Control for Energy Harvesting Devices with Outdated State Information

Energy harvesting can enable self-sustainable and perpetual wireless devices. By harvesting energy from the environment and storing it in a battery for future use, we can significantly improve energy efficiency and device lifetime. Harvested energy can come from solar, wind, vibrational, thermal, or even radio sources [PS05, SK11, UYE+15]. Energy harvesting has been identified as a key technology for wireless sensor networks [KHZS07], internet of things (IoT) [KMS+15], and 5G communication networks [HH15]. However, the development of harvesting algorithms is complex because the harvested energy is highly dynamic and the device environment and energy needs are also dynamic. Efficient algorithms should learn when to take energy from the battery to power device tasks that bring high utility, and when to save energy for future use.

There have been large amounts of work developing efficient power control policies to maximize the utility of energy harvesting devices. In the highly ideal case where the future system state (both the wireless channel sate and energy harvesting state) can be perfectly predicted, optimal power control strategies that maximize the throughput of wireless systems are considered in [YU12, TY12]. In a more realistic case with only the statistics and causal knowledge of the system state, power control policies based on Markov Decision Processes (MDP) are considered in [BGD13, MSZ13]. In the case when the statistical knowledge is unavailable but the current system state is observable, work [WWW+17] develops suboptimal power control policies based on approximation algorithms.

However, there is little work on the challenging scenario where neither the distribution information nor the system state information are known. In practice, the amount of harvested

energy on each slot is known to us only after it arrives and is stored into the battery. Further, the wireless environment is often unknown before the power action is chosen. For example, the wireless channel state in a communication link is measured at the receiver side and then reported back to the transmitter with a time delay. If the fading channel varies very fast, the channel state feedback received at the transmitter can be outdated. Another example is power control for sensor nodes that detect unknown targets where the state of targets is known only after the sensing action is performed.

In this chapter, we consider utility-optimal power control in an energy harvesting wireless device with outdated state information and unknown state distribution information. This problem setup is closely related to but different from the Lyapunov opportunistic power control considered in works [GGT10, HN13, UUNS11] with instantaneous wireless channel state information. The policies developed in [GGT10, HN13, UUNS11] are allowed to adapt their power actions to the instantaneous system states on each slot, which are unavailable in our problem setup. The problem setup in this chapter is also closely related to online convex optimization where control actions are performed without knowing instantaneous system states [Zin03, CBL06, SS11]. However, existing methods for online convex learning require the control actions to be chosen from a fixed set. This does not hold in our problem since the power to be used can only be drained from the battery whose backlog is time-varying and dependent on previous actions.

In Chapter 5, we extend the conventional online convex optimization (with a fixed known action set) to a more general setup with stochastic constraints. The stochastic constraints can be used to describe the uncertainty of energy harvesting and the fact that consumed energy is no more than the harvested energy in the long term. However, the stochastic constraint does not capture the fact that the used energy at any round must be no more than what is available in the battery and the fact that no more energy can be harvested when the battery is full. The algorithm developed in Chapter 5 for general online convex optimization with stochastic constraints only ensure that the stochastic constraint violations grow sublinearly in expectation and in high probability, which is not sufficient when used as a feasible power control algorithm for energy harvesting devices.

In this chapter, we develop a new power control algorithm for energy harvesting devices with outdated state information and show that this power control algorithm can achieve an $O(\epsilon)$ optimal utility by using a battery with capacity $O(1/\epsilon)$. The results in this chapter are originally

developed in our paper [YN18a].

7.1 Problem Formulation

Consider an energy harvesting wireless device that operates in normalized time slots $t \in \{1, 2, \ldots\}$. Let $\omega(t) = [e(t), \mathbf{s}(t)] \in \Omega$ represent the system state on each slot t, where

- $e(t)$ is the amount of harvested energy for slot t (for example, through solar, wind, radio signal, and so on).
- $\mathbf{s}(t)$ is the wireless device state on slot t (such as the vector of channel conditions over multiple subbands).
- Ω is the state space for all $\omega(t) = [e(t), \mathbf{s}(t)]$ states.

Assume $\{\omega(t)\}_{t=1}^{\infty}$ evolves in an independent and identically distributed (i.i.d.) manner according to an unknown distribution. Further, the state $\omega(t)$ is *unknown* to the device *until the end of slot* t. The device is powered by a finite-size battery. At the beginning of each slot $t \in \{1, 2, \ldots\}$, the device draws energy from the battery and allocates it as an n-dimensional power decision vector $\mathbf{p}(t) = [p_1(t), \ldots, p_n(t)]^{\mathsf{T}} \in \mathcal{P}$ where $\mathcal{P}$ is a compact convex set given by

$$\mathcal{P} = \{\mathbf{p} \in \mathbb{R}^n : \sum_{i=1}^{n} p_i \leq p^{\max}, p_i \geq 0, \forall i \in \{1, 2, \ldots, n\}\}.$$

Note that $p^{\max}$ is a given positive constant (restricted by hardware) and represents the maximum total power that can be used on each slot. The device receives a corresponding utility $U(\mathbf{p}(t); \omega(t))$. Since $\mathbf{p}(t)$ is chosen without knowledge of $\omega(t)$, the achieved utility is unknown until the end of slot t. For each $\omega \in \Omega$, the utility function $U(\mathbf{p}; \omega)$ is assumed to be continuous and concave over $\mathbf{p} \in \mathcal{P}$. An example is:

$$U(\mathbf{p}; \omega) = \sum_{i=1}^{n} \log(1 + p_i(t)s_i(t)) \tag{7.1}$$

where $\mathbf{s}(t) = [s_1(t), \ldots, s_n(t)]$ is the vector of (unknown) channel conditions over n orthogonal subbands available to the wireless device. In this example, $p_i(t)$ represents the amount of power invested over subband i in a rateless coding transmission scenario, and $U(p(t); \omega(t))$ is the total

throughput achieved on slot t. We focus on fast time-varying wireless channels, e.g., communication scenarios with high mobility transceivers, where $\mathbf{s}(t)$ known at the transmitter is outdated since $\mathbf{s}(t)$ must be measured at the receiver side and then reported back to the transmitter with a time delay.

7.1.1 Further Examples

The above formulation admits a variety of other useful application scenarios. For example, it can be used to treat power control in cognitive radio systems. Suppose an energy limited secondary user harvests energy and operates over licensed spectrum occupied by primary users. In this case, $\mathbf{s}(t) = [s_1(t), \ldots, s_n(t)]$ represents the channel activity of primary users over each subband. Since primary users are not controlled by the secondary user, $\mathbf{s}(t)$ is only known to the secondary user at the end of slot t.

Another application is a wireless sensor system. Consider an energy harvesting sensor node that collects information by detecting an unpredictable target. In this case, $\mathbf{s}(t)$ can be the state or action of the target on slot t. By using $\mathbf{p}(t)$ power for signaling and sensing, we receive utility $U(\mathbf{p}(t); \omega(t))$, which depends on state $\omega(t)$. For example, in a monitoring system, if the monitored target performs an action $\mathbf{s}(t)$ that we are not interested in, then the reward $U(\mathbf{p}(t); \omega(t))$ by using $\mathbf{p}(t)$ is small. Note that $\mathbf{s}(t)$ is typically unknown to us at the beginning of slot t and is only disclosed to us at the end of slot t.

7.1.2 Basic Assumption

Assumption 7.1.

- *There exist a constant* $e^{\max} > 0$ *such that* $0 \leq e(t) \leq e^{\max}, \forall t \in \{1, 2, \ldots\}$.
- *Let* $\nabla_{\mathbf{p}} U(\mathbf{p}; \omega)$ *denote a subgradient (or gradient if* $U(\mathbf{p}; \omega)$ *is differentiable) vector of* $U(\mathbf{p}; \omega)$ *with respect to* $\mathbf{p}$ *and let* $\frac{\partial}{\partial p_i} U(\mathbf{p}; \omega), \forall i \in \{1, 2, \ldots, n\}$ *denote each component of vector* $\nabla_{\mathbf{p}} U(\mathbf{p}; \omega)$. *There exist positive constants* $D_1, \ldots, D_n$ *such that* $|\frac{\partial}{\partial p_i} U(\mathbf{p}; \omega)| \leq D_i, \forall i \in \{1, 2, \ldots, n\}$ *for all* $\omega \in \Omega$ *and all* $\mathbf{p} \in \mathcal{P}$. *This further implies there exists* $D > 0$, *e.g.,* $D = \sqrt{\sum_{i=1}^{n} D_i^2}$, *such that* $\|\nabla_{\mathbf{p}} U(\mathbf{p}; \omega)\| \leq D$ *for all* $\omega \in \Omega$ *and all* $\mathbf{p} \in \mathcal{P}$, *where* $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{n} x_i^2}$ *is the standard* l_2 *norm.*

Such constants $D_1, \ldots, D_n$ exist in most cases of interest, such as for utility functions (7.1) with bounded $s_i(t)$ values. [1] The following fact follows directly from Assumption 7.1.

Fact 7.1. *For each $\omega \in \Omega$, $U(\mathbf{p}, \omega)$ is D-Lipschitz over $p \in \mathcal{P}$, i.e.,*

$$|U(\mathbf{p}_1, \omega) - U(\mathbf{p}_2, \omega)| \leq D\|\mathbf{p}_1 - \mathbf{p}_2\|, \forall \mathbf{p}_1, \mathbf{p}_2 \in \mathcal{P}$$

Proof. By the basic subgradient inequality for concave functions:

$$U(\mathbf{p}_2, \omega) + [\nabla_{\mathbf{p}} U(\mathbf{p}_2; \omega)]^{\mathsf{T}}(\mathbf{p}_1 - \mathbf{p}_2) \geq U(\mathbf{p}_1, \omega)$$
$$U(\mathbf{p}_1, \omega) + [\nabla_{\mathbf{p}} U(\mathbf{p}_1; \omega)]^{\mathsf{T}}(\mathbf{p}_2 - \mathbf{p}_1) \geq U(\mathbf{p}_2, \omega)$$

Rearranging terms and applying the Cauchy-Schwarz inequality yields

$$U(\mathbf{p}_1, \omega) - U(\mathbf{p}_2, \omega) \leq \|\nabla_{\mathbf{p}} U(\mathbf{p}_2; \omega)\| \|\mathbf{p}_1 - \mathbf{p}_2\|$$
$$U(\mathbf{p}_2, \omega) - U(\mathbf{p}_1, \omega) \leq \|\nabla_{\mathbf{p}} U(\mathbf{p}_1; \omega)\| \|\mathbf{p}_1 - \mathbf{p}_2\|$$

Combining the above inequalities and recalling that all subgradients are bounded by D gives $|U(\mathbf{p}_1, \omega) - U(\mathbf{p}_2, \omega)| \leq D\|\mathbf{p}_1 - \mathbf{p}_2\|$. □

7.1.3 Power Control and Energy Queue Model

The finite size battery can be considered as backlog in an *energy queue*. Let $E(0)$ be the initial energy backlog in the battery and $E(t)$ be the energy stored in the battery at the **end** of slot t. The power vector $\mathbf{p}(t)$ must satisfy the following *energy availability constraint*:

$$\sum_{i=1}^{n} p_i(t) \leq E(t-1), \forall t \in \{1, 2, \ldots\}. \tag{7.2}$$

which requires the consumed power to be no more than what is available in the battery.

Let $E^{\max}$ be the maximum capacity of the battery. If the energy availability constraint (7.2)

[1] This is always true since $s_i(t)$ are wireless signal strength attenuations.

is satisfied on each slot, the energy queue backlog $E(t)$ evolves as follows:

$$E(t) = \min\{E(t-1) - \sum_{i=1}^{n} p_i(t) + e(t), E^{\max}\}, \forall t. \tag{7.3}$$

7.1.4 An Upper Bound Problem

Let $\omega(t) = [e(t), s(t)]$ be the random state vector on slot t. Let $\mathbb{E}[e] = \mathbb{E}[e(t)]$ denote the expected amount of new energy that arrives in one slot. Define a function $h : \mathcal{P} \to \mathbb{R}$ by

$$h(\mathbf{p}) = \mathbb{E}[U(\mathbf{p}; \omega(t))].$$

Since $U(\mathbf{p}; \omega)$ is concave in $\mathbf{p}$ for all ω by Assumption 7.1 and is D-Lipschitz over $p \in \mathcal{P}$ for all ω by Fact 7.1, we know $h(\mathbf{p})$ is concave and continuous.

The function h is typically unknown because the distribution of ω is unknown. However, to establish a fundamental bound, suppose both h and $\mathbb{E}[e]$ are known and consider choosing a fixed vector $\mathbf{p}$ to solve the following deterministic problem:

$$\max_{\mathbf{p}} \ h(\mathbf{p}) \tag{7.4}$$

$$\text{s.t.} \ \sum_{i=1}^{n} p_i - \mathbb{E}[e] \leq 0 \tag{7.5}$$

$$\mathbf{p} \in \mathcal{P} \tag{7.6}$$

where constraint (7.5) requires that the consumed energy is no more than $\mathbb{E}[e]$.

Let $\mathbf{p}^*$ be an optimal solution of problem (7.4)-(7.6) and U^* be its corresponding utility value of (7.4). Define a *causal policy* as one that, on each slot t, selects $\mathbf{p}(t) \in \mathcal{P}$ based only on information up to the start of slot t (in particular, without knowledge of $\omega(t)$). Since $\omega(t)$ is i.i.d. over slots, any causal policy must have $\mathbf{p}(t)$ and $\omega(t)$ independent for all t. The next lemma shows that no causal policy $\mathbf{p}(t), t \in \{1, 2, \ldots\}$ satisfying (7.2)-(7.3) can attain a better utility than U^*.

Lemma 7.1. *Let $\mathbf{p}(t) \in \mathcal{P}, t \in \{1, 2, \ldots\}$ be yielded by any causal policy that consumes less*

energy than it harvests in the long term, so $\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\sum_{i=1}^{n} p_i(t)] \leq \mathbb{E}[e]$. *Then,*

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[U(\mathbf{p}(t);\omega(t))] \leq U^*.$$

Proof. Fix a slot $t \in \{1, 2, \ldots\}$. Then

$$\mathbb{E}[U(\mathbf{p}(t);\omega(t))] \overset{(a)}{=} \mathbb{E}[\mathbb{E}[U(\mathbf{p}(t);\omega(t))|\mathbf{p}(t)]] \overset{(b)}{=} \mathbb{E}[h(\mathbf{p}(t))] \tag{7.7}$$

where (a) holds by iterated expectations; (b) holds because $\mathbf{p}(t)$ and $\omega(t)$ are independent (by causality).

For each $T > 0$ define $\overline{\mathbf{p}}(T) = [\overline{p}_1(T), \ldots, \overline{p}_n(T)]^{\mathsf{T}}$ with

$$\overline{p}_i(T) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[p_i(t)], \forall i \in \{1, 2, \ldots, n\}.$$

We know by assumption that:

$$\limsup_{T\to\infty} \sum_{i=1}^{n} \overline{p}_i(T) \leq \mathbb{E}[e] \tag{7.8}$$

Further, since $\mathbf{p}(t) \in \mathcal{P}$ for all slots t, it holds that $\overline{\mathbf{p}}(T) \in \mathcal{P}$ for all $T > 0$. Also,

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[U(\mathbf{p}(t);\omega(t))] &\overset{(a)}{=} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[h(\mathbf{p}(t))] \\ &\overset{(b)}{\leq} h\left(\mathbb{E}[\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\mathbf{p}(t)]\right) \\ &= h(\bar{\mathbf{p}}(T))\end{aligned}$$

where (a) holds by (7.7); (b) holds by Jensen's inequality for the concave function h. It follows that:

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[U(\mathbf{p}(t);\omega(t))] \leq \limsup_{T\to\infty} h(\bar{\mathbf{p}}(T)).$$

Define $\theta = \limsup_{T\to\infty} h(\bar{\mathbf{p}}(T))$. It suffices to show that $\theta \leq U^*$. Since $\bar{\mathbf{p}}(T)$ is in the compact set $\mathcal{P}$ for all $T > 0$ and h is continuous, the Bolzano-Weierstrass theorem ensures there is a subsequence of times T_k such that $\overline{\mathbf{p}}(T_k)$ converges to a fixed vector $\mathbf{p}_0 \in \mathcal{P}$ and $h(\overline{\mathbf{p}}(T_k))$

converges to θ as $k \to \infty$:

$$\lim_{k\to\infty} \overline{\mathbf{p}}(T_k) = \mathbf{p}_0 \in \mathcal{P}$$

$$\lim_{k\to\infty} h(\overline{\mathbf{p}}(T_k)) = \theta$$

Continuity of h implies that $h(\mathbf{p}_0) = \theta$. By (7.8) the vector $\mathbf{p}_0 = [p_{0,1}, \ldots, p_{0,n}]^{\mathsf{T}}$ must satisfy $\sum_{i=1}^{n} p_{0,i} \leq \mathbb{E}[e]$. Hence, $\mathbf{p}_0$ is a vector that satisfies constraints (7.5)-(7.6) and achieves utility $h(\mathbf{p}_0) = \theta$. Since U^* is defined as the optimal utility value to problem (7.4)-(7.6), it holds that $\theta \leq U^*$. □

Note that the U^* utility upper bound of Lemma 7.1 holds for any policy that consumes no more energy than it harvests in the long term. Policies that satisfy the physical battery constraints (7.2)-(7.3) certainly consume no more energy than harvested in the long term. However, Lemma 7.1 even holds for policies that violate these physical battery constraints. For example, U^* is still a valid bound for a policy that is allowed to "borrow" energy from an external power source when its battery is empty and "return" energy when its battery is full.

7.2 New Algorithm

This subsection proposes a new learning aided dynamic power control algorithm that chooses power control actions based on system history, without requiring the current system state or its probability distribution.

7.2.1 New Algorithm

The new dynamic power control algorithm is described in Algorithm 7.1. At the end of slot t, Algorithm 7.1 chooses $\mathbf{p}(t+1)$ based on $\omega(t)$ without requiring $\omega(t+1)$. To enable these decisions, the algorithm introduces a (non-positive) *virtual* battery queue process $Q(t) \leq 0$, which shall later be shown to be related to a shifted version of the physical battery queue $E(t)$.

Note that Algorithm 7.1 does not explicitly enforce the energy availability constraint (7.2). Let $\mathbf{p}(t+1)$ be given by (7.10), one may expect to use

$$\hat{\mathbf{p}}(t+1) = \frac{\min\{\sum_{i=1}^{n} p_i(t+1), E(t)\}}{\sum_{i=1}^{n} p_i(t+1)} \mathbf{p}(t+1) \tag{7.11}$$

Algorithm 7.1 New Power Control Algorithm for Energy Harvesting Devices with Outdated State Information

Let $V > 0$ be a constant algorithm parameter. Initialize virtual battery queue variable $Q(0) = 0$. Choose $\mathbf{p}(1) = [0, 0, \ldots, 0]^{\mathsf{T}}$ as the power action at slot 1. At the *end* of each slot $t \in \{1, 2, \ldots\}$, observe $\omega(t) = [e(t), \mathbf{s}(t)]$ and do the following:

- **Update virtual battery queue** $Q(t)$: Update $Q(t)$ via:

$$Q(t) = \min\{Q(t-1) + e(t) - \sum_{i=1}^{n} p_i(t), 0\}. \tag{7.9}$$

- **Power control**: Choose

$$\mathbf{p}(t+1) = \text{Proj}_{\mathcal{P}}\Big\{\mathbf{p}(t) + \frac{1}{V}\nabla_{\mathbf{p}} U(\mathbf{p}(t); \omega(t)) + \frac{1}{V^2}Q(t)\mathbf{1}\Big\} \tag{7.10}$$

  as the power action for the next slot $t+1$ where $\text{Proj}_{\mathcal{P}}\{\cdot\}$ represents the projection onto set $\mathcal{P}$, $\mathbf{1}$ denotes a column vector of all ones and $\nabla_{\mathbf{p}} U(\mathbf{p}(t); \omega(t))$ represents a subgradient (or gradient if $U(\mathbf{p}; \omega(t))$ is differentiable) vector of function $U(\mathbf{p}; \omega(t))$ at point $\mathbf{p} = \mathbf{p}(t)$. Note that $\mathbf{p}(t)$, $Q(t)$ and $\nabla_{\mathbf{p}} U(\mathbf{p}(t); \omega(t))$ are given constants in (7.10).

that scales down $\mathbf{p}(t+1)$ to enforce the energy availability constraint (7.2). However, our analysis in Section 7.3 shows that if the battery capacity is at least as large as an $O(V)$ constant, then directly using $\mathbf{p}(t+1)$ from (7.10) is ensured to always satisfy the energy availability constraint (7.2). Thus, there is no need to take the additional step (7.11).

7.2.2 Algorithm Inuitions

Lemma 7.2. *The power control action* $\mathbf{p}(t+1)$ *chosen in* (7.10) *is to solve the following quadratic convex program*

$$\max_{\mathbf{p}} \; V[\nabla_{\mathbf{p}} U(\mathbf{p}(t); \omega(t))]^{\mathsf{T}}[\mathbf{p} - \mathbf{p}(t)] + Q(t)\mathbf{1}^{\mathsf{T}}\mathbf{p} - \frac{V^2}{2}\|\mathbf{p} - \mathbf{p}(t)\|^2 \tag{7.12}$$

$$s.t. \;\; \mathbf{p} \in \mathcal{P} \tag{7.13}$$

Proof. By the definition of projection, equation (7.10) is to solve

$$\min \; \|\mathbf{p} - \big[\mathbf{p}(t) + \frac{1}{V}\nabla_{\mathbf{p}} U(\mathbf{p}(t); \omega(t)) + \frac{1}{V^2}Q(t)\mathbf{1}\big]\|^2$$

$$\text{s.t.} \;\; \mathbf{p} \in \mathcal{P}$$

By expanding the square, eliminating constant terms and converting the minimization to the

maximization of its negative object, it is easy to show this problem is equivalent to problem (7.12)-(7.13). □

The convex projection (7.10), or equivalently, the quadratic convex program (7.12)-(7.13) can be easily solved. See e.g., Lemma 4.4 in Chapter 4 for an algorithm that solves an n-dimensional quadratic program over set $\mathcal{P}$ with complexity $O(n \log n)$. Thus, the overall complexity of Algorithm 7.1 is low.

7.3 Performance Analysis of Algorithm 7.1

This section shows Algorithm 7.1 can attain an $O(\epsilon)$ close-to-optimal utility by using a battery with capacity $O(1/\epsilon)$.

7.3.1 Drift Analysis

Define $L(t) = \frac{1}{2}[Q(t)]^2$ and call it a *Lyapunov function*. Define the *Lyapunov drift* as $\Delta(t) = L(t+1) - L(t)$.

Lemma 7.3. *Under Algorithm 7.1, for all $t \geq 0$, the Lyapunov drift satisfies*

$$\Delta(t) \leq Q(t)[e(t+1) - \sum_{i=1}^{n} p_i(t+1)] + \frac{1}{2}B \tag{7.14}$$

with constant $B = (\max\{e^{\max}, p^{\max}\})^2$, where $e^{\max}$ is the constant defined in Assumption 7.1.

Proof. Fix $t \geq 0$. Recall that for any $x \in \mathbb{R}$ if $y = \min\{x, 0\}$ then $y^2 \leq x^2$. It follows from (7.9) that

$$[Q(t+1)]^2 \leq \big[Q(t) + e(t+1) - \sum_{i=1}^{n} p_i(t+1)\big]^2.$$

Expanding the square on the right side, dividing both sides by 2 and rearranging terms yields $\Delta(t) \leq Q(t)[e(t+1) - \sum_{i=1}^{n} p_i(t+1)] + \frac{1}{2}[e(t+1) - \sum_{i=1}^{n} p_i(t+1)]^2$.

This lemma follows by noting that $|e(t+1) - \sum_{i=1}^{n} p_i(t+1)| \leq \max\{e^{\max}, p^{\max}\}$ since $0 \leq \sum_{i=1}^{n} p_i(t+1) \leq p^{\max}$ and $0 \leq e(t+1) \leq e^{\max}$. □

Lemma 7.4. *Let U^* be the utility upper bound defined in Lemma 7.1 and $\mathbf{p}^*$ be an optimal solution to problem* (7.4)-(7.6) *that attains U^*. At each iteration $t \in \{1, 2, \ldots\}$, Algorithm 7.1 guarantees*

$$V\mathbb{E}[U(\mathbf{p}(t);\omega(t))] - \Delta(t) \geq VU^* + \frac{V^2}{2}\mathbb{E}[\Phi(t)] - \frac{D^2+B}{2}$$

where $\Phi(t) = \|\mathbf{p}^* - \mathbf{p}(t+1)\|^2 - \|\mathbf{p}^* - \mathbf{p}(t)\|^2$, D is the constant defined in Assumption 7.1 and B is the constant defined in Lemma 7.3.

Proof. Note that $\sum_{i=1}^n p_i^* \leq \mathbb{E}[e]$. Fix $t \in \{1, 2, \ldots\}$. Note that $V[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))]^{\mathsf{T}}[\mathbf{p} - \mathbf{p}(t)] + Q(t)\sum_{i=1}^n p_i$ is a linear function with respect to $\mathbf{p}$. It follows that

$$V\big[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p} - \mathbf{p}(t)] + Q(t)\sum_{i=1}^n p_i - \frac{V^2}{2}\|\mathbf{p} - \mathbf{p}(t)\|^2 \tag{7.15}$$

is strongly concave with respect to $\mathbf{p} \in \mathcal{P}$ with modulus V^2. Since $\mathbf{p}(t+1)$ is chosen to maximize (7.15) over all $\mathbf{p} \in \mathcal{P}$, and since $\mathbf{p}^* \in \mathcal{P}$, by Corollary 1.3, we have

$$\begin{aligned}
&V\big[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}(t+1) - \mathbf{p}(t)] + Q(t)\sum_{i=1}^n p_i(t+1) - \frac{V^2}{2}\|\mathbf{p}(t+1) - \mathbf{p}(t)\|^2 \\
\geq &V\big[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}^* - \mathbf{p}(t)] + Q(t)\sum_{i=1}^n p_i^* - \frac{V^2}{2}\|\mathbf{p}^* - \mathbf{p}(t)\|^2 + \frac{V^2}{2}\|\mathbf{p}^* - \mathbf{p}(t+1)\|^2 \\
= &V\big[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}^* - \mathbf{p}(t)] + Q(t)\sum_{i=1}^n p_i^* + \frac{V^2}{2}\Phi(t).
\end{aligned}$$

Subtracting $Q(t)e(t+1)$ from both sides and rearranging terms yields

$$\begin{aligned}
&V\big[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}(t+1) - \mathbf{p}(t)] + Q(t)\big[\sum_{i=1}^n p_i(t+1) - e(t+1)\big] \\
\geq &V\big[\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}^* - \mathbf{p}(t)] + Q(t)\big[\sum_{i=1}^n p_i^* - e(t+1)\big] + \frac{V^2}{2}\Phi(t) + \frac{V^2}{2}\|\mathbf{p}(t+1) - \mathbf{p}(t)\|^2.
\end{aligned}$$

Adding $VU(\mathbf{p}(t);\omega(t))$ to both sides and noting that $U(\mathbf{p}(t);\omega(t)) + [\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t))]^{\mathsf{T}}[\mathbf{p}^* -$

$\mathbf{p}(t)] \geq U(\mathbf{p}^*;\omega(t))$ by the concavity of $U(\mathbf{p};\omega(t))$ yields

$$\begin{aligned}
&VU(\mathbf{p}(t);\omega(t)) + V\big[\nabla_{\mathbf{p}}U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}(t+1)-\mathbf{p}(t)] + Q(t)\big[\sum_{i=1}^{n} p_i(t+1) - e(t+1)\big] \\
\geq & VU(\mathbf{p}^*;\omega(t)) + Q(t)\big[\sum_{i=1}^{n} p_i^* - e(t+1)\big] + \frac{V^2}{2}\Phi(t) + \frac{V^2}{2}\|\mathbf{p}(t+1)-\mathbf{p}(t)\|^2.
\end{aligned}$$

Rearranging terms yields

$$\begin{aligned}
&VU(\mathbf{p}(t);\omega(t)) + Q(t)\big[\sum_{i=1}^{n} p_i(t+1) - e(t+1)\big] \\
\geq & VU(\mathbf{p}^*;\omega(t)) + Q(t)\big[\sum_{i=1}^{n} p_i^* - e(t+1)\big] + \frac{V^2}{2}\Phi(t) + \frac{V^2}{2}\|\mathbf{p}(t+1)-\mathbf{p}(t)\|^2 \\
&- V\big[\nabla_{\mathbf{p}}U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}(t+1)-\mathbf{p}(t)] \qquad (7.16)
\end{aligned}$$

Note that

$$\begin{aligned}
V\big[\nabla_{\mathbf{p}}U(\mathbf{p}(t);\omega(t))\big]^{\mathsf{T}}[\mathbf{p}(t+1)-\mathbf{p}(t)] &\overset{(a)}{\leq} \frac{1}{2}\|\nabla_{\mathbf{p}}U(\mathbf{p}(t);\omega(t))\|^2 + \frac{V^2}{2}\|\mathbf{p}(t+1)-\mathbf{p}(t)\|^2 \\
&\overset{(b)}{\leq} \frac{1}{2}D^2 + \frac{V^2}{2}\|\mathbf{p}(t+1)-\mathbf{p}(t)\|^2 \qquad (7.17)
\end{aligned}$$

where (a) follows by using basic inequality $\mathbf{x}^{\mathsf{T}}\mathbf{y} \leq \frac{1}{2}\|\mathbf{x}\|^2 + \frac{1}{2}\|\mathbf{y}\|^2$ for all $\mathbf{x},\mathbf{y} \in \mathbb{R}^n$ with $\mathbf{x} = \nabla_{\mathbf{p}}U(\mathbf{p}(t);\omega(t))$ and $\mathbf{y} = V[\mathbf{p}(t+1)-\mathbf{p}(t)]$; and (b) follows from Assumption 7.1. Substituting (7.17) into (7.16) yields

$$\begin{aligned}
&VU(\mathbf{p}(t);\omega(t)) + Q(t)\big[\sum_{i=1}^{n} p_i(t+1) - e(t+1)\big] \\
\geq & VU(\mathbf{p}^*;\omega(t)) + Q(t)\big[\sum_{i=1}^{n} p_i^* - e(t+1)\big] + \frac{V^2}{2}\Phi(t) - \frac{1}{2}D^2 \qquad (7.18)
\end{aligned}$$

By Lemma 7.3, we have

$$-\Delta(t) \geq Q(t)\big[\sum_{i=1}^{n} p_i(t+1) - e(t+1)\big] - \frac{B}{2} \qquad (7.19)$$

Summing (7.18) and (7.19); and cancelling common terms on both sides yields

$$VU(\mathbf{p}(t);\omega(t)) - \Delta(t) \geq VU(\mathbf{p}^*;\omega(t)) + Q(t)\big[\sum_{i=1}^{n} p_i^* - e(t+1)\big] + \frac{V^2}{2}\Phi(t) - \frac{D^2+B}{2} \quad (7.20)$$

Note that each $Q(t)$ (depending only on $e(\tau), p(\tau)$ with $\tau \in \{1,2,\ldots,t\}$) is independent of $e(t+1)$. Thus,

$$\begin{aligned}\mathbb{E}\Big[Q(t)[\sum_{i=1}^{n} p_i^* - e(t+1)]\Big] =& \mathbb{E}[Q(t)]\mathbb{E}[\sum_{i=1}^{n} p_i^* - e(t+1)] \\ \overset{(a)}{\geq}& 0 \end{aligned} \quad (7.21)$$

where (a) follows because $Q(t) \leq 0$ and $\sum_{i=1}^{n} p_i^* \leq \mathbb{E}[e]$ (recall that $e(t+1)$ is an i.i.d. sample of e).

Taking expectations on both sides of (7.20) and using (7.21) and $\mathbb{E}[U(\mathbf{p}^*;\omega(t))] = U^*$ yields the desired result. □

7.3.2 Utility Optimality Analysis

The next theorem summarizes that the average expected utility attained by Algorithm 7.1 is within an $O(1/V)$ distance to U^* defined in Lemma 7.1.

Theorem 7.1. *Let U^* be the utility bound defined in Lemma 7.1. For all $t \in \{1,2,\ldots\}$, Algorithm 7.1 guarantees*

$$\frac{1}{t}\sum_{\tau=1}^{t} \mathbb{E}[U(\mathbf{p}(\tau);\omega(\tau))] \geq U^* - \frac{V(p^{\max})^2}{2t} - \frac{B}{2Vt} - \frac{D^2+B}{2V} \quad (7.22)$$

where D is the constant defined in Assumption 7.1 and B is the constant defined in Lemma 7.3. This implies,

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=1}^{t} \mathbb{E}[U(\mathbf{p}(\tau);\omega(\tau))] \geq U^* - \frac{D^2+B}{2V}. \quad (7.23)$$

In particular, if we take $V = 1/\epsilon$ in Algorithm 7.1, then

$$\frac{1}{t}\sum_{\tau=1}^{t} \mathbb{E}[U(\mathbf{p}(\tau);\omega(\tau))] \geq U^* - O(\epsilon), \forall t \geq \Omega(\frac{1}{\epsilon^2}). \quad (7.24)$$

Proof. Fix $t \in \{1, 2, \ldots\}$. For each $\tau \in \{1, 2, \ldots, t\}$, by Lemma 7.4, we have

$$\mathbb{E}[VU(\mathbf{p}(\tau); \omega(\tau))] - \mathbb{E}[\Delta(\tau)] \geq VU^* + \frac{V^2}{2}\mathbb{E}[\Phi(\tau)] - \frac{D^2 + B}{2}.$$

Summing over $\tau \in \{1, 2, \ldots, t\}$, dividing both sides by Vt and rearranging terms yields

$$\begin{aligned}
&\frac{1}{t}\sum_{\tau=1}^{t}\mathbb{E}[U(\mathbf{p}(\tau); \omega(\tau))] \\
\geq& U^* + \frac{V}{2t}\sum_{\tau=1}^{t}\mathbb{E}[\Phi(\tau)] + \frac{1}{Vt}\sum_{\tau=1}^{t}\mathbb{E}[\Delta(\tau)] - \frac{D^2 + B}{2V} \\
\overset{(a)}{=}& U^* + \frac{V}{2t}\mathbb{E}[\|\mathbf{p}^* - \mathbf{p}(t+1)\|^2 - \|\mathbf{p}^* - \mathbf{p}(1)\|^2] + \frac{1}{2Vt}\mathbb{E}[[Q(t+1)]^2 - [Q(1)]^2] - \frac{D^2 + B}{2V} \\
\geq& U^* - \frac{V}{2t}\mathbb{E}[\|\mathbf{p}^* - \mathbf{p}(1)\|^2] - \frac{1}{2Vt}\mathbb{E}[[Q(1)]^2] - \frac{D^2 + B}{2V} \\
\overset{(b)}{\geq}& U^* - \frac{V(p^{\max})^2}{2t} - \frac{B}{2Vt} - \frac{D^2 + B}{2V}
\end{aligned}$$

where (a) follows by recalling that $\Phi(\tau) = \|\mathbf{p}^* - \mathbf{p}(\tau+1)\|^2 - \|\mathbf{p}^* - \mathbf{p}(\tau)\|^2$ and $\Delta(\tau) = \frac{1}{2}[Q(\tau+1)]^2 - \frac{1}{2}[Q(\tau)]^2$; and (b) follows because $\|\mathbf{p}^* - \mathbf{p}(1)\| = \|\mathbf{p}^*\| = \sqrt{\sum_{i=1}^{n}(p_i^*)^2} \leq \sum_{i=1}^{n} p_i^* \leq p^{\max}$ and $|Q(1)| = |Q(0) + e(1) - \sum_{i=1}^{n} p_i(1)| = |e(1) - \sum_{i=1}^{n} p_i(1)| \leq \max\{e^{\max}, p^{\max}\} = \sqrt{B}$ where B is defined in Lemma 7.3. So far we have proven (7.22).

Equation (7.23) follows directly by taking lim sup on both sides of (7.22). Equation (7.24) follows by substituting $V = \frac{1}{\epsilon}$ and $t = \frac{1}{\epsilon^2}$ into (7.22). □

7.3.3 Lower Bound for Virtual Battery Queue $Q(t)$

Note that $Q(t) \leq 0$ by (7.9). This subsection further shows that $Q(t)$ is bounded from below. The projection $\text{Proj}_{\mathcal{P}}\{\cdot\}$ satisfies the following lemma:

Lemma 7.5. *For any $\mathbf{p}(t) \in \mathcal{P}$ and vector $\mathbf{b} \leq \mathbf{0}$, where $\leq$ between two vectors means component-wisely less than or equal to, $\tilde{\mathbf{p}} = Proj_{\mathcal{P}}\{\mathbf{p}(t) + \mathbf{b}\}$ is given by*

$$\tilde{p}_i = \max\{p_i(t) + b_i, 0\}, \forall i \in \{1, 2, \ldots, n\}. \tag{7.25}$$

Proof. Recall that projection $\text{Proj}_{\mathcal{P}}\{\mathbf{p}(t)+\mathbf{b}\}$ by definition is to solve

$$\min_{\mathbf{p}} \quad \sum_{i=1}^{n}[p_i - [p_i(t)+b_i]]^2 \tag{7.26}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} p_i \le p^{\max} \tag{7.27}$$

$$p_i \ge 0, \forall i \in \{1,2,\ldots,n\} \tag{7.28}$$

Let $\mathcal{I} \subseteq \{1,2,\ldots,n\}$ be the coordinate index set given by $\mathcal{I} = \{i \in \{1,2,\ldots,n\} : p_i(t)+b_j < 0\}$. For any $\mathbf{p}$ such that $\sum_{i=1}^{n} p_i \le p^{\max}$ and $p_i \ge 0, \forall i \in \{1,2,\ldots,n\}$, we have

$$\begin{aligned}
&\sum_{i=1}^{n}[p_i - [p_i(t)+b_i]]^2 \\
=&\sum_{i\in\mathcal{I}}[p_i - [p_i(t)+b_i]]^2 + \sum_{i\in\{1,2,\ldots,n\}\setminus\mathcal{I}}[p_i - [p_i(t)+b_i]]^2 \\
\ge&\sum_{i\in\mathcal{I}}[p_i - [p_i(t)+b_i]]^2 \\
\overset{(a)}{\ge}&\sum_{i\in\mathcal{I}}[p_i(t)+b_i]^2
\end{aligned}$$

where (a) follows because $p_i(t)+b_i < 0$ for $i \in \mathcal{I}$ and $p_i \ge 0, \forall i \in \{1,2,\ldots,n\}$. Thus, $\sum_{i\in\mathcal{I}}[p_i(t)+b_i]^2$ is an object value lower bound of problem (7.26)-(7.28).

Note that $\tilde{\mathbf{p}}$ given by (7.25) is feasible to problem (7.26)-(7.28) since $\tilde{p}_i \ge 0, \forall i \in \{1,2,\ldots,n\}$ and $\sum_{i=1}^{n}\tilde{p}_i \le \sum_{i=1}^{n} p_i(t) \le p^{\max}$ because $\tilde{p}_i \le p_i(t)$ for all i and $\mathbf{p}(t) \in \mathcal{P}$. We further note that

$$\sum_{i=1}^{n}[\tilde{p}_i - [p_i(t)+b_i]]^2 = \sum_{i\in\mathcal{I}}[p_i(t)+b_i]^2.$$

That is, $\tilde{\mathbf{p}}$ given by (7.25) attains the object value lower bound of problem (7.26)-(7.28) and hence is the optimal solution to problem (7.26)-(7.28). Thus, $\tilde{\mathbf{p}} = \text{Proj}_{\mathcal{P}}\{\mathbf{p}(t)+\mathbf{b}\}$. □

Corollary 7.1. *If $Q(t) \le -V(D^{\max}+p^{\max})$ with $D^{\max} = \max\{D_1,\ldots,D_n\}$, then Algorithm 7.1 guarantees*

$$p_i(t+1) \le \max\{p_i(t) - \frac{1}{V}p^{\max}, 0\}, \forall i \in \{1,2,\ldots,n\}.$$

where $D_1,\ldots,D_n$ are constants defined in Assumption 7.1.

Proof. Let $\mathbf{b} = \frac{1}{V}\nabla_{\mathbf{p}} U(\mathbf{p}(t);\omega(t)) + \frac{1}{V^2}Q(t)\mathbf{1}$. Since $\frac{\partial}{\partial p_i}U(\mathbf{p}(t);\omega(t)) \leq D_i, \forall i \in \{1,2,\ldots,n\}$ by Assumption 7.1 and $Q(t) \leq -V(D^{\max} + p^{\max})$, we know $b_i \leq -\frac{1}{V}p^{\max}, \forall i \in \{1,2,\ldots,n\}$. By Lemma 7.5, we have

$$\begin{aligned} p_i(t+1) =& \max\{p_i(t) + b_i, 0\} \\ \leq& \max\{p_i(t) - \frac{1}{V}p^{\max}, 0\}, \forall i \in \{1,2,\ldots,n\}. \end{aligned}$$

□

By Corollary 7.1, if $Q(t) \leq -V(D^{\max} + p^{\max})$, then each component of $\mathbf{p}(t+1)$ decreases by $\frac{1}{V}p^{\max}$ until it hits 0. That is, if $Q(t) \leq -V(D^{\max} + p^{\max})$ for sufficiently many slots, Algorithm 7.1 eventually chooses $\mathbf{0}$ as the power decision. By virtual queue update equation (7.9), $Q(t)$ decreases only when $\sum_{i=1}^{n} p_i(t) > 0$. These two observations suggest that $Q(t)$ yielded by Algorithm 7.1 should be eventually bounded from below. This is formally summarized in the next theorem.

Theorem 7.2. *Let V in Algorithm 7.1 be a positive integer. Define positive constant Q^l, where superscript l denotes "lower" bound, as*

$$Q^l = V(D^{\max} + 2p^{\max} + e^{\max}) \tag{7.29}$$

where $e^{\max}$ is the constant defined in Assumption 7.1 and $D^{\max}$ is the constant defined in Corollary 7.1. Algorithm 7.1 guarantees

$$Q(t) \geq -Q^l, \forall t \in \{0,1,2,\ldots\}.$$

Proof. By virtual queue update equation (7.9), we know $Q(t)$ can increase by at most $e^{\max}$ and can decrease by at most $p^{\max}$ on each slot. Since $Q(0) = 0$, we know $Q(t) \geq -Q^l$ for all $t \leq V$. We need to show $Q(t) \geq -Q^l$ for all $t > V$. This can be proven by contradiction as follows:

Assume $Q(t) < -Q^l$ for some $t > V$. Let $\tau > V$ be the *first* (smallest) slot index when this happens. By the definition of τ, we have $Q(\tau) < -Q^l$ and

$$Q(\tau) < Q(\tau - 1). \tag{7.30}$$

Now consider the value of $Q(\tau - V)$ in two cases (note that $\tau - V > 0$).

- Case $Q(\tau - V) \geq -V(D^{\max} + p^{\max} + e^{\max})$: Since $Q(t)$ can decrease by at most $p^{\max}$ on each slot, we know $Q(\tau) \geq -V(D^{\max} + 2p^{\max} + e^{\max}) = -Q^l$. This contradicts the definition of τ.

- Case $Q(\tau - V) < -V(D^{\max} + p^{\max} + e^{\max})$: Since $Q(t)$ can increase by at most $e^{\max}$ on each slot, we know $Q(t) < -V(D^{\max} + p^{\max})$ for all $\tau - V \leq t \leq \tau - 1$. By Corollary 7.1, for all $\tau - V \leq t \leq \tau - 1$, we have

$$p_i(t+1) \leq \max\{p_i(t) - \frac{1}{V}p^{\max}, 0\}, \forall i \in \{1, 2, \ldots, n\}.$$

  Since the above inequality holds for all $t \in \{\tau - V, \tau - V + 1, \ldots, \tau - 1\}$, and since at the start of this interval we trivially have $p_i(\tau - V) \leq p^{\max}, \forall i \in \{1, 2, \ldots, n\}$, at each step of this interval each component of the power vector either hits zero or decreases by $\frac{1}{V}p^{\max}$, and so after the V steps of this interval we have $p_i(\tau) = 0, \forall i \in \{1, 2, \ldots, n\}$. By (7.9), we have

$$\begin{aligned} Q(\tau) =& \min\{Q(\tau - 1) + e(\tau) - \sum_{i=1}^{n} p_i(\tau), 0\} \\ =& \min\{Q(\tau - 1) + e(\tau), 0\} \\ \geq& \min\{Q(\tau - 1), 0\} \\ =& Q(\tau - 1) \end{aligned}$$

  where the final equality holds because the queue is never positive (see (7.9)). This contradicts (7.30).

Both cases lead to contradictions. Thus, $Q(t) \geq -Q^l$ for all $t > V$. □

7.3.4 Energy Availability Guarantee

To implement the power decisions of Algorithm 7.1 for the physical battery system $E(t)$ from equations (7.2)-(7.3), we must ensure the energy availability constraint (7.2) holds on each slot. The next theorem shows that Algorithm 7.1 ensures the constraint (7.2) always holds as long as the battery capacity satisfies $E^{\max} \geq Q^l + p^{\max}$ and the initial energy satisfies $E(0) = E^{\max}$. It

also explains that $Q(t)$ used in Algorithm 7.1 is a shifted version of the physical battery backlog $E(t)$.

Theorem 7.3. *If $E(0) = E^{\max} \geq Q^l + p^{\max}$, where Q^l is the constant defined in Theorem 7.2, then Algorithm 7.1 ensures the energy availability constraint* (7.2) *on each slot $t \in \{1, 2, \ldots\}$. Moreover*

$$E(t) = Q(t) + E^{\max}, \forall t \in \{0, 1, 2, \ldots\}. \tag{7.31}$$

Proof. Note that to show the energy availability constraint $\sum_{i=1}^{n} p_i(t) \leq E(t-1), \forall t \in \{1, 2, \ldots\}$ is equivalent to show

$$\sum_{i=1}^{n} p_i(t+1) \leq E(t), \forall t \in \{0, 1, 2, \ldots\}. \tag{7.32}$$

This lemma can be proven by inductions.

Note that $E(0) = E^{\max}$ and $Q(0) = 0$. It is immediate that (7.31) holds for $t = 0$. Since $E(0) = E^{\max} \geq p^{\max}$ and $\sum_{i=1}^{n} p_i(1) \leq p^{\max}$, equation (7.32) also holds for $t = 0$. Assume (7.32) and (7.31) hold for $t = t_0$ and consider $t = t_0 + 1$. By virtual queue dynamic (7.9), we have

$$Q(t_0 + 1) = \min\{Q(t_0) + e(t_0 + 1) - \sum_{i=1}^{n} p_i(t_0 + 1), 0\}$$

Adding $E^{\max}$ on both sides yields

$$\begin{aligned}
&Q(t_0 + 1) + E^{\max} \\
=& \min\{Q(t_0) + e(t_0 + 1) - \sum_{i=1}^{n} p_i(t_0 + 1) + E^{\max}, E^{\max}\} \\
\overset{(a)}{=}& \min\{E(t_0) + e(t_0 + 1) - \sum_{i=1}^{n} p_i(t_0 + 1), E^{\max}\} \\
\overset{(b)}{=}& E(t_0 + 1)
\end{aligned}$$

where (a) follows from the induction hypothesis $E(t_0) = Q(t_0) + E^{\max}$ and (b) follows from the energy queue dynamic (7.3). Thus, (7.31) holds for $t = t_0 + 1$.

Now observe

$$\begin{aligned} E(t_0+1) &= Q(t_0+1) + E^{\max} \\ &\overset{(a)}{\geq} E^{\max} - Q^l \\ &\geq p^{\max} \\ &\overset{(b)}{\geq} \sum_{i=1}^{n} p_i(t_0+2) \end{aligned}$$

where (a) follows from the fact that $Q(t) \geq -Q^l, \forall t \in \{0,1,2,\ldots\}$ by Theorem 7.2; (b) holds since sum power is never more than $p^{\max}$. Thus, (7.32) holds for $t = t_0 + 1$.

Thus, this theorem follows by induction. □

7.3.5 Utility Optimality and Battery Capacity Tradeoff

By Theorem 7.1, Algorithm 7.1 is guaranteed to attain a utility within an $O(1/V)$ distance to the optimal utility U^*. To obtain an $O(\epsilon)$-optimal utility, we can choose $V = \lceil 1/\epsilon \rceil$, where $\lceil x \rceil$ represents the smallest integer no less than x. In this case, Q^l defined in (7.3) is order $O(V)$. By Theorem 7.3,we need the battery capacity $E^{\max} \geq Q^l + p^{\max} = O(V) = O(1/\epsilon)$ to satisfy the energy availability constraint. Thus, there is a $[O(\epsilon), O(1/\epsilon)]$ tradeoff between the utility optimality and the required battery capacity.

7.3.6 Extensions

Thus far, we have assumed that $\omega(t)$ is known with one slot delay, i.e., at the end of slot t, or equivalently, at the beginning of slot $t+1$. In fact, if $\omega(t)$ is observed with t_0 slot delay (at the end of slot $t + t_0 - 1$), we can modify Algorithm 7.1 by initializing $\mathbf{p}(\tau) = \mathbf{0}, \tau \in \{1, 2, \ldots, t_0\}$ and updating $Q(t - t_0 + 1) = \min\{Q(t - t_0) + e(t - t_0 + 1) - \sum_{i=1}^{n} p_i(t - t_0 + 1), 0\}$, $\mathbf{p}(t+1) = \text{Proj}_{\mathcal{P}}\{\mathbf{p}(t - t_0 + 1) + \frac{1}{V}\nabla_{\mathbf{p}} U(\mathbf{p}(t - t_0 + 1); \omega(t - t_0 + 1)) + \frac{1}{V^2} Q(t - t_0 + 1)\mathbf{1}\}$ at the end of each slot $t \in \{t_0, t_0 + 1, \ldots\}$. By extending the analysis in this section (from a $t_0 = 1$ version to a general t_0 version), a similar $[O(\epsilon), O(1/\epsilon)]$ tradeoff can be established.

7.4 Numerical Experiment

In this section, we consider an energy harvesting wireless device transmitting over 2 subbands whose channel strength is represented by $s_1(t)$ and $s_2(t)$, respectively. Our goal is to decide the power action $\mathbf{p}(t)$ to maximize the utility/throughput given by (7.1). Let $\mathcal{P} = \{\mathbf{p} : p_1 + p_2 \leq 5, p_1 \geq 0, p_2 \geq 0\}$. Let harvested energy $e(t)$ satisfy the uniform distribution over interval $[0, 3]$. Assume both subbands are Rayleigh fading channels where $s_1(t)$ follows the Rayleigh distribution with parameter $\sigma = 0.5$ truncated in the range $[0, 4]$ and $s_2(t)$ follows the Rayleigh distribution with parameter $\sigma = 1$ truncated in the range $[0, 4]$.

By assuming the perfect knowledge of distributions, we solve the deterministic problem (7.4)-(7.6) and obtain $U^* = 1.0391$. To verify the performance proven in Theorems 7.1 and 7.3, we run Algorithm 7.1 with $V \in \{5, 10, 20, 40\}$ and $E(0) = E^{\max} = Q^l + p^{\max}$ over 1000 independent simulation runs. In all the simulation runs, the power actions yielded by Algorithm 7.1 always satisfy the energy availability constraints. We also plot the averaged utility performance in Figure 7.1, where the y-axis is the running average of expected utility. Figure 7.1 shows that the utility performance can approach U^* by using larger V parameter.

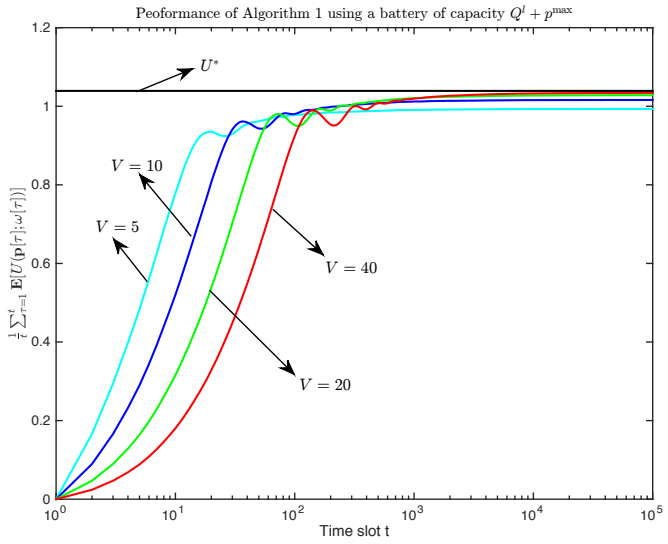

Figure 7.1: Utility performance (averaged over 1000 independent simulation runs) of Algorithm 7.1 with $E(0) = E^{\max} = Q^l + p^{\max}$ for different V.

In practice, it is possible that for a given V, the battery capacity $E^{\max} = Q^l + p^{\max}$ required in Theorem 7.3 is too large. If we run Algorithm 7.1 with small capacity batteries such that

$\sum_{i=1}^{n} p_i(t+1) \geq E(t)$ for certain slot t, a reasonable choice is to scale down $\mathbf{p}(t+1)$ by (7.11) and use $\hat{\mathbf{p}}(t+1)$ as the power action. Now, we run simulations by fixing $V = 40$ in Algorithm 7.1 and test its performance with small capacity batteries. By Theorem 7.3, the required battery capacity to ensure energy availability is $E^{\max} = 685$. In our simulations, we choose small $E^{\max} \in \{10, 20, 50\}$ and $E(0) = 0$, i.e., the battery is initially empty. If $\mathbf{p}(t+1)$ from Algorithm 7.1 violates energy availability constraint (7.2), we use $\hat{\mathbf{p}}(t+1)$ from (7.11) as the true power action that is enforced to satisfy (7.2) and update the energy backlog by $E(t+1) = \min\{E(t) - \sum_{i=1}^{n} \hat{p}_i(t+1) + e(t+1), E^{\max}\}$. Figure 7.2 plots the utility performance of Algorithm 7.1 in this practical scenario and shows that even with small capacity batteries, Algorithm 7.1 still achieves a utility close to U^*. This further demonstrates the superior performance of our algorithm.

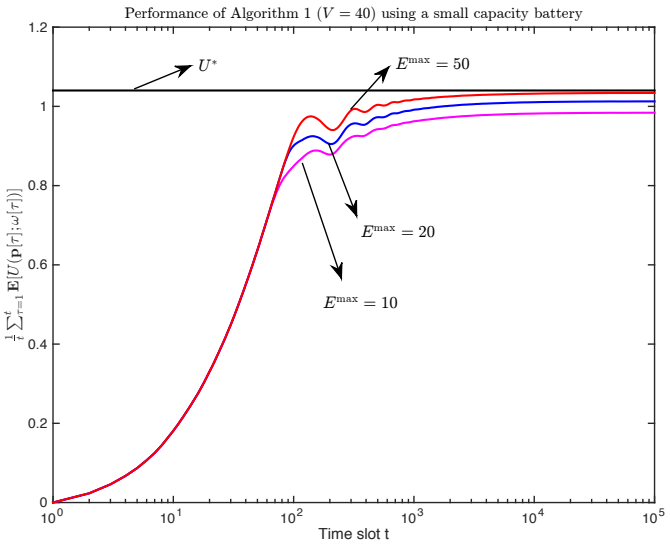

Figure 7.2: Utility performance (averaged over 1000 independent simulation runs) of Algorithm 7.1 with $V = 40$ for different $E^{\max}$.

7.5 Chapter Summary

This chapter develops a new learning aided power control algorithm for energy harvesting devices, without requiring the current system state or the distribution information. This new algorithm can achieve an $O(\epsilon)$ optimal utility by using a battery with capacity $O(1/\epsilon)$.

Chapter 8

Dynamic Transmit Covariance Design in MIMO Fading Systems With Unknown Channel Distributions and Inaccurate Channel State Information

During the past decade, the multiple-input multiple-output (MIMO) technique has been recognized as one of the most important techniques for increasing the capabilities of wireless communication systems. In the wireless fading channel, where the channel changes over time, the problem of transmit covariance design is to determine the transmit covariance of the transmitter to maximize the capacity subject to both long term and short term power constraints. It is often reasonable to assume that instantaneous channel state information (CSI) is available at the receiver through training. Most works on transmit covariance design in MIMO fading systems also assume that statistical information about the channel state, referred to as channel distribution information (CDI), is available at the transmitter. Under the assumption of perfect channel state information at the receiver (CSIR) and perfect channel distribution information at the transmitter (CDIT), prior work on transmit covariance design in point-to-point MIMO fading systems can be grouped into two categories:

- Instantaneous channel state information at the transmitter: In the ideal case of perfect[1] CSIT, optimal transmit covariance design for MIMO links with both long term and short term power constraints is a water-filling solution [Tel99]. Computation of water-levels involves a one-dimensional integral equation for fading channels with independent and identically distributed (i.i.d.) Rayleigh entries or a multi-dimensional integral equation for

[1] In this paper, CSIT is said to be "perfect" if it is both instantaneous (i.e., has no delay) and accurate.

general fading channels [JP03]. The involved multi-dimensional integration equation is in general intractable and can only be approximately solved with numerical algorithms with huge complexity. MIMO fading systems with dynamic CSIT is considered in [VP07].

- No CSIT: If CSIT is unavailable, the optimal transmit covariance design is in general still open. If the channel matrix has i.i.d. Rayleigh entries, then the optimal transmit covariance is known to be the identity transmit covariance scaled to satisfy the power constraint [Tel99]. The optimal transmit covariance in MIMO fading channels with correlated Rayleigh entries is obtained in [JVG01, JB04]. The transmit covariance design in MIMO fading channels is further considered in [VLS05] under a more general channel correlation model.

These prior works rely on accurate CDIT and/or on restrictive channel distribution assumptions. It can be difficult to accurately estimate the CDI, especially when there are complicated correlations between entries in the channel matrix. Solutions that base decisions on CDIT can be suboptimal due to mismatches. Work [PCL03] considers MIMO fading channels without CDIT and aims to find the transmit covariance to maximize the worst channel capacity using a game theoretical approach rather than solve the original ergodic capacity maximization problem. In contrast, the current chapter proposes algorithms that do not require prior knowledge of the channel distribution, yet perform arbitrarily close to the optimal value of the ergodic capacity maximization that can be achieved by having CDI knowledge. The results in this chapter are originally developed in our papers [YN16a, YN17a].

In time-division duplex (TDD) systems with symmetric wireless channels, the CSI can be measured directly at the transmitter using the unlink channel. However, in frequency-division duplex (FDD) scenarios and other scenarios without channel symmetry, the CSI must be measured at the receiver, quantized, and reported back to the transmitter with a time delay [TV05].

Depending on the measurement delay in TDD systems or the overall channel acquisition delay in FDD systems, the CSIT can be instantaneous or delayed. In general, the CSIT can also be inaccurate due to the measurement, quantization or feedback error. This paper first considers the instantaneous (but possibly inaccurate) CSIT case and develops an algorithm that does not require CDIT. This algorithm can achieve a utility within $O(\delta)$ of the best utility that can be achieved with CDIT and perfect CSIT, where δ is the inaccuracy measure of CSIT. This further implies that accurate instantaneous CSIT (with $\delta = 0$) is almost as good as having both CDIT and accurate instantaneous CSIT.

Next, the case of delayed (but possibly inaccurate) CSIT is considered and a fundamentally different algorithm is developed for that case. The latter algorithm again does not use CDIT, but achieves a utility within $O(\delta)$ of the best utility that can be achieved even with CDIT, where δ is the inaccuracy measure of CSIT. This further implies that delayed but accurate CSIT (with $\delta = 0$) is almost as good as having CDIT.

Related Work and Our Contributions

In the instantaneous (and possibly inaccurate) CSIT case, the proposed dynamic transmit covariance design extends the general drift-plus-penalty algorithm for stochastic network optimization [Nee03, Nee10] to deal with inaccurate observations of system states. In this MIMO context, the current chapter shows the algorithm provides strong sample path convergence time guarantees. The dynamic of the drift-plus-penalty algorithm is similar to that of the stochastic dual subgradient algorithm, although the optimality analysis and performance bounds are different. The stochastic dual subgradient algorithm has been applied to optimization in wireless fading channels without CDI, e.g., downlink power scheduling in single antenna cellular systems [LMS06], power allocation in single antenna broadcast OFDM channels [Rib10], scheduling and resource allocation in random access channels [HR11], transmit covariance design in multi-carrier MIMO networks [LHSS09].

In the delayed (and possibly inaccurate) CSIT case, the situation is similar to the scenario of online convex optimization [Zin03] except that we are unable to observe true history reward functions due to channel error. The proposed dynamic power allocation policy can be viewed as an online algorithm with inaccurate history information. The current chapter analyzes the performance loss due to CSIT inaccuracy and provides strong sample path convergence time guarantees of this algorithm. The analysis in this MIMO context can be extended to more general online convex optimization with inaccurate history information. Online optimization has been applied in power allocation in wireless fading channels without CDIT and with delayed and accurate CSIT, e.g., suboptimal online power allocation in single antenna single user channels [BLEM$^+$09], suboptimal online power allocation in single antenna multiple user channels [BLEM$^+$10]. Online transmit covariance design in MIMO systems with inaccurate CSIT is also considered in recent works [SMT15, MM16, MB16]. The online algorithms in [SMT15, MM16, MB16] follow either a matrix exponential learning scheme or an online projected gradient scheme. However, all of

these works assume that the imperfect CSIT is unbiased, i.e., expected CSIT error conditional on observed previous CSIT is zero. This assumption of imperfect CSIT is suitable when modeling the CSIT measurement error or feedback error but cannot capture the CSI quantization error. In contrast, the current chapter only requires that CSIT error is bounded.

8.1 Signal Model and Problem Formulation

8.1.1 Signal Model

Consider a point-to-point MIMO block fading channel with N_T transmit antennas and N_R receive antennas. In a block fading channel model, the channel matrix remains constant at each block and changes from block to block in an independent and identically distributed (i.i.d.) manner. Throughout this chapter, each block is also called a slot and is assigned an index $t \in \{0, 1, 2, \ldots\}$. At each slot t, the received signal [Tel99] is described by

$$\mathbf{y}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{z}(t)$$

where $t \in \{0, 1, 2, \ldots\}$ is the time index, $\mathbf{z}(t) \in \mathbb{C}^{N_R}$ is the additive noise vector, $\mathbf{x}(t) \in \mathbb{C}^{N_T}$ is the transmitted signal vector, $\mathbf{H}(t) \in \mathbb{C}^{N_R \times N_T}$ is the channel matrix, and $\mathbf{y}(t) \in \mathbb{C}^{N_R}$ is the received signal vector. Assume that noise vectors $\mathbf{z}(t)$ are i.i.d. normalized circularly symmetric complex Gaussian random vectors with $\mathbb{E}[\mathbf{z}(t)\mathbf{z}^{\mathsf{H}}(t)] = \mathbf{I}_{N_R}$, where $\mathbf{I}_{N_R}$ denotes an $N_R \times N_R$ identity matrix.[2] Note that channel matrices $\mathbf{H}(t)$ are i.i.d. across slot t and have a fixed but arbitrary probability distribution, possibly one with correlations between entries of the matrix. Assume there is a constant $B > 0$ such that $\|\mathbf{H}\|_F \leq B$ with probability one, where $\|\cdot\|_F$ denotes the Frobenius norm.[3] Recall that the Frobenius norm of a complex $m \times n$ matrix $\mathbf{A} = (a_{ij})$ is

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\mathrm{tr}(\mathbf{A}^{\mathsf{H}}\mathbf{A})} \tag{8.1}$$

where $\mathbf{A}^{\mathsf{H}}$ is the Hermitian transpose of $\mathbf{A}$ and $\mathrm{tr}(\cdot)$ is the trace operator.

[2] If the size of the identity matrix is clear, we often simply write $\mathbf{I}$.

[3] A bounded Frobenius norm always holds in the physical world because the channel attenuates the signal. Particular models such as Rayleigh and Rician fading violate this assumption in order to have simpler distribution functions [BA03].

Assume that the receiver can track $\mathbf{H}(t)$ exactly at each slot t and hence has perfect CSIR. In practice, CSIR is obtained by sending designed training sequences, also known as pilot sequences, which are commonly known to both the transmitter and the receiver, such that the channel matrix $\mathbf{H}(t)$ can be estimated at the receiver [TV05]. CSIT is obtained in different ways in different wireless systems. In TDD systems, the transmitter exploits channel reciprocity and use the measured uplink channel as approximated CSIT. In FDD systems, the receiver creates a quantized version of CSI, which is a function of $\mathbf{H}(t)$, and reports back to the transmitter after a certain amount of delay. In general, there are two possibilities of CSIT availabilities:

- **Instantaneous CSIT Case**: In TDD systems or FDD systems where the measurement, quantization and feedback delays are negligible with respect to the channel coherence time, an approximate version $\widetilde{\mathbf{H}}(t)$ for the true channel $\mathbf{H}(t)$ is known at the transmitter at each time slot t.

- **Delayed CSIT Case**: In FDD systems with a large CSIT acquisition delay, the transmitter only knows $\widetilde{\mathbf{H}}(t-1)$, which is an approximate version of channel $\mathbf{H}(t-1)$, and does not know $\mathbf{H}(t)$ at each time slot t.[4]

In both cases, we assume the CSIT inaccuracy is bounded, i.e., there exists $\delta > 0$ such that $\|\widetilde{\mathbf{H}}(t) - \mathbf{H}(t)\|_F \leq \delta$ for all t.

8.1.2 Problem Formulation

At each slot t, if the channel matrix is $\mathbf{H}(t)$ and the transmit covariance is $\mathbf{Q}(t)$, then the MIMO capacity is given by [Tel99]:

$$\log\det(\mathbf{I} + \mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t))$$

[4]In general, the dynamic transmit covariance design developed in this chapter can be extended to deal with arbitrary CSIT acquisition delay as discussed in Section 8.3.3. For the simplicity of presentations, we assume the CSIT acquisition delay is always one slot in this chapter.

where $\det(\cdot)$ denotes the determinant operator of matrices. The (long term) average capacity[5] of the MIMO block fading channel [Gol05] is given by

$$\mathbb{E}_{\mathbf{H}}\big[\log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}})\big]$$

where $\mathbf{Q}$ can adapt to $\mathbf{H}$ when CSIT is available and is a constant matrix when CSIT is unavailable. Consider two types of power constraints at the transmitter: A long term average power constraint $\mathbb{E}_{\mathbf{H}}[\mathrm{tr}(\mathbf{Q})] \leq \bar{P}$ and a short term power constraint $\mathrm{tr}(\mathbf{Q}) \leq P$ enforced at each slot. The long term constraint arises from battery or energy limitations while the short term constraint is often due to hardware or regulation limitations.

If CSIT is available, the problem is to choose $\mathbf{Q}$ as a (possibly random) function of the observed $\mathbf{H}$ to maximize the (long term) average capacity subject to both power constraints:

$$\max_{\mathbf{Q}(\mathbf{H})} \quad \mathbb{E}_{\mathbf{H}}\big[\log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}(\mathbf{H})\mathbf{H}^{\mathsf{H}})\big] \tag{8.2}$$

$$\text{s.t.} \quad \mathbb{E}_{\mathbf{H}}[\mathrm{tr}(\mathbf{Q}(\mathbf{H}))] \leq \bar{P}, \tag{8.3}$$

$$\mathbf{Q}(\mathbf{H}) \in \mathcal{Q}, \forall \mathbf{H}, \tag{8.4}$$

where $\mathcal{Q}$ is a set that enforces the short term power constraint:

$$\mathcal{Q} = \big\{\mathbf{Q} \in \mathbb{S}_{+}^{N_T} : \mathrm{tr}(\mathbf{Q}) \leq P\big\} \tag{8.5}$$

where $\mathbb{S}_{+}^{N_T}$ denotes the $N_T \times N_T$ positive semidefinite matrix space. To avoid trivialities, we assume that $P \geq \bar{P}$. In (8.2)-(8.4), we use notation $\mathbf{Q}(\mathbf{H})$ to emphasize that $\mathbf{Q}$ can depend on $\mathbf{H}$, i.e., adapt to channel realizations. Under the long term power constraint, the optimal power allocation should be opportunistic, i.e., use more power over good channel realizations and less

[5]The expression $\mathbb{E}_{\mathbf{H}}\big[\log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}})\big]$ is also known as the ergodic capacity. In fast fading channels where the channel coherence time is smaller than the codeword length, ergodic capacity can be attained if each codeword spans across sufficiently many channel blocks. In slow fading channels where the channel coherence time is larger than the codeword length, ergodic capacity can be attained by adapting both transmit covariances and data rates to the CSIT of each channel block (see [LK06] for related discussions). In slow fading channels, the ergodic capacity is essentially the long term average capacity since it is asymptotically equal to the average capacity of each channel block (by the law of large numbers). Note that another concept "outage capacity" is sometimes considered for slow fading channels when there is no rate adaptation and the data rate is constant regardless of channel realizations (In this case, the data rate can be larger than the block capacity for poor channel realizations such that "outage" occurs). In this chapter, we have both transmit covariance design and rate adaptation; and hence consider "ergodic capacity".

power over poor channel realizations. It is known that opportunistic power allocation provides a significant capacity gain in low SNR regimes and a marginal gain in high SNR regimes compared with fixed power allocation [ÖLR14].

Without CSIT, the optimal transmit covariance design problem is different, given as follows.

$$\max_{\mathbf{Q}} \quad \mathbb{E}_{\mathbf{H}}\big[\log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}})\big] \tag{8.6}$$

$$\text{s.t.} \quad \mathbb{E}_{\mathbf{H}}[\mathrm{tr}(\mathbf{Q})] \leq \bar{P}, \tag{8.7}$$

$$\mathbf{Q} \in \mathcal{Q}, \tag{8.8}$$

where set $\mathcal{Q}$ is defined in (8.5). Again assume $P \geq \bar{P}$. Since the instantaneous CSIT is unavailable, the transmit covariance cannot adapt to $\mathbf{H}$. By the convexity of this problem and Jensen's inequality, a randomized $\mathbf{Q}$ is useless. It suffices to consider a constant $\mathbf{Q}$. Since $P \geq \bar{P}$, this implies the problem is equivalent to a problem that removes the constraint (8.7) and that changes the constraint (8.8) to:

$$\mathbf{Q} \in \widetilde{\mathcal{Q}} = \{\mathbf{Q} \in \mathbb{S}_{+}^{N_T} : \mathrm{tr}(\mathbf{Q}) \leq \bar{P}\}$$

The problems (8.2)-(8.4) and (8.6)-(8.8) are fundamentally different and have different optimal objective function values. Most existing works [JP03, JVG01, JB04, VLS05] on MIMO fading channels can be interpreted as solutions to either of the above two stochastic optimization under specific channel distributions. Moreover, those works require perfect channel distribution information (CDI). In this chapter, the above two stochastic optimization problems are solved via dynamic algorithms that works for arbitrary channel distributions and does not require any CDI. The algorithms are different for the two cases, and use different techniques.

8.2 Instantaneous CSIT Case

Consider the case of instantaneous but inaccurate CSIT where at each slot $t \in \{0, 1, 2, \ldots\}$, channel $\mathbf{H}(t)$ is unknown and only an approximate version $\widetilde{\mathbf{H}}(t)$ is known. In this case, the problem (8.2)-(8.4) can be interpreted as a stochastic optimization problem where channel $\mathbf{H}(t)$ is the instantaneous system state and transmit covariance $\mathbf{Q}(t)$ is the control action at each slot t. This is similar to the scenario of stochastic optimization with i.i.d. time-varying system states, where the decision maker chooses an action based on the observed instantaneous system state

at each slot such that time average expected utility is maximized and the time average expected constraints are guaranteed. The *drift-plus-penalty (DPP)* technique reviewed in Chapter 1 is a mature framework to solve stochastic optimization without distribution information of system states.

This is different from the conventional stochastic optimization considered by the DPP technique because at each slot t, the true "system state" $\mathbf{H}(t)$ is unavailable and only an approximate version $\widetilde{\mathbf{H}}(t)$ is known. Nevertheless, a modified version of the standard DPP algorithm is developed in Algorithm 8.1.

Algorithm 8.1 Dynamic Transmit Covariance Design with Instantaneous CSIT

Let $V > 0$ be a constant parameter and $Z(0) = 0$. At each time $t \in \{0, 1, 2, \ldots\}$, observe $\widetilde{\mathbf{H}}(t)$ and $Z(t)$. Then do the following:

- Choose transmit covariance $\mathbf{Q}(t) \in \mathcal{Q}$ to solve :

$$\max_{\mathbf{Q}\in\mathcal{Q}}\{V \log\det(\mathbf{I} + \widetilde{\mathbf{H}}(t)\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}(t) - Z(t)\mathrm{tr}(\mathbf{Q})\}.$$

- Update $Z(t+1) = \max\{0, Z(t) + \mathrm{tr}(\mathbf{Q}(t)) - \bar{P}\}$.

In Algorithm 8.1, a *virtual queue* $Z(t)$ with $Z(0) = 0$ and with update $Z(t+1) = \max\{0, Z(t) + \mathrm{tr}(\mathbf{Q}(t)) - \bar{P}\}$ is introduced to enforce the average power constraint (8.3) and can be viewed as the "queue backlog" of long term power constraint violations since it increases at slot t if the power consumption at slot t is larger than $\bar{P}$ and decreases otherwise. The next Lemma relates $Z(t)$ and the average power consumption.

Lemma 8.1. *Under Algorithm 8.1, we have*

$$\frac{1}{t}\sum_{\tau=0}^{t-1} tr(\mathbf{Q}(\tau)) \leq \bar{P} + \frac{Z(t)}{t}, \quad \forall t > 0.$$

Proof. Fix $t > 0$. For all slots $\tau \in \{0, 1, \ldots, t-1\}$, the update for $Z(\tau)$ satisfies $Z(\tau+1) = \max\{0, Z(\tau) + \mathrm{tr}(\mathbf{Q}(\tau)) - \bar{P}\} \geq Z(\tau) + \mathrm{tr}(\mathbf{Q}(\tau)) - \bar{P}$. Rearranging terms gives: $\mathrm{tr}(\mathbf{Q}(\tau)) \leq \bar{P} + Z(\tau+1) - Z(\tau)$. Summing over $\tau \in \{0, \ldots, t-1\}$ and dividing by factor t gives:

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\mathrm{tr}(\mathbf{Q}(\tau)) \leq \bar{P} + \frac{Z(t) - Z(0)}{t} \overset{(a)}{=} \bar{P} + \frac{Z(t)}{t}$$

where (a) follows from $Z(t) = 0$. □

For each slot $t \in \{0, 1, 2, \ldots\}$ define the *reward* $R(t)$:

$$R(t) = \log \det(\mathbf{I} + \mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t)). \tag{8.9}$$

Define R^{opt} as the optimal average utility in (8.2). The value R^{opt} depends on the (unknown) distribution for $\mathbf{H}(t)$. Fix $\epsilon > 0$ and define $V = \max\{\bar{P}^2, (P - \bar{P})^2\}/(2\epsilon)$. If $\widetilde{\mathbf{H}}(t) = \mathbf{H}(t), \forall t$, regardless of the distribution of $\mathbf{H}(t)$, the standard DPP technique [Nee10] ensures:

$$\frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}[R(\tau)] \geq R^{\text{opt}} - \epsilon, \qquad \forall t > 0 \tag{8.10}$$

$$\lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}[\text{tr}(\mathbf{Q}(\tau))] \leq \bar{P} \tag{8.11}$$

This holds for arbitrarily small values of $\epsilon > 0$, and so the algorithm comes arbitrarily close to optimality. However, the above is true only if $\widetilde{\mathbf{H}}(t) = \mathbf{H}(t), \forall t$.

The development and analysis of Algorithm 8.1 extends the DPP technique in two aspects:

- At each slot t, the standard drift-plus-penalty technique requires accurate "system state" $\mathbf{H}(t)$ and cannot deal with inaccurate "system state" $\widetilde{\mathbf{H}}(t)$. In contrast, Algorithm 8.1 works with $\widetilde{\mathbf{H}}(t)$. The next subsections show that the performance of Algorithm 8.1 degrades linearly with respect to CSIT inaccuracy measure δ. If $\delta = 0$, then (8.10) is recovered.

- Inequality (8.11) only treats infinite horizon time average expected power. The next subsections show that Algorithm 8.1 can guarantee

$$\frac{1}{t}\sum_{\tau=0}^{t-1} \text{tr}(\mathbf{Q}(\tau)) \leq \bar{P} + \frac{(B+\delta)^2 \max\{\bar{P}^2, (P-\bar{P})^2\} + 2\epsilon(P-\bar{P})}{2\epsilon t}$$

  for all $t > 0$. This sample path guarantee on average power consumption is much stronger than (8.11). In fact, (8.11) is recovered by taking expectation and taking limit $t \to \infty$.

8.2.1 Transmit Covariance Update in Algorithm 8.1

This subsection shows the $\mathbf{Q}(t)$ selection in Algorithm 8.1 has an (almost) closed-form solution. The convex program involved in the transmit covariance update of Algorithm 8.1 is in the

form

$$\max_{\mathbf{Q}} \quad \log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}) - \frac{Z}{V}\mathrm{tr}(\mathbf{Q}) \tag{8.12}$$

$$\text{s.t.} \quad \mathrm{tr}(\mathbf{Q}) \leq P \tag{8.13}$$

$$\mathbf{Q} \in \mathbb{S}_{+}^{N_T} \tag{8.14}$$

This convex program is similar to the conventional problem of transmit covariance design with a deterministic channel $\mathbf{H}$, except that the objective (8.12) has an additional penalty term $-\frac{Z}{V}\mathrm{tr}(\mathbf{Q})$. It is well known that, without this penalty term, the solution is to diagonalize the channel matrix and allocate power over eigen-modes according to a water-filling technique [Tel99]. The next lemma summarizes that the optimal solution to the problem (8.12)-(8.14) has a similar structure.

Lemma 8.2. *Consider singular value decomposition (SVD)* $\mathbf{H}^{\mathsf{H}}\mathbf{H} = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$*, where* $\mathbf{U}$ *is a unitary matrix and* $\mathbf{\Sigma}$ *is a diagonal matrix with non-negative entries* $\sigma_1, \ldots, \sigma_{N_T}$*. Then the optimal solution to* (8.12)-(8.14) *is given by* $\mathbf{Q}^* = \mathbf{U}^{\mathsf{H}}\mathbf{\Theta}^*\mathbf{U}$*, where* $\mathbf{\Theta}^*$ *is a diagonal matrix with entries* $\theta_1^*, \ldots, \theta_{N_T}^*$ *given by:*

$$\theta_i^* = \max\big\{0, \frac{1}{\mu^* + Z/V} - \frac{1}{\sigma_i}\big\}, \quad \forall i \in \{1, \ldots, N_T\},$$

where μ^* *is chosen such that* $\sum_{i=1}^{N_T}\theta_i^* \leq P$*,* $\mu^* \geq 0$ *and* $\mu^*\big(\sum_{i=1}^{N_T}\theta_i^* - P\big) = 0$*. The exact* μ^* *can be determined using Algorithm 8.2 with complexity* $O(N_T \log N_T)$*.*

Proof. The proof is a simple extension of the classical proof for the optimal transmit covariance in deterministic MIMO channels, e.g. Section 3.2 in [Tel99], to deal with the additional penalty term $-\frac{Z}{V}\mathrm{tr}(\mathbf{Q})$. See Section 8.7.2 for a complete proof. □

The complexity of Algorithm 8.2 is dominated by the sorting of all σ_i in step (2). Recall that the water-filling solution of power allocation in multiple parallel channels can also be found by an exact algorithm (see Section 6 in [PL03]), which is similar to Algorithm 8.2. The main difference is that Algorithm 8.2 has a first step to verify if $\mu^* = 0$. This is because unlike the power allocation in multiple parallel channels, where the optimal solution always uses full power, the optimal solution to the problem (8.12)-(8.14) may not use full power for large Z due to the

Algorithm 8.2 Algorithm to Solve Problem (8.12)-(8.14)

1. Check if $\sum_{i=1}^{N_T} \max\{0, \frac{1}{Z/V} - \frac{1}{\sigma_i}\} \leq P$ holds. If yes, let $\mu^* = 0$ and $\theta_i^* = \max\{0, \frac{1}{Z/V} - \frac{1}{\sigma_i}\}, \forall i \in \{1, 2, \ldots, N_T\}$ and terminate the algorithm; else, continue to the next step.
2. Sort all $\sigma_i, \in \{1, 2, \ldots, N_T\}$ in a decreasing order π such that $\sigma_{\pi(1)} \geq \sigma_{\pi(2)} \geq \cdots \geq \sigma_{\pi(N_T)}$. Define $S_0 = 0$.
3. For $i = 1$ to N_T
   - Let $S_i = S_{i-1} + \frac{1}{\sigma_{\pi(i)}}$. Let $\mu^* = \frac{i}{S_i + P} - \frac{Z}{V}$.
   - If $\mu^* \geq 0$, $\frac{1}{\mu^* + Z/V} - \frac{1}{\sigma_{\pi(i)}} > 0$ and $\frac{1}{\mu^* + Z/V} - \frac{1}{\sigma_{\pi(i+1)}} \leq 0$, then terminate the loop; else, continue to the next iteration in the loop.
4. Let $\theta_i^* = \max\left\{0, \frac{1}{\mu^* + Z/V} - \frac{1}{\sigma_i}\right\}, \forall i \in \{1, 2, \ldots, N_T\}$ and terminate the algorithm.

penalty term $-\frac{Z}{V}\text{tr}(\mathbf{Q})$ in objective (8.12).

8.2.2 Performance of Algorithm 8.1

Define a Lyapunov function $L(t) = \frac{1}{2}Z^2(t)$ and its corresponding Lyapunov drift $\Delta(t) = L(t+1) - L(t)$. The expression $-\Delta(t) + VR(t)$ is called the DPP expression. The analysis of the standard drift-plus-penalty (DPP) algorithm with accurate "system states" relies on a bound of the DPP expression in terms of R^{opt} [Nee10]. The performance analysis of Algorithm 8.1, which can be viewed as a DPP algorithm based on inaccurate "system states", requires a new bound of the DPP expression in Lemma 8.3 and a new deterministic bound of virtual queue $Z(t)$ in Lemma 8.4.

Lemma 8.3. *Under Algorithm 8.1, we have*

$$-\mathbb{E}[\Delta(t)] + V\mathbb{E}[R(t)] \geq VR^{opt} - \frac{1}{2}\max\{\bar{P}^2, (P - \bar{P})^2\} - 2VP\sqrt{N_T}(2B + \delta)\delta,$$

where B, δ, N_T, P and $\bar{P}$ are defined in Section 8.1.1; and R^{opt} is the optimal average utility in the problem (8.2)-(8.4).

Proof. See Section 8.7.3. □

Lemma 8.4. *Under Algorithm 8.1, we have $Z(t) \leq V(B + \delta)^2 + (P - \bar{P}), \forall t > 0$, where B, δ, P and $\bar{P}$ are defined in Section 8.1.1.*

Proof. We first show that if $Z(t) \geq V(B+\delta)^2$, then Algorithm 8.1 chooses $\mathbf{Q}(t) = \mathbf{0}$. Consider $Z(t) \geq V(B+\delta)^2$. Let SVD $\widetilde{\mathbf{H}}^{\mathsf{H}}(t)\widetilde{\mathbf{H}}(t) = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$, where diagonal matrix $\mathbf{\Sigma}$ has non-negative diagonal entries $\sigma_1, \ldots, \sigma_{N_T}$. Note that $\forall i \in \{1, 2, \ldots, N_T\}$, $\sigma_i \overset{(a)}{\leq} \text{tr}(\widetilde{\mathbf{H}}^{\mathsf{H}}(t)\widetilde{\mathbf{H}}(t)) \overset{(b)}{=} \|\widetilde{\mathbf{H}}(t)\|_F^2 \leq (\|\mathbf{H}(t)\|_F + \|\widetilde{\mathbf{H}}(t) - \mathbf{H}(t)\|_F)^2 \leq (B+\delta)^2$ where (a) follows from $\text{tr}(\widetilde{\mathbf{H}}^{\mathsf{H}}(t)\widetilde{\mathbf{H}}(t)) = \sum_{i=1}^{N_T} \sigma_i$; and (b) follows from the definition of Frobenius norm. By Lemma 8.2, $\mathbf{Q}(t) = \mathbf{U}^{\mathsf{H}}\mathbf{\Theta}^*\mathbf{U}$, where $\mathbf{\Theta}^*$ is diagonal with entries $\theta_1^*, \ldots, \theta_{N_T}^*$ given by $\theta_i^* = \max\left\{0, \frac{1}{\mu^* + Z(t)/V} - \frac{1}{\sigma_i}\right\}$, where $\mu^* \geq 0$. Since $\sigma_i \leq (B+\delta)^2, \forall i \in \{1, 2, \ldots, N_T\}$, it follows that if $Z(t) \geq V(B+\delta)^2$, then $\frac{1}{\mu + Z(t)/V} - \frac{1}{\sigma_i} \leq 0$ for all $\mu \geq 0$ and hence $\theta_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}$. This implies that Algorithm 8.1 chooses $\mathbf{Q}(t) = 0$ by Lemma 8.2, which further implies that $Z(t+1) \leq Z(t)$ by the update equation of $Z(t+1)$.

On the other hand, if $Z(t) \leq V(B+\delta)^2$, then $Z(t+1)$ is at most $V(B+\delta)^2 + (P - \bar{P})$ by the update equation of $Z(t+1)$ and the short term power constraint $\text{tr}(\mathbf{Q}(t)) \leq P$. □

The next theorem summarizes the performance of Algorithm 8.1 and follows directly from Lemma 8.3 and Lemma 8.4.

Theorem 8.1. *Fix $\epsilon > 0$ and choose $V = \frac{\max\{\bar{P}^2, (P-\bar{P})^2\}}{2\epsilon}$ in Algorithm 8.1, then for all $t > 0$:*

$$\frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}[R(\tau)] \geq R^{opt} - \epsilon - \phi(\delta),$$
$$\frac{1}{t}\sum_{\tau=0}^{t-1} tr(\mathbf{Q}(\tau)) \leq \bar{P} + \frac{(B+\delta)^2 \max\{\bar{P}^2, (P-\bar{P})^2\} + 2\epsilon(P-\bar{P})}{2\epsilon t},$$

where $\phi(\delta) = 2P\sqrt{N_T}(2B+\delta)\delta$ satisfying $\phi(\delta) \to 0$ as $\delta \to 0$, i.e., $\phi(\delta) \in O(\delta)$; and B, δ, N_T, P and $\bar{P}$ are defined in Section 8.1.1. In particular, the average expected utility is within $\epsilon + \phi(\delta)$ of R^{opt} and the sample path time average power is within ϵ of its required constraint $\bar{P}$ whenever $t \geq \Omega(\frac{1}{\epsilon^2})$.

Proof.

Proof of the first inequality: Fix $t > 0$. For all slots $\tau \in \{0, 1, \ldots, t-1\}$, Lemma 8.3 guarantees that $\mathbb{E}[R(\tau)] \geq R^{\text{opt}} + \frac{1}{V}\mathbb{E}[\Delta(\tau)] - \frac{1}{2V}\max\{\bar{P}^2, (P-\bar{P})^2\} - 2P\sqrt{N_T}(2B+\delta)\delta$.

Summing over $\tau \in \{0, \ldots, t-1\}$ and dividing by t gives:

$$\begin{aligned}
&\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}[R(\tau)] \\
\geq & R^{\text{opt}} + \frac{1}{Vt}\sum_{\tau=0}^{t-1}\mathbb{E}[\Delta(\tau)] - \frac{1}{2V}\max\{\bar{P}^2, (P-\bar{P})^2\} - 2P\sqrt{N_T}(2B+\delta)\delta \\
\overset{(a)}{=} & R^{\text{opt}} + \frac{1}{2Vt}\big([\mathbb{E}[Z^2(t)] - \mathbb{E}[Z^2(0)]\big) - \frac{1}{2V}\max\{\bar{P}^2, (P-\bar{P})^2\} - 2P\sqrt{N_T}(2B+\delta)\delta \\
\overset{(b)}{\geq} & R^{\text{opt}} - \frac{1}{2V}\max\{\bar{P}^2, (P-\bar{P})^2\} - 2P\sqrt{N_T}(2B+\delta)\delta \\
\overset{(c)}{=} & R^{\text{opt}} - \epsilon - 2P\sqrt{N_T}(2B+\delta)\delta
\end{aligned}$$

where (a) follows from the definition that $\Delta(t) = \frac{1}{2}Z^2(t+1) - \frac{1}{2}Z^2(t)$ and by simplifying the telescoping sum $\sum_{\tau=0}^{t-1}\mathbb{E}[\Delta(\tau)]$; (b) follows from $Z(0) = 0$ and $Z(t) \geq 0$; and (c) follows by substituting $V = \frac{1}{2\epsilon}\max\{\bar{P}^2, (P-\bar{P})^2\}$.

Proof of the second inequality: Fix $t > 0$. By Lemma 8.1, we have

$$\begin{aligned}
\frac{1}{t}\sum_{\tau=0}^{t-1}\text{tr}(\mathbf{Q}(\tau)) &\leq \bar{P} + \frac{Z(t)}{t} \\
&\overset{(a)}{=} \bar{P} + \frac{(B+\delta)^2\max\{\bar{P}^2, (P-\bar{P})^2\} + 2\epsilon(P-\bar{P})}{2\epsilon t}
\end{aligned}$$

where (a) follows from Lemma 8.4 and $V = \frac{1}{2\epsilon}\max\{\bar{P}^2, (P-\bar{P})^2\}$. □

Theorem 8.1 provides a sample path guarantee on average power, which is much stronger than the guarantee in (8.11). It also shows that convergence time to reach an $\epsilon + O(\delta)$ approximate solution is $O(\frac{1}{\epsilon^2})$.

8.2.3 Discussion

It is shown that $Z(t)$ in the DPP algorithm is "attracted" to an optimal Lagrangian dual multiplier of an unknown deterministic convex program in [HMNK13]. In fact, if we have a good guess of this Lagrangian multiplier and initialize $Z(0)$ close to it, then Algorithm 8.1 has faster convergence. In addition, the performance bounds derived in Theorem 8.1 are not tightest possible. The proof of Lemma 8.3 involves many relaxations to derive bounds that are simple but can still enable Theorem 8.1 to show the effect of missing CDIT can be made arbitrarily small by choosing the algorithm parameter V properly and the performance degradation of CSIT

inaccuracy scales linearly with respect to δ. In fact, tighter but more complicated bounds are possible by refining the proof of Lemma 8.3.

A heuristic approach to solve the problem (8.2)-(8.4) without channel distribution information is to sample the channel for a large number of realizations and use the empirical distribution as an approximate distribution to solve the problem (8.2)-(8.4) directly. This approach has three drawbacks:

- For a scalar channel, the empirical distribution based on $O(\frac{1}{\epsilon^2})$ realizations is an ϵ approximation to the true channel distribution with high probability by the Dvoretzky-Kiefer-Wolfowitz inequality [Ser09]. However, for an $N_R \times N_T$ MIMO channel, the multi-dimensional empirical distribution requires $O(\frac{N_T^2 N_R^2}{\epsilon^2})$ samples to achieve an ϵ approximation of the true channel distribution [Dev77]. Thus, this approach does not scale well with the number of antennas.

- Even if the empirical distribution is accurate, the complexity of solving the problem (8.2)-(8.4) based on the empirical distribution is huge if the channel is from a continuous distribution. This is known as the curse of dimensionality of empirical methods for stochastic optimization due to the large sample size. In contrast, the complexity of Algorithm 8.1 is independent of the sample space.

- This approach is an offline method such that a large number of slots are wasted during the channel sampling process. In contrast, Algorithm 8.1 is an online method with performance guarantees for all slots.

Note that even if we assume the distribution of $\mathbf{H}(t)$ is known and $\mathbf{Q}^*(\mathbf{H})$ can be computed by solving the problem (8.2)-(8.4), the optimal policy $\mathbf{Q}^*(\mathbf{H})$ in general cannot achieve R^{opt} and can violate the long term power constraints when only the approximate versions $\widetilde{\mathbf{H}}(t)$ are known. For example, consider a MIMO fading system with two possible channel realizations $\mathbf{H}_1$ and $\mathbf{H}_2$ with equal probabilities. Suppose the average power constraint is $\bar{P} = 5$ and the optimal policy $\mathbf{Q}^*(\mathbf{H})$ satisfies $\text{tr}(\mathbf{Q}^*(\mathbf{H}_1)) = 8$ and $\text{tr}(\mathbf{Q}^*(\mathbf{H}_2)) = 2$. However, if $\widetilde{\mathbf{H}}_1 \neq \mathbf{H}_1$ and $\widetilde{\mathbf{H}}_2 \neq \mathbf{H}_2$, it can be hard to decide the transmit covariance based on $\widetilde{\mathbf{H}}_1$ or $\widetilde{\mathbf{H}}_2$ since the associations between $\widetilde{\mathbf{H}}_1$ and $\mathbf{H}_1$ (or between $\widetilde{\mathbf{H}}_2$ and $\mathbf{H}_2$) are unknown. In an extreme case when $\widetilde{\mathbf{H}}_1 = \widetilde{\mathbf{H}}_2 = \mathbf{H}_1$, if the transmitter uses $\mathbf{Q}^*(\widetilde{\mathbf{H}}(t))$ at each slot t, the average power constraint is violated and hence the transmit covariance scheme is infeasible. In contrast, Algorithm 8.1 can attain the performance

in Theorem 8.1 with inaccurate instantaneous CSIT and no CDIT.

8.3 Delayed CSIT Case

Consider the case of delayed and inaccurate CSIT. At the beginning of each slot $t \in \{0, 1, 2, \ldots\}$, channel $\mathbf{H}(t)$ is unknown and only quantized channels of previous slots $\widetilde{\mathbf{H}}(\tau), \tau \in \{0, 1, \ldots, t-1\}$ are known. This is similar to the scenario of online optimization where the decision maker selects $x(t) \in \mathcal{X}$ at each slot t to maximize an unknown reward function $f^t(x)$ based on the information of previous reward functions $f^\tau(x(\tau)), \tau \in \{0, 1, \ldots, t-1\}$. The goal is to minimize average regret $\frac{1}{t}\max_{x\in\mathcal{X}}\left[\sum_{\tau=0}^{t-1} f^\tau(x)\right] - \frac{1}{t}\sum_{\tau=0}^{t-1} f^\tau(x(\tau))$. The best possible average regret of online convex optimization with general convex reward functions is $O(\frac{1}{\sqrt{t}})$ [Zin03, HAK07].

The situation in the current chapter is different from conventional online optimization because at each slot t, the rewards of previous slots, i.e., $R(\tau) = \log\det(\mathbf{I} + \mathbf{H}(\tau)\mathbf{Q}(\tau)\mathbf{H}^{\mathsf{H}}(\tau)), \tau \in \{0, 1, \ldots, t-1\}$, are still unknown due to the fact that the reported channels $\widetilde{\mathbf{H}}(\tau)$ are approximate versions. Nevertheless, an online algorithm without using CDIT is developed in Algorithm 8.3.

Algorithm 8.3 Dynamic Transmit Covariance Design with Delayed CSIT

Let $\gamma > 0$ be a constant parameter and $\mathbf{Q}(0) \in \mathcal{Q}$ be arbitrary. At each time $t \in \{1, 2, \ldots\}$, observe $\widetilde{\mathbf{H}}(t-1)$ and do the following:

- Let $\widetilde{\mathbf{D}}(t-1) = \widetilde{\mathbf{H}}^{\mathsf{H}}(t-1)(\mathbf{I}_{N_R} + \widetilde{\mathbf{H}}(t-1)\mathbf{Q}(t-1)\widetilde{\mathbf{H}}^{\mathsf{H}}(t-1))^{-1}\widetilde{\mathbf{H}}(t-1)$. Choose transmit covariance
$$\mathbf{Q}(t) = \mathcal{P}_{\widetilde{\mathcal{Q}}}\big[\mathbf{Q}(t-1) + \gamma\widetilde{\mathbf{D}}(t-1)\big],$$
where $\mathcal{P}_{\widetilde{\mathcal{Q}}}[\cdot]$ is the projection onto convex set $\widetilde{\mathcal{Q}} = \{\mathbf{Q} \in \mathbb{S}_+^{N_T} : \mathrm{tr}(\mathbf{Q}) \le \bar{P}\}$.

Define $\mathbf{Q}^* \in \widetilde{\mathcal{Q}}$ as an optimal solution to the problem (8.6)-(8.8), which depends on the (unknown) distribution for $\mathbf{H}(t)$. Define

$$R^{\mathrm{opt}}(t) = \log\det(\mathbf{I} + \mathbf{H}(t)\mathbf{Q}^*\mathbf{H}^{\mathsf{H}}(t))$$

as the utility at slot t attained by $\mathbf{Q}^*$.

If the channel feedback is *accurate*, i.e., $\widetilde{\mathbf{H}}(t-1) = \mathbf{H}(t-1), \forall t \in \{1, 2, \ldots\}$, then $\widetilde{\mathbf{D}}(t-1)$ is the gradient of $R(t-1)$ at point $\mathbf{Q}(t-1)$. Fix $\epsilon > 0$ and take $\gamma = \epsilon$. The results in [Zin03]

ensure that, regardless of the distribution of $\mathbf{H}(t)$:

$$\frac{1}{t}\sum_{\tau=0}^{t-1} R(\tau) \geq \frac{1}{t}\sum_{\tau=0}^{t-1} R^{\text{opt}}(\tau) - \frac{2\bar{P}^2}{\epsilon t} - \frac{N_R B^4}{2}\epsilon, \forall t > 0 \tag{8.15}$$

$$\text{tr}(\mathbf{Q}(\tau)) \leq \bar{P}, \forall \tau \in \{0, 1, \ldots, t-1\} \tag{8.16}$$

The next subsections show that the performance of Algorithm 8.3 with *inaccurate* channels degrades linearly with respect to channel inaccuracy δ. If $\delta = 0$, then (8.15) and (8.16) are recovered.

8.3.1 Transmit Covariance Update in Algorithm 8.3

This subsection shows that the $\mathbf{Q}(t)$ selection in Algorithm 8.3 has an (almost) closed-form solution.

The projection operator involved in Algorithm 8.3 by definition is

$$\min \quad \frac{1}{2}\|\mathbf{Q} - \mathbf{X}\|_F^2 \tag{8.17}$$

$$\text{s.t.} \quad \text{tr}(\mathbf{Q}) \leq \bar{P} \tag{8.18}$$

$$\mathbf{Q} \in \mathbb{S}_+^{N_T} \tag{8.19}$$

where $\mathbf{X} = \mathbf{Q}(t-1) + \gamma\widetilde{\mathbf{D}}(t-1)$ is a Hermitian matrix at each slot t.

Without constraint $\text{tr}(\mathbf{Q}) \leq \bar{P}$, the projection of Hermitian matrix $\mathbf{X}$ onto the positive semidefinite cone $\mathbb{S}_+^n$ is simply taking the eigenvalue expansion of $\mathbf{X}$ and dropping terms associated with negative eigenvalues (see Section 8.1.1. in [BV04]). Work [BX05] considered the projection onto the intersection of the positive semidefinite cone $\mathbb{S}_+^n$ and an affine subspace given by $\{\mathbf{Q} : \text{tr}(\mathbf{A}_i\mathbf{Q}) = b_i, i \in \{1, 2, \ldots, p\}, \text{tr}(\mathbf{B}_j\mathbf{Q}) \leq d_j, j \in \{1, 2, \ldots, m\}\}$ and developed the dual-based iterative numerical algorithm to calculate the projection. The problem (8.17)-(8.19) is a special case, where the affine subspace is given by $\text{tr}(\mathbf{Q}) \leq \bar{P}$, of the projection considered in [BX05]. Instead of solving the problem (8.17)-(8.19) using numerical algorithms, the next lemma summarizes that the problem (8.17)-(8.19) has an (almost) closed-form solution.

Lemma 8.5. *Consider SVD* $\mathbf{X} = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$*, where* $\mathbf{U}$ *is a unitary matrix and* $\mathbf{\Sigma}$ *is a diagonal matrix with entries* $\sigma_1, \ldots, \sigma_{N_T}$*. Then the optimal solution to the problem* (8.17)-(8.19) *is given*

by $\mathbf{Q}^* = \mathbf{U}^{\mathsf{H}}\mathbf{\Theta}^*\mathbf{U}$*, where* $\mathbf{\Theta}^*$ *is a diagonal matrix with entries* $\theta_1^*, \ldots, \theta_{N_T}^*$ *given by,*

$$\theta_i^* = \max\{0, \sigma_i - \mu^*\}, \forall i \in \{1, 2, \ldots, N_T\},$$

where μ^* *is chosen such that* $\sum_{i=1}^{N_T} \theta_i^* \leq \bar{P}$*,* $\mu^* \geq 0$ *and* $\mu^*\big(\sum_{i=1}^{N_T} \theta_i^* - \bar{P}\big) = 0$*. The exact* μ^* *can be determined using Algorithm 8.4 with complexity* $O(N_T \log N_T)$*.*

Proof. See Section 8.7.4. □

Algorithm 8.4 Algorithm to Solve Problem (8.17)-(8.19)

1. Check if $\sum_{i=1}^{N_T} \max\{0, \sigma_i\} \leq \bar{P}$ holds. If yes, let $\mu^* = 0$ and $\theta_i^* = \max\{0, \sigma_i\}, \forall i \in \{1, 2, \ldots, N_T\}$ and terminate the algorithm; else, continue to the next step.
2. Sort all $\sigma_i, \in \{1, 2, \ldots, N_T\}$ in a decreasing order π such that $\sigma_{\pi(1)} \geq \sigma_{\pi(2)} \geq \cdots \geq \sigma_{\pi(N_T)}$. Define $S_0 = 0$.
3. For $i = 1$ to N_T
   - Let $S_i = S_{i-1} + \sigma_i$. Let $\mu^* = \frac{S_i - \bar{P}}{i}$.
   - If $\mu^* \geq 0$, $\sigma_{\pi(i)} - \mu^* > 0$ and $\sigma_{\pi(i+1)} - \mu^* \leq 0$, then terminate the loop; else, continue to the next iteration in the loop.
4. Let $\theta_i^* = \max\{0, \sigma_i - \mu^*\}, \forall i \in \{1, 2, \ldots, N_T\}$ and terminate the algorithm.

8.3.2 Performance of Algorithm 8.3

Define $\mathbf{D}(t-1) = \mathbf{H}^{\mathsf{H}}(t-1)(\mathbf{I}_{N_R} + \mathbf{H}(t-1)\mathbf{Q}(t-1)\mathbf{H}^{\mathsf{H}}(t-1))^{-1}\mathbf{H}(t-1)$, which is the gradient of $R(t-1)$ at point $\mathbf{Q}(t-1)$ and is unknown to the transmitter due to the unavailability of $\mathbf{H}(t-1)$. The next lemma relates $\widetilde{\mathbf{D}}(t-1)$ and $\mathbf{D}(t-1)$.

Lemma 8.6. *For all slots* $t \in \{1, 2, \ldots\}$*, we have*

1. $\|\mathbf{D}(t-1)\|_F \leq \sqrt{N_R}B^2$.
2. $\|\mathbf{D}(t-1) - \widetilde{\mathbf{D}}(t-1)\|_F \leq \psi(\delta)$*, where* $\psi(\delta) = \big(\sqrt{N_R}B + \sqrt{N_R}(B+\delta) + (B+\delta)^2 N_R \bar{P}(2B+\delta)\big)\delta$ *satisfying* $\psi(\delta) \to 0$ *as* $\delta \to 0$*, i.e.,* $\psi(\delta) \in O(\delta)$*.*
3. $\|\widetilde{\mathbf{D}}(t-1)\|_F \leq \psi(\delta) + \sqrt{N_R}B^2$

where B, δ, N_R, N_T, P *and* $\bar{P}$ *are defined in Section 8.1.1*

Proof. See Section 8.7.5. □

The next theorem summarizes the performance of Algorithm 8.3.

Theorem 8.2. *Fix $\epsilon > 0$ and define $\gamma = \epsilon$. Under Algorithm 8.3, we have*[6] *for all $t > 0$:*

$$\frac{1}{t}\sum_{\tau=0}^{t-1} R(\tau) \geq \frac{1}{t}\sum_{\tau=0}^{t-1} R^{opt}(\tau) - \frac{2\bar{P}^2}{\epsilon t} - \frac{(\psi(\delta)+\sqrt{N_R}B^2)^2}{2}\epsilon - 2\psi(\delta)\bar{P}$$
$$tr(\mathbf{Q}(\tau)) \leq \bar{P}, \forall \tau \in \{0,1,\ldots,t-1\}$$

where $\psi(\delta)$ is the constant defined in Lemma 8.6 and B, δ, N_R, P and $\bar{P}$ are defined in Section 8.1.1. In particular, the sample path time average utility is within $O(\epsilon) + 2\psi(\delta)\bar{P}$ of the optimal average utility for the problem (8.6)-(8.8) *whenever $t \geq \frac{1}{\epsilon^2}$.*

Proof. The second inequality trivially follows from the fact that $\mathbf{Q}(t) \in \widetilde{\mathcal{Q}}, \forall t \in \{0,1,\ldots\}$. It remains to prove the first inequality. This proof extends the regret analysis of conventional online convex optimization [Zin03] by considering inexact gradient $\widetilde{\mathbf{D}}(t-1)$.

For all slots $\tau \in \{1,2,\ldots\}$, the transmit covariance update in Algorithm 8.3 satisfies:

$$\begin{aligned}
&\|\mathbf{Q}(\tau) - \mathbf{Q}^*\|_F^2 \\
=&\|\mathcal{P}_{\widetilde{\mathcal{Q}}}\big[\mathbf{Q}(\tau-1) + \gamma\widetilde{\mathbf{D}}(\tau-1)\big] - \mathbf{Q}^*\|_F^2 \\
\overset{(a)}{\leq}&\|\mathbf{Q}(\tau-1) + \gamma\widetilde{\mathbf{D}}(\tau-1) - \mathbf{Q}^*\|_F^2 \\
=&\|\mathbf{Q}(\tau-1) - \mathbf{Q}^*\|_F^2 + 2\gamma\mathrm{tr}\big(\widetilde{\mathbf{D}}^{\mathsf{H}}(\tau-1)(\mathbf{Q}(\tau-1)-\mathbf{Q}^*)\big) + \gamma^2\|\widetilde{\mathbf{D}}(\tau-1)\|_F^2 \\
=&\|\mathbf{Q}(\tau-1) - \mathbf{Q}^*\|_F^2 + 2\gamma\mathrm{tr}\big(\mathbf{D}^{\mathsf{H}}(\tau-1)(\mathbf{Q}(\tau-1)-\mathbf{Q}^*)\big) \\
&+ 2\gamma\mathrm{tr}\big((\widetilde{\mathbf{D}}(t-1) - \mathbf{D}(\tau-1))^{\mathsf{H}}(\mathbf{Q}(\tau-1)-\mathbf{Q}^*)\big) + \gamma^2\|\widetilde{\mathbf{D}}(\tau-1)\|_F^2,
\end{aligned}$$

where (a) follows from the non-expansive property of projections onto convex sets. Define $\Delta(t) = \|\mathbf{Q}(t+1)-\mathbf{Q}^*\|_F^2 - \|\mathbf{Q}(t)-\mathbf{Q}^*\|_F^2$. Rearranging terms in the last equation and dividing by factor 2γ implies

$$\begin{aligned}
&\mathrm{tr}\big(\mathbf{D}^{\mathsf{H}}(\tau-1)(\mathbf{Q}(\tau-1)-\mathbf{Q}^*)\big) \\
\geq&\frac{1}{2\gamma}\Delta(\tau-1) - \frac{\gamma}{2}\|\widetilde{\mathbf{D}}(\tau-1)\|_F^2 - \mathrm{tr}\big((\widetilde{\mathbf{D}}(\tau-1) - \mathbf{D}(\tau-1))^{\mathsf{H}}(\mathbf{Q}(\tau-1)-\mathbf{Q}^*)\big) \qquad (8.20)
\end{aligned}$$

[6] In our conference version [YN16a], the first inequality of this theorem is mistakenly given by $\frac{1}{t}\sum_{\tau=0}^{t-1} R(\tau) \geq \frac{1}{t}\sum_{\tau=0}^{t-1} R^{\mathrm{opt}}(\tau) - \frac{\bar{P}}{\epsilon t} - \frac{(\psi(\delta)+\sqrt{N_R B^2})^2}{2}\epsilon - 2\psi(\delta)\bar{P}$.

Define $f^{\tau-1}(\mathbf{Q}) = \log\det(\mathbf{I} + \mathbf{H}(\tau-1)\mathbf{Q}\mathbf{H}^{\mathsf{H}}(\tau-1))$. By Fact 8.3 in Section 8.7.1, $f^{\tau-1}(\cdot)$ is concave over $\widetilde{\mathcal{Q}}$ and $\mathbf{D}(t-1) = \nabla_{\mathbf{Q}} f^{\tau-1}(\mathbf{Q}(t-1))$. Note that $\mathbf{Q}^* \in \widetilde{\mathcal{Q}}$. By Fact 8.4 in Section 8.7.1, we have

$$f^{\tau-1}(\mathbf{Q}(\tau-1)) - f^{\tau-1}(\mathbf{Q}^*) \geq \mathrm{tr}(\mathbf{D}^{\mathsf{H}}(\tau-1)(\mathbf{Q}(\tau-1) - \mathbf{Q}^*)) \tag{8.21}$$

Note that $f^{\tau-1}(\mathbf{Q}(\tau-1)) = R(\tau-1)$ and $f^{\tau-1}(\mathbf{Q}^*) = R^{\mathrm{opt}}(\tau-1)$. Combining (8.20) and (8.21) yields

$$\begin{aligned}
&R(\tau-1) - R^{\mathrm{opt}}(\tau-1) \\
\geq &\frac{1}{2\gamma}\Delta(\tau-1) - \frac{\gamma}{2}\|\widetilde{\mathbf{D}}(\tau-1)\|_F^2 - \mathrm{tr}\big((\widetilde{\mathbf{D}}(\tau-1) - \mathbf{D}(\tau-1))^{\mathsf{H}}(\mathbf{Q}(\tau-1) - \mathbf{Q}^*)\big) \\
\overset{(a)}{\geq} &\frac{1}{2\gamma}\Delta(\tau-1) - \frac{\gamma}{2}\|\widetilde{\mathbf{D}}(\tau-1)\|_F^2 - \|\widetilde{\mathbf{D}}(\tau-1) - \mathbf{D}(\tau-1)\|_F\|\mathbf{Q}(\tau-1) - \mathbf{Q}^*\|_F \\
\overset{(b)}{\geq} &\frac{1}{2\gamma}\Delta(\tau-1) - \frac{\gamma}{2}(\psi(\delta) + \sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P}
\end{aligned}$$

where (a) follows from Fact 8.1 in Section 8.7.1 and (b) follows from Lemma 8.6 and the fact that $\|\mathbf{Q}(\tau-1) - \mathbf{Q}^*\|_F \leq \|\mathbf{Q}(\tau-1)\|_F + \|\mathbf{Q}^*\|_F \leq \mathrm{tr}(\mathbf{Q}(\tau-1)) + \mathrm{tr}(\mathbf{Q}^*) \leq 2\bar{P}$, which is implied by Fact 8.1, Fact 8.2 in Section 8.7.1 and the fact that $\mathbf{Q}(\tau-1), \mathbf{Q}^* \in \widetilde{\mathcal{Q}}$. Replacing $\tau-1$ with τ yields for all $\tau \in \{0, 1, \ldots\}$

$$R(\tau) - R^{\mathrm{opt}}(\tau) \geq \frac{1}{2\gamma}\Delta(\tau) - \frac{\gamma}{2}(\psi(\delta) + \sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P} \tag{8.22}$$

Fix $t > 0$. Summing over $\tau \in \{0, 1, \ldots, t-1\}$, dividing by factor t and simplifying telescope sum $\sum_{\tau=0}^{t-1}\Delta(\tau)$ gives

$$\begin{aligned}
&\frac{1}{t}\sum_{\tau=0}^{t-1} R(\tau) - \frac{1}{t}\sum_{\tau=0}^{t-1} R^{\mathrm{opt}}(\tau)) \\
\geq &\frac{1}{2\gamma t}(\|\mathbf{Q}(t) - \mathbf{Q}^*\|_F^2 - \|\mathbf{Q}(0) - \mathbf{Q}^*\|_F^2) - \frac{\gamma}{2}(\psi(\delta) + \sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P} \\
\overset{(a)}{\geq} &-\frac{2\bar{P}^2}{\gamma t} - \frac{\gamma}{2}(\psi(\delta) + \sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P}
\end{aligned}$$

where (a) follows from $\|\mathbf{Q}(0) - \mathbf{Q}^*\|_F \leq \|\mathbf{Q}(0)\|_F + \|\mathbf{Q}^*\|_F \leq \mathrm{tr}(\mathbf{Q}(0)) + \mathrm{tr}(\mathbf{Q}^*) \leq 2\bar{P}$ and $\|\mathbf{Q}(t) - \mathbf{Q}^*\|_F^2 \geq 0$. □

Theorem 8.2 proves a sample path guarantee on the utility. It shows that the convergence time to reach an $O(\epsilon)+2\psi(\delta)\bar{P}$ approximate solution is $1/\epsilon^2$. Note that if $\delta=0$, then equations (8.15) and (8.16) are recovered by Theorem 8.2. Theorem 8.2 also isolates the effect of missing CDIT and CSIT inaccuracy. The error term $O(\epsilon)$ is corresponding to the effect of missing CDIT and can be made arbitrarily small by choosing a small γ and running the algorithm for more than $\frac{1}{\epsilon^2}$ iterations. The observation is that the effect of missing CDIT vanishes as Algorithm 8.3 runs for a sufficiently long time and hence delayed but accurate CSIT is almost as good as CDIT. The other error term $2\psi(\delta)\bar{P}$ is corresponding to the effect of CSIT inaccuracy and does not vanish. The performance degradation due to channel inaccuracy scales linearly with respect to the channel error since $\psi(\delta)\in O(\delta)$. Intuitively, this is reasonable since any algorithm based on inaccurate CSIT is actually optimizing another different MIMO system.

8.3.3 Extensions

T-Slot Delayed and Inaccurate CSIT

Thus far, we have assumed that CSIT is always delayed by one slot. In fact, if CSIT is delayed by T slots, we can modify the update of transmit covariances in Algorithm 8.3 as $\mathbf{Q}(t)=\mathcal{P}_{\widetilde{\mathcal{Q}}}[\mathbf{Q}(t-T)+\gamma\widetilde{\mathbf{D}}(t-T)]$. A T-slot version of Theorem 8.2 can be similarly proven.

Algorithm 8.3 with Time Varying γ

Algorithm 8.3 can be extended to have time varying step size $\gamma(t)=\frac{1}{\sqrt{t}}$ at slot t. The next lemma shows that such an algorithm can approach an $\epsilon+2\psi(\delta)\bar{P}$ approximate solution with $O(1/\epsilon^2)$ iterations.

Lemma 8.7. *Fix $\epsilon>0$. If we modify Algorithm 8.3 by using $\gamma(t)=\frac{1}{\sqrt{t}}$ as the step size γ at each slot t, then for all $t>0$:*

$$\begin{aligned}\frac{1}{t}\sum_{\tau=0}^{t-1}R(\tau)\geq&\frac{1}{t}\sum_{\tau=0}^{t-1}R^{opt}(\tau)-\frac{2\bar{P}^2}{\sqrt{t}}-\frac{1}{\sqrt{t}}(\psi(\delta)+\sqrt{N_R}B^2)^2-2\psi(\delta)\bar{P},\\ \frac{1}{t}\sum_{\tau=0}^{t-1}tr(\mathbf{Q}(\tau))\leq&\bar{P},\end{aligned}$$

where B,δ,N_R,P and $\bar{P}$ are defined in Section 8.1.1

Proof. The second inequality again follows from the fact that $\mathbf{Q}(t) \in \widetilde{\mathcal{Q}}, \forall t \in \{0,1,\ldots\}$. It remains to prove the first inequality. With $\gamma(t) = \frac{1}{\sqrt{t}}$, equation (8.22) in the proof of Theorem 8.2 becomes $R(\tau) - R^{\text{opt}}(\tau) \geq \frac{1}{2\gamma(\tau+1)}\Delta(\tau) - \frac{\gamma(\tau+1)}{2}(\psi(\delta)+\sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P}$ for all $\tau \in \{0,1,\ldots\}$. Fix $t > 0$. Summing over $\tau \in \{0,1,\ldots,t-1\}$ and dividing by factor t yields that for all $t > 0$:

$$\begin{aligned}
&\frac{1}{t}\sum_{\tau=0}^{t-1} R(\tau) - \frac{1}{t}\sum_{\tau=0}^{t-1} R^{\text{opt}}(\tau) \\
\geq &\frac{1}{2t}\sum_{\tau=0}^{t-1}\sqrt{\tau+1}\Delta(\tau) - \frac{1}{t}\left(\sum_{\tau=0}^{t-1}\frac{1}{2\sqrt{\tau+1}}\right)(\psi(\delta)+\sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P} \\
\overset{(a)}{\geq} &-\frac{2\bar{P}^2}{\sqrt{t}} - \frac{1}{\sqrt{t}}(\psi(\delta)+\sqrt{N_R}B^2)^2 - 2\psi(\delta)\bar{P}
\end{aligned}$$

where (a) follows because $\sum_{\tau=0}^{t-1}\sqrt{\tau+1}\Delta(\tau) = \sqrt{t}\|\mathbf{Q}(t)-\mathbf{Q}^*\|_F^2 - \|\mathbf{Q}(0)-\mathbf{Q}^*\|_F^2 + \sum_{\tau=0}^{t-2}(\sqrt{\tau+1}-\sqrt{\tau+2})\|\mathbf{Q}(\tau+1)-\mathbf{Q}^*\|_F^2 \geq -\|\mathbf{Q}(0)-\mathbf{Q}^*\|_F^2 + 4\bar{P}^2\sum_{\tau=0}^{t-2}(\sqrt{\tau+1}-\sqrt{\tau+2}) \geq -4\bar{P}^2\sqrt{t}$ and $\sum_{\tau=0}^{t-1}\frac{1}{2\sqrt{\tau+1}} \leq \sqrt{t}$. □

An advantage of time varying step sizes is the performance automatically gets improved as the algorithm runs and there is no need to restart the algorithm with a different constant step size if a better performance is demanded.

8.4 Rate Adaptation

To achieve the capacity characterized by either the problem (8.2)-(8.4) or the problem (8.6)-(8.8), we also need to consider the rate allocation associated with the transmit covariance, namely, we need to decide how much data is delivered at each slot. If the accurate instantaneous CSIT is available, the transmitter can simply deliver $\log\det(\mathbf{I}+\mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t))$ amount of data at slot t once $\mathbf{Q}(t)$ is decided. However, in the cases when instantaneous CSIT is inaccurate or only delayed CSIT is available, the transmitter does not know the associated instantaneous channel capacity without knowing $\mathbf{H}(t)$. These cases belong to the representative communication scenarios where channels are unknown to the transmitter and *rateless codes* are usually used as a solution. To send N bits of source data, the rateless code keeps sending encoded information bits without knowing instantaneous channel capacity such that the receiver can decode all N bits as long as the accumulated channel capacity for sufficiently many slots is larger than N. Many practical rateless codes for scalar or MIMO fading channels have been designed in [ETW12, FLEP10, LTCS16].

This section provides an information theoretical rate adaptation policy based on rateless codes that can be combined with the dynamic power allocation algorithms developed in this chapter.

The rate adaptation scheme is as follows: Let N be a large number. Encode N bits of source data with a capacity achieving code for a channel with capacity no less than N bits per slot. At slot 0, deliver the above encoded data with transmit covariance $\mathbf{Q}(0)$ given by Algorithm 8.1 or Algorithm 8.3. The receiver knows channel $\mathbf{H}(0)$, calculates the channel capacity $R(0) = \log\det(\mathbf{I} + \mathbf{H}(0)\mathbf{Q}(0)\mathbf{H}^{\mathsf{H}}(0))$; and reports back the scalar $R(0)$ to the transmitter. At slot 1, the transmitter removes the first $R(0)$ bits from the N bits of source data, encodes the remaining $N - R(0)$ bits with a capacity achieving code for a channel with capacity no less than $N - R(0)$ bits per slot; and delivers the encoded data with transmit covariance $\mathbf{Q}(1)$ given by Algorithm 8.1 or Algorithm 8.3. The receiver knows channel $\mathbf{H}(1)$, calculates the channel capacity $R(1) = \log\det(\mathbf{I}+\mathbf{H}(1)\mathbf{Q}(1)\mathbf{H}^{\mathsf{H}}(1))$; and reports back the scalar $R(1)$ to the transmitter. Repeat the above process until slot $T-1$ such that $\sum_{t=0}^{T-1} R(t) > N$.

For the decoding, the receiver can decode all the N bits in a reverse order using the idea of successive decoding [TV05]. At slot $T-1$, since $N - \sum_{t=1}^{T-2} R(t) < R(T-1)$, that is, $N - \sum_{t=0}^{T-2} R(t) < R(T-1)$ bits of source data are delivered over a channel with capacity $R(T-1)$ bits per slot, the receiver can decode all delivered data ($N - \sum_{t=0}^{T-2} R(t)$ bits) with zero error. Note that $N - \sum_{t=0}^{T-3} R(t) = R(T-2) + N - \sum_{t=0}^{T-2} R(t)$ bits are delivered at slot $T-2$ over a channel with capacity $R(T-2)$ bits per slot. The receiver subtracts the $N - \sum_{t=0}^{T-2} R(t)$ bits that are already decoded such that only $R(T-2)$ bits remain to be decoded. Thus, the $R(T-2)$ bits can be successfully decoded. Repeat this process until all N bits are decoded.

Using the above rate adaptation and decoding strategy, N bits are delivered and decoded within $T-1$ slots during which the sum capacity is $\sum_{t=0}^{T-1} R(t)$ bits. When N is large enough, the rate loss $\sum_{t=0}^{T-1} R(t) - N$ is negligible. This rate adaptation scheme does not require $\mathbf{H}(t)$ and only requires to report back the scalar $R(t-1)$ to the transmitter at each slot t.

8.5 Simulations

8.5.1 A Simple MIMO System with Two Channel Realizations

Consider a 2×2 MIMO system with two equally likely channel realizations:

$$\mathbf{H}_1 = \begin{bmatrix} 1.3131e^{j1.9590\pi} & 2.3880e^{j0.7104\pi} \\ 2.5567e^{j1.5259\pi} & 2.8380e^{j0.3845\pi} \end{bmatrix},$$

$$\mathbf{H}_2 = \begin{bmatrix} 1.4781e^{j0.9674\pi} & 1.5291e^{j0.1396\pi} \\ 0.0601e^{j0.9849\pi} & 0.1842e^{j1.9126\pi} \end{bmatrix}.$$

This simple scenario is considered as a test case because, when there are only two possible channels with known channel probabilities, it is easy to find an optimal baseline algorithm by solving the problem (8.2)-(8.4) or the problem (8.6)-(8.8) directly. The goal is to show that the proposed algorithms (which do not have channel distribution information) come close to this baseline. The proposed algorithms can be implemented just as easily in cases when there are an infinite number of possible channel state matrices, rather than just two. However, in that case it is difficult to find an optimal baseline algorithm since the problem (8.2)-(8.4) or the problem (8.6)-(8.8) are difficult to solve.[7]

The power constraints are $\bar{P} = 2$ and $P = 3$. If CSIT has error, $\mathbf{H}_1$ and $\mathbf{H}_2$ are observed as $\widetilde{\mathbf{H}}_1$ and $\widetilde{\mathbf{H}}_2$, respectively. Consider two CSIT error cases. CSIT Error Case 1: $\widetilde{\mathbf{H}}_1 = \begin{bmatrix} 1.3131e^{j2\pi} & 2.3880e^{j0.75\pi} \\ 2.5567e^{j1.5\pi} & 2.8380e^{j0.5\pi} \end{bmatrix}$ and $\widetilde{\mathbf{H}}_2 = \begin{bmatrix} 1.4781e^{j1\pi} & 1.5291e^{j0.25\pi} \\ 0.0601e^{j1\pi} & 0.1842e^{j2\pi} \end{bmatrix}$, where the magnitudes are accurate but the phases are rounded to the nearest $\pi/4$ phase; CSIT Error Case 2: $\widetilde{\mathbf{H}}_1 = \begin{bmatrix} 1.3e^{j2\pi} & 2.4e^{j0.5\pi} \\ 2.6e^{j1.5\pi} & 2.8e^{j0.5\pi} \end{bmatrix}$ and $\widetilde{\mathbf{H}}_2 = \begin{bmatrix} 1.5e^{j1\pi} & 1.5e^{j0\pi} \\ 0 & 0.2e^{j2\pi} \end{bmatrix}$, where the magnitudes are rounded to the first digit after the decimal point and the phases are rounded to the nearest $\pi/2$ phase.

In the instantaneous CSIT case, consider Baseline 1 where the optimal solution $\mathbf{Q}^*(\mathbf{H})$ to the problem (8.2)-(8.4) is calculated by assuming the knowledge that $\mathbf{H}_1$ and $\mathbf{H}_2$ appear with equal probabilities and $\mathbf{Q}(t) = \mathbf{Q}^*(\mathbf{H}(t))$ is used at each slot t. Figure 8.1 compares the performance

[7] As discussed in Section 8.2.3, this is known as the curse of dimensionality of empirical methods for stochastic optimization due to the large sample size.

of Algorithm 8.1 (with $V = 100$) under various CSIT accuracy conditions and Baseline 1. It can be seen that Algorithm 8.1 has a performance close to that attained by the optimal solution to the problem (8.2)-(8.4) requiring channel distribution information. (Note that a larger V gives a even closer performance with a longer convergence time.) It can also be observed that the performance of Algorithm 8.1 becomes worse as CSIT error gets larger.

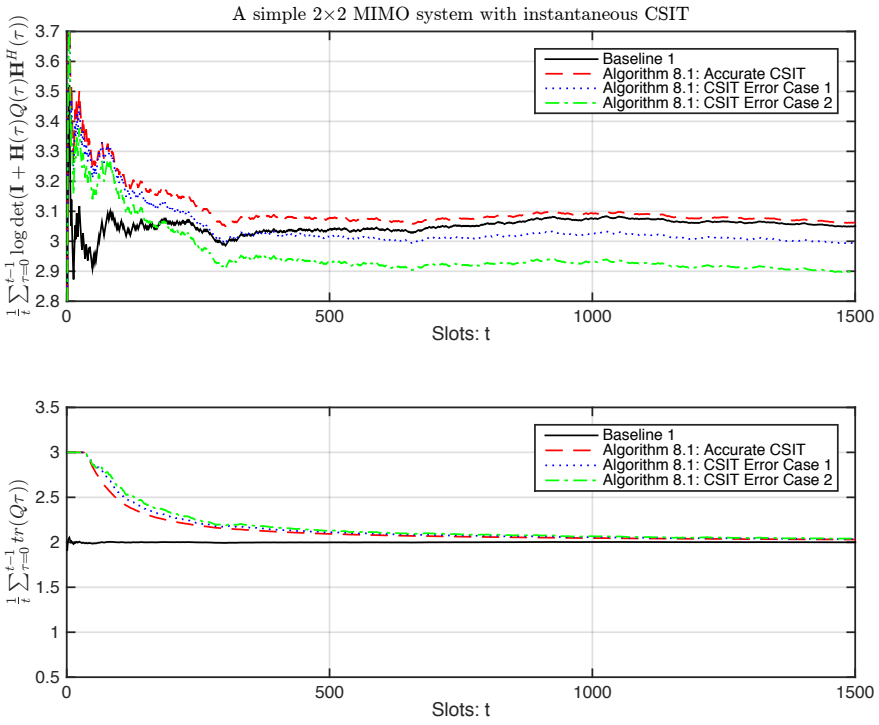

Figure 8.1: A simple MIMO system with instantaneous CSIT.

In the delayed CSIT case, consider Baseline 2 where the optimal solution $\mathbf{Q}^*$ to the problem (8.6)-(8.8) is calculated by assuming the knowledge that $\mathbf{H}_1$ and $\mathbf{H}_2$ appear with equal probabilities; and $\mathbf{Q}(t) = \mathbf{Q}^*$ is used at each slot t. Figure 8.2 compares the performance of Algorithm 8.3 (with $\gamma = 0.01$) under various CSIT accuracy conditions and Baseline 2. Note that the average power is not drawn since the average power constraint is satisfied for all t in all schemes. It can be seen that Algorithm 8.3 has a performance close to that attained by the optimal solution to the problem (8.6)-(8.8) requiring channel distribution information. (Note that a smaller γ gives a even closer performance with a longer convergence time.) It can also be observed that the performance of Algorithm 8.3 becomes worse as CSIT error gets larger.

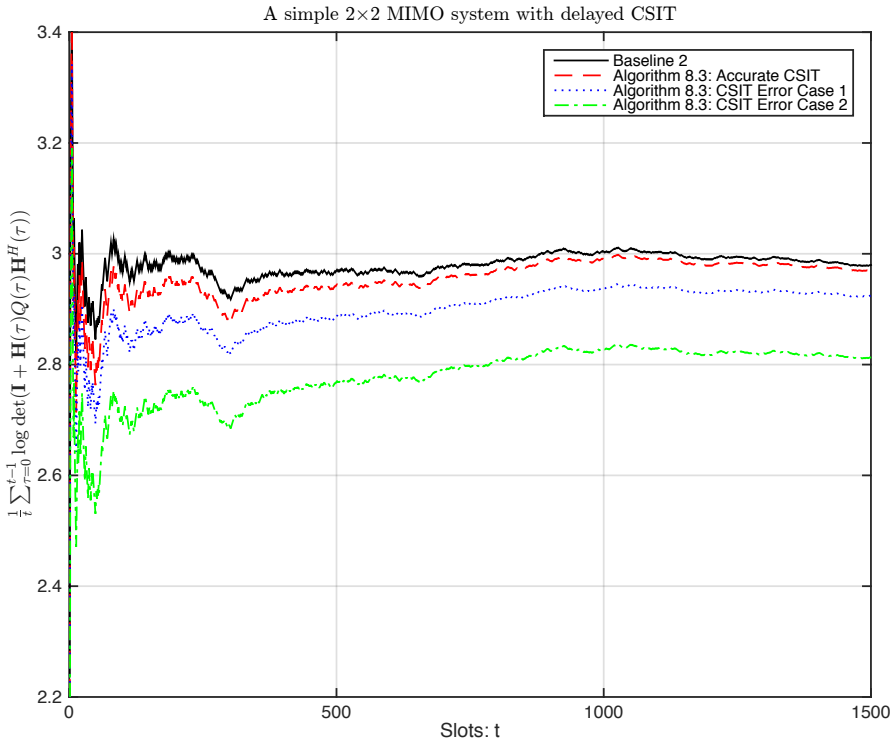

Figure 8.2: A simple MIMO system with delayed CSIT.

8.5.2 A MIMO System with Continuous Channel Realizations

This section considers a 2×2 MIMO system with continuous channel realizations. Each entry in $\mathbf{H}(t)$ is equal to uv where u is a complex number whose real part and complex part are standard normal and v is uniform over $[0, 0.5]$. In this case, even if the channel distribution information is perfectly known, the problem (8.2)-(8.4) and the problem (8.6)-(8.8) are infinite dimensional problems and are extremely hard to solve. In practice, to solve the stochastic optimization, people usually approximate the continuous distribution by a discrete distribution with a reasonable number of realizations and solve the approximate optimization that is a large scale deterministic optimization problem. (Baselines 3 and 4 considered below are essentially using this idea.)

In the instantaneous CSIT case, consider Baseline 3 where we spend 100 slots to obtain an empirical channel distribution by observing 100 accurate channel realizations [8]; obtain the

[8] By doing so, 100 slots are wasted without sending any data. The 100 slots are not counted in the simulation. If they are counted, Algorithm 8.1's performance advantage over Baseline 3 is even bigger. The delayed CSIT case is similar.

optimal solution $\mathbf{Q}^*(\mathbf{H}), \mathbf{H} \in \mathcal{H}$ to the problem (8.2)-(8.4) using the empirical distribution; choose $\mathbf{Q}^*(\mathbf{H})$ where $\mathbf{H} = \text{argmin}_{\mathbf{H}\in\mathcal{H}} \|\mathbf{H} - \mathbf{H}(t)\|_F$ at each slot t. Figure 8.3 compares the performance of Algorithm 8.1 (with $V = 100$) and Baseline 3; and shows that Algorithm 8.1 has a better performance than Baseline 3.

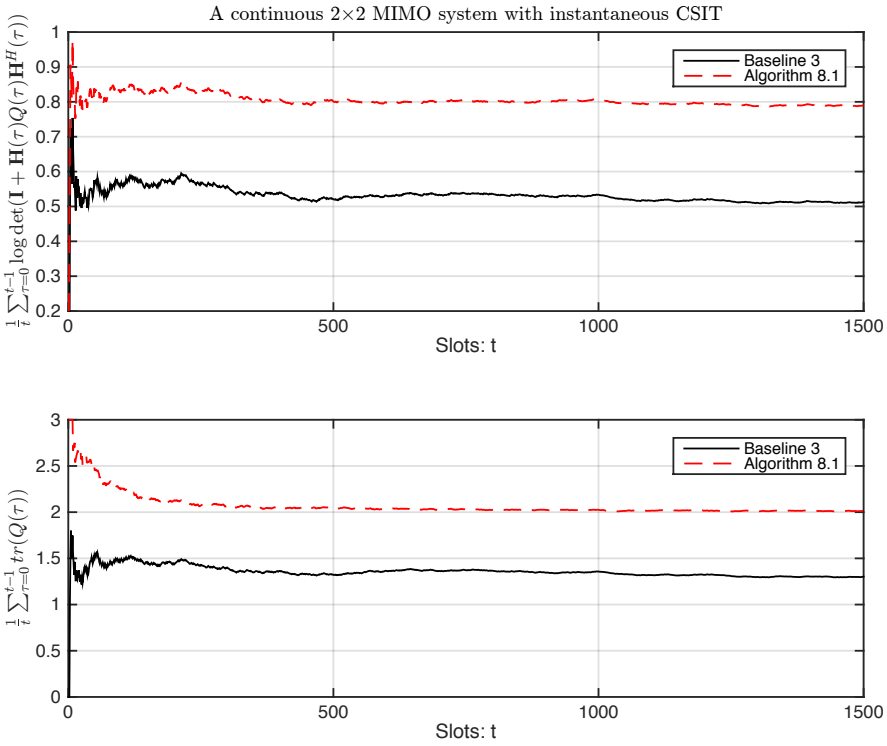

Figure 8.3: A continuous MIMO system with instantaneous CSIT.

In the delayed CSIT case, consider Baseline 4 where we spend 100 slots to obtain an empirical channel distribution by observing 100 accurate channel realizations; obtain the optimal solution $\mathbf{Q}^*$ to the problem (8.6)-(8.8) using the empirical distribution; choose $\mathbf{Q}^*$ at each slot t. Figure 8.4 compares the performance of Algorithm 8.3 (with $\gamma = 0.01$) and Baseline 4; and shows that Algorithm 8.3 has a better performance than Baseline 4.

8.6 Chapter Summary

This chapter considers dynamic transmit covariance design in point-to-point MIMO fading systems without CDIT. Two different dynamic policies are proposed to deal with the cases of instantaneous CSIT and delayed CSIT, respectively. In both cases, the proposed dynamic policies

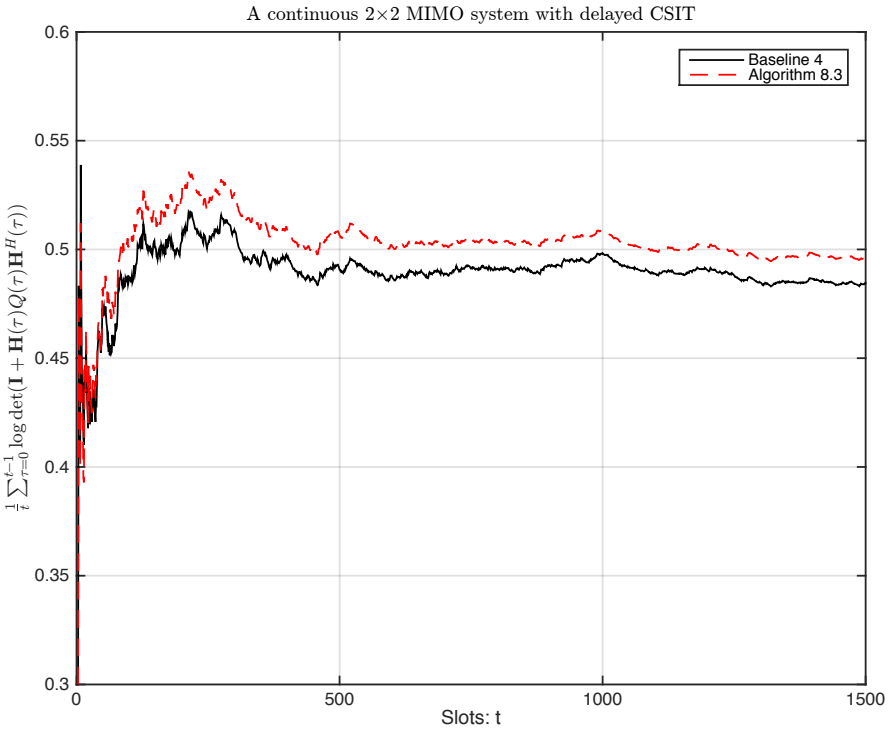

Figure 8.4: A continuous MIMO system with delayed CSIT.

can achieve $O(\delta)$ sub-optimality, where δ is the inaccuracy measure of CSIT.

8.7 Supplement to this Chapter

8.7.1 Linear Algebra and Matrix Derivatives

Fact 8.1 ([HJ85]). *For any* $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{m\times n}$ *and* $\mathbf{C} \in \mathbb{C}^{n\times k}$ *we have:*

1. $\|\mathbf{A}\|_F = \|\mathbf{A}^{\mathsf{H}}\|_F = \|\mathbf{A}^{\mathsf{T}}\|_F = \|-\mathbf{A}\|_F$.
2. $\|\mathbf{A}+\mathbf{B}\|_F \le \|\mathbf{A}\|_F + \|\mathbf{B}\|_F$.
3. $\|\mathbf{AC}\|_F \le \|\mathbf{A}\|_F\|\mathbf{C}\|_F$.
4. $|tr(\mathbf{A}^{\mathsf{H}}\mathbf{B})| \le \|\mathbf{A}\|_F\|\mathbf{B}\|_F$.

Fact 8.2 ([HJ85]). *For any* $\mathbf{A} \in \mathbb{S}_+^n$ *we have* $\|\mathbf{A}\|_F \le tr(\mathbf{A})$.

Fact 8.3 ([FHM07]). *The function* $f : \mathbb{S}_+^n \to \mathbb{R}$ *defined by* $f(\mathbf{Q}) = \log\det(\mathbf{I}+\mathbf{HQH}^{\mathsf{H}})$ *is concave and its gradient is given by* $\nabla_{\mathbf{Q}} f(\mathbf{Q}) = \mathbf{H}^{\mathsf{H}}(\mathbf{I}+\mathbf{HQH}^{\mathsf{H}})^{-1}\mathbf{H}, \forall \mathbf{Q} \in \mathbb{S}_+^n$.

The above fact is developed in [FHM07]. A general theory on developing derivatives for functions with complex matrix variables is available in [Hjø11]. The next fact is the complex matrix version of the first order condition for concave functions of real number variables, i.e., $f(y) \leq f(x) + f'(x)(y-x), \forall x, y \in \text{dom} f$ if f is concave. We also provide a brief proof for this fact.

Fact 8.4. *Let function $f(\mathbf{Q}) : \mathbb{S}_+^n \to \mathbb{R}$ be a concave function and have gradient $\nabla_{\mathbf{Q}} f(\mathbf{Q}) \in \mathbb{S}^n$ at point $\mathbf{Q}$. Then, $f(\widehat{\mathbf{Q}}) \leq f(\mathbf{Q}) + tr\big([\nabla_{\mathbf{Q}} f(\mathbf{Q})]^{\mathsf{H}}(\widehat{\mathbf{Q}} - \mathbf{Q})\big), \forall \widehat{\mathbf{Q}} \in \mathbb{S}_+^n$.*

Proof. Recall that a function is concave if and only if it is concave when restricted to any line that intersects its domain (see page 67 in [BV04]). For any $\mathbf{Q}, \widehat{\mathbf{Q}} \in \mathbb{S}_+^n$, define $g(t) = f(\mathbf{Q} + t(\widehat{\mathbf{Q}} - \mathbf{Q}))$. Thus, $g(t)$ is concave over $[0,1]$; $g(0) = f(\mathbf{Q})$; and $g(1) = f(\widehat{\mathbf{Q}})$. Note that $g'(t) = \text{tr}([\nabla_{\mathbf{Q}} f(\mathbf{Q} + t(\widehat{\mathbf{Q}} - \mathbf{Q}))]^{\mathsf{H}}(\widehat{\mathbf{Q}} - \mathbf{Q}))$ by the chain rule of derivatives when the inner product in complex matrix space $\mathbb{C}^{n \times n}$ is defined as $\langle \mathbf{A}, \mathbf{B} \rangle = \text{tr}(\mathbf{A}^{\mathsf{H}}\mathbf{B}), \forall \mathbf{A}, \mathbf{B} \in \mathbb{C}^{n \times n}$. By the first-order condition of concave function $g(t)$, we have $g(1) \leq g(0) + g'(0)(1-0)$. Note that $g'(0) = \text{tr}([\nabla_{\mathbf{Q}} f(\mathbf{Q})]^{\mathsf{H}}(\widehat{\mathbf{Q}} - \mathbf{Q}))$. Thus, we have $f(\widehat{\mathbf{Q}}) \leq f(\mathbf{Q}) + \text{tr}\big([\nabla_{\mathbf{Q}} f(\mathbf{Q})]^{\mathsf{H}}(\widehat{\mathbf{Q}} - \mathbf{Q})\big)$. □

8.7.2 Proof of Lemma 8.2

The proof method is an extension of Section 3.2 in [Tel99], which gives the structure of the optimal transmit covariance in deterministic MIMO channels.

Note that $\log\det(\mathbf{I} + \mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}) \overset{(a)}{=} \log\det(\mathbf{I} + \mathbf{Q}\mathbf{H}^{\mathsf{H}}\mathbf{H}) \overset{(b)}{=} \log\det(\mathbf{I} + \mathbf{Q}\mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}) \overset{(c)}{=} \log\det(\mathbf{I} + \mathbf{\Sigma}^{1/2}\mathbf{U}\mathbf{Q}\mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}^{1/2})$, where (a) and (c) follows from the elementary identity $\det(\mathbf{I} + \mathbf{A}\mathbf{B}) = \det(\mathbf{I} + \mathbf{B}\mathbf{A}), \forall \mathbf{A} \mathbb{C}^{m \times n}$ and $\mathbf{B} \in \mathbb{C}^{n \times m}$; and (b) follows from the fact that $\mathbf{H}^{\mathsf{H}}\mathbf{H} = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$. Define $\widetilde{\mathbf{Q}} = \mathbf{U}\mathbf{Q}\mathbf{U}^{\mathsf{H}}$, which is semidefinite positive if and only if $\mathbf{Q}$ is. Note that $\text{tr}(\widetilde{\mathbf{Q}}) = \text{tr}(\mathbf{U}\mathbf{Q}\mathbf{U}^{\mathsf{H}}) = \text{tr}(\mathbf{Q})$ by the fact that $\text{tr}(\mathbf{A}\mathbf{B}) = \text{tr}(\mathbf{B}\mathbf{A}), \forall \mathbf{A} \in \mathbb{C}^{m \times n}, \mathbf{B} \in \mathbb{C}^{n \times m}$. Thus, the problem (8.12)-(8.14) is equivalent to

$$\max_{\widetilde{\mathbf{Q}}} \quad \log\det(\mathbf{I} + \mathbf{\Sigma}^{1/2}\widetilde{\mathbf{Q}}\mathbf{\Sigma}^{1/2}) - \frac{Z}{V}\text{tr}(\widetilde{\mathbf{Q}}) \tag{8.23}$$

$$\text{s.t.} \quad \text{tr}(\widetilde{\mathbf{Q}}) \leq P \tag{8.24}$$

$$\widetilde{\mathbf{Q}} \in \mathbb{S}_+^{N_T} \tag{8.25}$$

Fact 8.5 (Hadamard's Inequality, Theorem 7.8.1 in [HJ85])**.** *For all $\mathbf{A} \in \mathbb{S}_+^n$, $\det(\mathbf{A}) \leq \prod_{i=1}^n A_{ii}$*

with equality if $\mathbf{A}$ *is diagonal.*

The next claim can be proven using Hadamard's inequality.

Claim 8.1. *The problem* (8.23)-(8.25) *has a diagonal optimal solution.*

Proof. Suppose the problem (8.23)-(8.25) has a non-diagonal optimal solution given by matrix $\widetilde{\mathbf{Q}}$. Consider a diagonal matrix $\widehat{\mathbf{Q}}$ whose entries are identical to the diagonal entries of $\widetilde{\mathbf{Q}}$. Note that $\mathrm{tr}(\widehat{\mathbf{Q}}) = \mathrm{tr}(\widetilde{\mathbf{Q}})$. To show $\widehat{\mathbf{Q}}$ is a solution no worse than $\widetilde{\mathbf{Q}}$, it suffices to show that $\log\det(\mathbf{I} + \mathbf{\Sigma}^{1/2}\widehat{\mathbf{Q}}\mathbf{\Sigma}^{1/2}) \geq \log\det(\mathbf{I} + \mathbf{\Sigma}^{1/2}\widetilde{\mathbf{Q}}\mathbf{\Sigma}^{1/2})$. This is true becase $\det(\mathbf{I} + \mathbf{\Sigma}^{1/2}\widehat{\mathbf{Q}}\mathbf{\Sigma}^{1/2}) = \prod_{i=1}^{N_T}(1+\widehat{Q}_{ii}\sigma_i) = \prod_{i=1}^{N_T}(1+\widetilde{Q}_{ii}\sigma_i) \geq \det(\mathbf{I}+\mathbf{\Sigma}^{1/2}\widetilde{\mathbf{Q}}\mathbf{\Sigma}^{1/2})$, where the last inequality follows from Hadamard's inequality. Thus, $\widehat{\mathbf{Q}}$ is a solution no worse than $\widetilde{\mathbf{Q}}$ and hence optimal. □

By Claim 8.1, we can consider $\widetilde{\mathbf{Q}} = \Theta = \mathrm{diag}(\theta_1, \theta_2, \ldots, \theta_{N_T})$ and the problem (8.23)-(8.25) is equivalent to

$$\max \quad \sum_{i=1}^{N_T} \log(1 + \theta_i\sigma_i) - \frac{Z}{V}\sum_{i=1}^{N_T}\theta_i \tag{8.26}$$

$$\text{s.t.} \quad \sum_{i=1}^{N_T}\theta_i \leq P \tag{8.27}$$

$$\theta_i \geq 0, \forall i \in \{1, 2, \ldots, N_T\} \tag{8.28}$$

Note that the problem (8.26)-(8.28) satisfies Slater's condition. So the optimal solution to the problem (8.26)-(8.28) is characterized by KKT conditions [BV04]. The remaining part is similar to the derivation of the water-filling solution of power allocation in parallel channels, e.g., the proof of Example 5.2 in [BV04]. Introducing Lagrange multipliers $\mu \in \mathbb{R}^+$ for inequality constraint $\sum_{i=1}^{N_T}\theta_i \leq P$ and $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_{N_T}]^\mathsf{T} \in \mathbb{R}^+$ for inequality constraints $\theta_i \geq 0, i \in \{1, 2, \ldots, N_T\}$. Let $\boldsymbol{\theta}^* = [\theta_1^*, \ldots, \theta_{N_T}^*]^\mathsf{T}$ and $(\mu^*, \boldsymbol{\nu}^*)$ be any primal and dual optimal points with zero duality gap. By the KKT conditions, we have $-\frac{\sigma_i}{1+\theta_i^*\sigma_i} + \frac{Z}{V} + \mu^* - \nu_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}; \sum_{i=1}^{N_T}\theta_i^* \leq P; \mu^* \geq 0; \mu^*\big(\sum_{i=1}^{N_T}\theta_i^* - P\big) = 0; \theta_i^* \geq 0, \forall i \in \{1, 2, \ldots, N_T\}; \nu_i^* \geq 0, \forall i \in \{1, 2, \ldots, N_T\}; \nu_i^*\theta_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}$.

Eliminating $\nu_i^*, \forall i \in \{1, 2, \ldots, N_T\}$ in all equations yields $\mu^* + \frac{Z}{V} \geq \frac{\sigma_i}{1+\theta_i^*\sigma_i}, \forall i \in \{1, 2, \ldots, N_T\}$; $\sum_{i=1}^{N_T}\theta_i^* \leq P$; $\mu^* \geq 0$; $\mu^*\big(\sum_{i=1}^{N_T}\theta_i^* - P\big) = 0$; $\theta_i^* \geq 0, \forall i \in \{1, 2, \ldots, N_T\}$; $(\mu^* + \frac{Z}{V} - \frac{\sigma_i}{1+\theta_i^*\sigma_i})\theta_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}$.

For all $i \in \{1, 2, \ldots, N_T\}$, we consider $\mu^* + \frac{Z}{V} < \sigma_i$ and $\mu^* + \frac{Z}{V} \geq \sigma_i$ separately:

1. If $\mu^* + \frac{Z}{V} < \sigma_i$, then $\mu^* + \frac{Z}{V} \geq \frac{\sigma_i}{1+\theta_i^*\sigma_i}$ holds only when $\theta_i^* > 0$, which by $(\mu^* + \frac{Z}{V} - \frac{\sigma_i}{1+\theta_i^*\sigma_i})\theta_i^*$ implies that $\mu^* + \frac{Z}{V} - \frac{\sigma_i}{1+\theta_i^*\sigma_i} = 0$, i.e., $\theta_i^* = \frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_i}$.

2. If $\mu^* + \frac{Z}{V} \geq \sigma_i$, then $\theta_i^* > 0$ is impossible, because $\theta_i^* > 0$ implies that $\mu^* + \frac{Z}{V} - \frac{\sigma_i}{1+\theta_i^*\sigma_i} > 0$, which together with $\theta_i^* > 0$ contradict the slackness condition $(\mu^* + \frac{Z}{V} - \frac{\sigma_i}{1+\theta_i^*\sigma_i})\theta_i^* = 0$. Thus, if $\mu^* + \frac{Z}{V} \geq \sigma_i$, we must have $\theta_i^* = 0$.

Summarizing both cases, we have $\theta_i^* = \max\{0, \frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_i}\}, \forall i \in \{1, 2, \ldots, N_T\}$, where μ^* is chosen such that $\sum_{i=1}^n \theta_i^* \leq P$, $\mu^* \geq 0$ and $\mu^*\big(\sum_{i=1}^{N_T} \theta_i^* - P\big) = 0$.

To find such μ^*, we first check if $\mu^* = 0$. If $\mu^* = 0$ is true, the slackness condition $\mu^*\big(\sum_{i=1}^{N_T} \theta_i^* - P\big) = 0$ holds and we need to further ensure $\sum_{i=1}^{N_T} \theta_i^* = \sum_{i=1}^{N_T} \max\{0, \frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_i}\} \leq P$. Thus $\mu^* = 0$ if and only if $\sum_{i=1}^{N_T} \max\{0, \frac{1}{Z/V} - \frac{1}{\sigma_i}\} \leq P$. Thus, Algorithm 8.2 checks if $\sum_{i=1}^{N_T} \max\{0, \frac{1}{Z/V} - \frac{1}{\sigma_i}\} \leq P$ holds at the first step. If this is true, then we conclude $\mu^* = 0$ and we are done!

Otherwise, we know $\mu^* > 0$. By the slackness condition $\mu^*\big(\sum_{i=1}^{N_T} \theta_i^* - P\big) = 0$, we must have $\sum_{i=1}^{N_T} \theta_i^* = \sum_{i=1}^{N_T} \max\{0, \frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_i}\} = P$. To find $\mu^* > 0$ such that $\sum_{i=1}^{N_T} \max\{0, \frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_i}\} = P$, we could apply a bisection search by noting that all θ_i^* are decreasing with respect to μ^*.

Another algorithm of finding μ^* is inspired by the observation that if $\sigma_j \geq \sigma_k, \forall j, k \in \{1, 2, \ldots, N_T\}$, then $\theta_j^* \geq \theta_k^*$. Thus, we first sort all σ_i in a decreasing order, say π is the permutation such that $\sigma_{\pi(1)} \geq \sigma_{\pi(2)} \geq \cdots \geq \sigma_{\pi(N_T)}$; and then sequentially check if $i \in \{1, 2, \ldots, N_T\}$ is the index such that $\sigma_{\pi(i)} - \mu^* \geq 0$ and $\sigma_{\pi(i+1)} - \mu^* \leq 0$. To check this, we first assume i is indeed such an index and solve the equation $\sum_{j=1}^{i} \big(\frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_{\pi(j)}}\big) = P$ to obtain μ^*; (Note that in Algorithm 8.2, to avoid recalculating the partial sum $\sum_{j=1}^{i} \frac{1}{\sigma_{\pi(j)}}$ for each i, we introduce the parameter $S_i = \sum_{j=1}^{i} \frac{1}{\sigma_{\pi(j)}}$ and update S_i incrementally. By doing this, the complexity of each iteration in the loop is only $O(1)$.) then verify the assumption by checking if $\frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_{\pi(i)}} \geq 0$ and $\frac{1}{\mu^*+Z/V} - \frac{1}{\sigma_{\pi(i+1)}} \leq 0$. This algorithm is described in Algorithm 8.2.

8.7.3 Proof of Lemma 8.3

Fact 8.6. *For all $\mathbf{X} \in \mathbb{S}_+^n$, we have $\|(\mathbf{I} + \mathbf{X})^{-1}\|_F \leq \sqrt{n}$.*

Proof. Since $\mathbf{X} \in \mathbb{S}_+^n$, matrix $\mathbf{X}$ has SVD $\mathbf{X} = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$, where $\mathbf{U}$ is unitary and $\mathbf{\Sigma}$ is diagonal with non-negative entries $\sigma_1, \ldots, \sigma_n$. Then $\mathbf{Y} = (\mathbf{I} + \mathbf{X})^{-1} = \mathbf{U}^{\mathsf{H}}\mathrm{diag}(\frac{1}{1+\sigma_1}, \ldots, \frac{1}{1+\sigma_n})\mathbf{U}$ is Hermitian.

Thus, $\|(\mathbf{I}+\mathbf{X})^{-1}\|_F = \sqrt{\mathrm{tr}(\mathbf{Y}^2)} = \sqrt{\sum_{i=1}^n (\frac{1}{1+\sigma_i})^2} \le \sqrt{n}$. □

Fact 8.7. *For any $\mathbf{H}, \widetilde{\mathbf{H}} \in \mathbb{C}^{N_R \times N_T}$ with $\|\mathbf{H}\|_F \le B$ and $\|\mathbf{H}-\widetilde{\mathbf{H}}\|_F \le \delta$, we have $\|\mathbf{H}^{\mathsf{H}}\mathbf{H} - \widetilde{\mathbf{H}}^{\mathsf{H}}\widetilde{\mathbf{H}}\|_F \le (2B+\delta)\delta$.*

Proof.

$$\begin{aligned}
&\|\mathbf{H}^{\mathsf{H}}\mathbf{H} - \widetilde{\mathbf{H}}^{\mathsf{H}}\widetilde{\mathbf{H}}\|_F \\
\overset{(a)}{\le} &\|\mathbf{H}^{\mathsf{H}}\mathbf{H} - \mathbf{H}^{\mathsf{H}}\widetilde{\mathbf{H}}\|_F + \|\mathbf{H}^{\mathsf{H}}\widetilde{\mathbf{H}} - \widetilde{\mathbf{H}}^{\mathsf{H}}\widetilde{\mathbf{H}}\|_F \\
\overset{(b)}{\le} &\|\mathbf{H}^{\mathsf{H}}\|_F\|\mathbf{H}-\widetilde{\mathbf{H}}\|_F + \|\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}^{\mathsf{H}}\|_F\|\widetilde{\mathbf{H}}\|_F \\
\overset{(c)}{\le} &\|\mathbf{H}^{\mathsf{H}}\|_F\|\mathbf{H}-\widetilde{\mathbf{H}}\|_F + \|\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}^{\mathsf{H}}\|_F\big(\|\widetilde{\mathbf{H}}-\mathbf{H}\|_F + \|\mathbf{H}\|_F\big) \\
\le &2B\delta + \delta^2
\end{aligned}$$

where (a) and (c) follow from part 2 of Fact 8.1; and (b) follows from part 3 of Fact 8.1. □

Fix $Z(t)$ and V. Define $\phi(\mathbf{Q},\mathbf{H}) = V\log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}) - Z(t)\mathrm{tr}(\mathbf{Q})$ and $\psi(\mathbf{L},\mathbf{T}) = V\log\det(\mathbf{I}+\mathbf{L}\mathbf{T}\mathbf{L}^{\mathsf{H}}) - Z(t)\mathrm{tr}(\mathbf{L}^{\mathsf{H}}\mathbf{L})$.

Fact 8.8. *Let $\mathbf{Q} \in \mathbb{S}_+^{N_T}$ have Cholesky decomposition $\mathbf{Q} = \mathbf{L}^{\mathsf{H}}\mathbf{L}$. Then, $\phi(\mathbf{Q},\mathbf{H}) = V\log\det(\mathbf{I}+\mathbf{L}\mathbf{T}\mathbf{L}^{\mathsf{H}}) - Z(t)tr(\mathbf{L}^{\mathsf{H}}\mathbf{L}) = \psi(\mathbf{L},\mathbf{T})$ with $\mathbf{T} = \mathbf{H}^{\mathsf{H}}\mathbf{H}$. Moreover, if $\mathbf{L}$ is fixed, then $\psi(\mathbf{L},\mathbf{T})$ is concave with respect to $\mathbf{T}$ and has gradient $\nabla_{\mathbf{T}}\psi(\mathbf{L},\mathbf{T}) = V\mathbf{L}^H(\mathbf{I}+\mathbf{L}\mathbf{T}\mathbf{L}^H)^{-1}\mathbf{L}$.*

Proof. Note that

$$\begin{aligned}
&V\log\det(\mathbf{I}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}) - Z(t)\mathrm{tr}(\mathbf{Q}) \\
=&V\log\det(\mathbf{I}+\mathbf{H}\mathbf{L}^{\mathsf{H}}\mathbf{L}\mathbf{H}^{\mathsf{H}}) - Z(t)\mathrm{tr}(\mathbf{L}^{\mathsf{H}}\mathbf{L}) \\
\overset{(a)}{=}&V\log\det(\mathbf{I}+\mathbf{L}\mathbf{H}^{\mathsf{H}}\mathbf{H}\mathbf{L}^{\mathsf{H}}) - Z(t)\mathrm{tr}(\mathbf{L}^{\mathsf{H}}\mathbf{L}) \\
\overset{(b)}{=}&V\log\det(\mathbf{I}+\mathbf{L}\mathbf{T}\mathbf{L}^{\mathsf{H}}) - Z(t)\mathrm{tr}(\mathbf{L}^{\mathsf{H}}\mathbf{L}) \\
=&\psi(\mathbf{L},\mathbf{T})
\end{aligned}$$

where (a) follows from the elementary identity $\det(\mathbf{I}+\mathbf{A}\mathbf{B}) = \det(\mathbf{I}+\mathbf{B}\mathbf{A})$ for any $\mathbf{A} \in \mathbb{C}^{m\times n}$ and $\mathbf{B} \in \mathbb{C}^{n\times m}$; and (b) follows from the definition $\mathbf{T} = \mathbf{H}^{\mathsf{H}}\mathbf{H}$.

Note that if $\mathbf{L}$ is fixed, then $Z(t)\mathrm{tr}(\mathbf{L}^{\mathsf{H}}\mathbf{L})$ is a constant. It follows from Fact 8.3 that $\psi(\mathbf{L},\mathbf{T})$ is concave with respect to $\mathbf{T}$ and has gradient $\nabla_{\mathbf{T}}\psi(\mathbf{L},\mathbf{T}) = V\mathbf{L}^H(\mathbf{I}+\mathbf{L}\mathbf{T}\mathbf{L}^H)^{-1}\mathbf{L}$. □

Let $\mathbf{Q}^*(\mathbf{H})$ be an optimal solution to the problem (8.2)-(8.4). Note that $\mathbf{Q}^*(\mathbf{H})$ is a mapping from channel states to transmit covariances and $R^{\text{opt}} = \mathbb{E}[\log\det(\mathbf{I} + \mathbf{H}\mathbf{Q}^*(\mathbf{H})\mathbf{H}^{\mathsf{H}})]$. To simplify notation, we denote $\mathbf{Q}^*(t) = \mathbf{Q}^*(\mathbf{H}(t))$, i.e. the transmit covariance at slot t selected according to $\mathbf{Q}^*(\mathbf{H})$. The next lemma relates the performance of Algorithm 8.1 and $\mathbf{Q}^*$ at each slot t.

Lemma 8.8. *Let $\mathbf{Q}(t)$ be yielded by Algorithm 8.1. At each slot t, we have $V\log\det(\mathbf{I} + \mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t)) - Z(t)tr(\mathbf{Q}(t)) \geq V\log\det(\mathbf{I}+\mathbf{H}(t)\mathbf{Q}^*(t)\mathbf{H}^{\mathsf{H}}(t)) - Z(t)tr(\mathbf{Q}^*(t)) - 2VP\sqrt{N_T}(2B+\delta)\delta$.*

Proof. Fix $t > 0$. Let $\widetilde{\mathbf{H}}(t) \in \mathbb{C}^{N_R\times N_T}$ be the observed (inaccurate) CSIT satisfying $\|\mathbf{H}(t) - \widetilde{\mathbf{H}}(t)\|_F \leq \delta$. The main proof of this lemma can be decomposed into 3 steps:

• **Step 1:** Show that $\phi(\mathbf{Q}(t), \mathbf{H}(t)) \geq \phi(\mathbf{Q}(t), \widetilde{\mathbf{H}}(t)) - VP\sqrt{N_T}(2B+\delta)\delta$. Let $\mathbf{Q}(t) = \mathbf{L}^{\mathsf{H}}(t)\mathbf{L}(t)$ be an Cholesky decomposition. Define $\mathbf{T}(t) = \mathbf{H}^{\mathsf{H}}(t)\mathbf{H}(t)$ and $\widetilde{\mathbf{T}}(t) = \widetilde{\mathbf{H}}^{\mathsf{H}}(t)\widetilde{\mathbf{H}}(t)$. By Fact 8.8, we have $\psi(\mathbf{L}(t), \mathbf{T}(t)) = \phi(\mathbf{Q}(t), \mathbf{H}(t))$ and $\psi(\mathbf{L}(t), \widetilde{\mathbf{T}}(t)) = \phi(\mathbf{Q}(t), \widetilde{\mathbf{H}}(t))$; and ψ is concave with respect to $\mathbf{T}$. By Fact 8.4, we have

$$
\begin{aligned}
&\psi(\mathbf{L}(t), \mathbf{T}(t)) \\
\geq&\psi(\mathbf{L}(t), \widetilde{\mathbf{T}}(t)) - \mathrm{tr}\big([\nabla_{\mathbf{T}}\psi(\mathbf{L}(t), \mathbf{T}(t))]^{\mathsf{H}}(\widetilde{\mathbf{T}}(t) - \mathbf{T}(t)\big) \\
\overset{(a)}{\geq}&\psi(\mathbf{L}(t), \widetilde{\mathbf{T}}(t)) - \|\nabla_{\mathbf{T}}\psi(\mathbf{L}(t), \mathbf{T}(t))\|_F\|\widetilde{\mathbf{T}}(t) - \mathbf{T}(t)\|_F \\
\overset{(b)}{\geq}&\psi(\mathbf{L}(t), \widetilde{\mathbf{T}}(t)) - V\|\mathbf{L}^{\mathsf{H}}(t)(\mathbf{I} + \mathbf{L}(t)\mathbf{T}(t)\mathbf{L}^{\mathsf{H}}(t))^{-1}\mathbf{L}(t)\|_F(2B+\delta)\delta \\
\overset{(c)}{\geq}&\psi(\mathbf{L}(t), \widetilde{\mathbf{T}}(t)) - VP\sqrt{N_T}(2B+\delta)\delta
\end{aligned}
$$

where (a) follows from part 4 in Fact 8.1; (b) follows from $\nabla_{\mathbf{T}}\psi(\mathbf{L}(t), \mathbf{T}(t)) = V\mathbf{L}^{\mathsf{H}}(t)(\mathbf{I} + \mathbf{L}(t)\mathbf{T}(t)\mathbf{L}^{\mathsf{H}}(t))^{-1}\mathbf{L}(t)$ by Fact 8.8 and $\|\widetilde{\mathbf{T}}(t) - \mathbf{T}(t)\|_F \leq \delta(2B+\delta)$ which is further implied by Fact 8.7; and (c) follows from $\|\mathbf{L}^{\mathsf{H}}(t)(\mathbf{I}+\mathbf{L}(t)\mathbf{T}(t)\mathbf{L}^{\mathsf{H}}(t))^{-1}\mathbf{L}(t)\|_F \leq \|\mathbf{L}^{\mathsf{H}}(t)\|_F^2\|(\mathbf{I}+\mathbf{L}(t)\mathbf{T}(t)\mathbf{L}^{\mathsf{H}}(t))^{-1}\|_F \leq P\sqrt{N_T}$ where the first inequality follows from Fact 8.1 and the second inequality follows from $\|\mathbf{L}(t)\|_F = \sqrt{\mathrm{tr}(\mathbf{L}^{\mathsf{H}}(t)\mathbf{L}(t))} = \sqrt{\mathrm{tr}(\mathbf{Q}(t))} \leq \sqrt{P}$ and Fact 8.6.

• **Step 2:** Show that $\phi(\mathbf{Q}(t), \widetilde{\mathbf{H}}(t)) \geq \phi(\mathbf{Q}^*(t), \widetilde{\mathbf{H}}(t))$. This step simply follows from the fact that Algorithm 8.1 choses $\mathbf{Q}(t)$ to maximize $\phi(\mathbf{Q}, \widetilde{\mathbf{H}}(t)) = V\log\det(\mathbf{I}+\widetilde{\mathbf{H}}(t)\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}(t)) - Z(t)\mathrm{tr}(\mathbf{Q})$ and hence $\mathbf{Q}(t)$ should be no worse than $\mathbf{Q}^*(t)$.

• **Step 3:** Show that $\phi(\mathbf{Q}^*(t), \widetilde{\mathbf{H}}(t)) \geq \phi(\mathbf{Q}^*(t), \mathbf{H}(t)) - VP\sqrt{N_T}(2B+\delta)\delta$. This step is similar to step 1. Let $\mathbf{Q}^*(t) = \mathbf{M}^{\mathsf{H}}(t)\mathbf{M}(t)$ be an Cholesky decomposition. Define $\mathbf{T}(t) = \mathbf{H}^{\mathsf{H}}(t)\mathbf{H}(t)$ and

$\widetilde{\mathbf{T}}(t) = \widetilde{\mathbf{H}}^{\mathsf{H}}(t)\widetilde{\mathbf{H}}(t)$. By Fact 8.8, we have $\psi(\mathbf{M}(t), \mathbf{T}(t)) = \phi(\mathbf{Q}^*(t), \mathbf{H}(t))$ and $\psi(\mathbf{M}(t), \widetilde{\mathbf{T}}(t)) = \phi(\mathbf{Q}^*(t), \widetilde{\mathbf{H}}(t))$; and ψ is concave with respect to $\mathbf{T}$. By Fact 8.4, we have

$$\begin{aligned}
&\psi(\mathbf{M}(t), \widetilde{\mathbf{T}}(t)) \\
\geq &\psi(\mathbf{M}(t), \mathbf{T}(t)) - \mathrm{tr}\big([\nabla_{\mathbf{T}}\psi(\mathbf{M}(t)), \widetilde{\mathbf{T}}(t)]^{\mathsf{H}}[\mathbf{T}(t) - \widetilde{\mathbf{T}}(t)]\big) \\
\overset{(a)}{\geq} &\psi(\mathbf{M}(t), \mathbf{T}(t)) - \|\nabla_{\mathbf{T}}\psi(\mathbf{M}(t), \widetilde{\mathbf{T}}(t))\|_F \|\mathbf{T}(t) - \widetilde{\mathbf{T}}(t)\|_F \\
\overset{(b)}{\geq} &\psi(\mathbf{M}(t), \mathbf{T}(t)) - V\|\mathbf{M}^{\mathsf{H}}(t)(\mathbf{I} + \mathbf{M}(t)\widetilde{\mathbf{T}}(t)\mathbf{M}^{\mathsf{H}}(t))^{-1}\mathbf{M}(t)\|_F(2B + \delta)\delta \\
\overset{(c)}{\geq} &\psi(\mathbf{M}(t), \mathbf{T}(t)) - VP\sqrt{N_T}(2B + \delta)\delta
\end{aligned}$$

where (a) follows from part 4 in Fact 8.1; (b) follows from $\nabla_{\mathbf{T}}\psi(\mathbf{M}(t), \widetilde{\mathbf{T}}(t)) = V\mathbf{M}^{\mathsf{H}}(t)(\mathbf{I} + \mathbf{M}(t)\widetilde{\mathbf{T}}(t)\mathbf{M}^{\mathsf{H}}(t))^{-1}\mathbf{M}(t)$ by Fact 8.8 and $\|\mathbf{T}(t) - \widetilde{\mathbf{T}}(t)\|_F \leq \delta(2B + \delta)$ which is further implied by Fact 8.7; and (c) follows from $\|\mathbf{M}^{\mathsf{H}}(t)(\mathbf{I} + \mathbf{M}(t)\widetilde{\mathbf{T}}(t)\mathbf{L}^{\mathsf{H}}(t))^{-1}\mathbf{M}(t)\|_F \leq \|\mathbf{M}^{\mathsf{H}}(t)\|_F^2\|(\mathbf{I} + \mathbf{M}(t)\widetilde{\mathbf{T}}(t)\mathbf{M}^{\mathsf{H}}(t))^{-1}\|_F \leq P\sqrt{N_T}$ where the first inequality follows from Fact 8.1 and the second inequality follows from $\|\mathbf{M}(t)\|_F = \sqrt{\mathrm{tr}(\mathbf{M}^{\mathsf{H}}(t)\mathbf{M}(t))} = \sqrt{\mathrm{tr}(\mathbf{Q}^*(t))} \leq \sqrt{P}$ and Fact 8.6.

Combining the above steps yields $\phi(\mathbf{Q}(t), \mathbf{H}(t)) \geq \phi(\mathbf{Q}^*(t), \mathbf{H}(t)) - 2VP\sqrt{N_T}(2B + \delta)\delta$. □

Lemma 8.9. *At each time $t \in \{0, 1, 2, \ldots\}$, we have*

$$-\Delta(t) \geq -Z(t)\big(tr(\mathbf{Q}(t)) - \bar{P}\big) - \frac{1}{2}\max\{\bar{P}^2, (P - \bar{P})^2\}. \tag{8.29}$$

Proof. Fix $t \in \{0, 1, 2, \ldots\}$. Note that $Z(t+1) = \max\{0, Z(t) + \mathrm{tr}(\mathbf{Q}(t)) - \bar{P}\}$ implies that

$$\begin{aligned}
Z^2(t+1) \leq &\big[Z(t) + \mathrm{tr}(\mathbf{Q}(t)) - \bar{P}\big]^2 \\
\leq &Z^2(t) + 2Z(t)\big(\mathrm{tr}(\mathbf{Q}(t)) - \bar{P}\big) + (\mathrm{tr}(\mathbf{Q}(t)) - \bar{P})^2 \\
\overset{(a)}{\leq} &Z^2(t) + 2Z(t)\big(\mathrm{tr}(\mathbf{Q}(t) - \bar{P}\big) + \max\{\bar{P}^2, (P - \bar{P})^2\}
\end{aligned}$$

where (a) follows from $|\mathrm{tr}(\mathbf{Q}(t)) - \bar{P}| \leq \max\{\bar{P}, P - \bar{P}\}$, which further follows from $0 \leq \mathrm{tr}(\mathbf{Q}(t)) \leq P$. Rearranging terms and dividing by factor 2 yields the desired result. □

Now, we are ready to present the main proof of Lemma 8.3. Adding $V \log\det(\mathbf{I} + \mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t))$

to both sides in (8.29) yields

$$\begin{aligned}
&-\Delta(t)+V\log\det(\mathbf{I}+\mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t))\\
\geq& V\log\det(\mathbf{I}+\mathbf{H}(t)\mathbf{Q}(t)\mathbf{H}^{\mathsf{H}}(t))-Z(t)\big(\mathrm{tr}(\mathbf{Q}(t))-\bar{P}\big)-\frac{1}{2}\max\{\bar{P}^2,(P-\bar{P})^2\}\\
\overset{(a)}{\geq}& V\log\det(\mathbf{I}+\mathbf{H}(t)\mathbf{Q}^*(t)\mathbf{H}^{\mathsf{H}}(t))-Z(t)\mathrm{tr}(\mathbf{Q}^*(t)-\bar{P})-\frac{1}{2}\max\{\bar{P}^2,(P-\bar{P})^2\}\\
&-2VP\sqrt{N_T}(2B+\delta)\delta
\end{aligned}$$

where (a) follows from Lemma 8.8.

Taking expectations on both sides yields

$$\begin{aligned}
&-\mathbb{E}[\Delta(t)]+V\mathbb{E}[R(t)]\\
\geq& VR^{\mathrm{opt}}-\mathbb{E}[Z(t)(\mathrm{tr}(\mathbf{Q}^*(t))-\bar{P})]-\frac{1}{2}\max\{\bar{P}^2,(P-\bar{P})^2\}-2VP\sqrt{N_T}(2B+\delta)\delta\\
\overset{(a)}{=}& VR^{\mathrm{opt}}-\mathbb{E}[\mathbb{E}[Z(t)(\mathrm{tr}(\mathbf{Q}^*(t))-\bar{P})|Z(t)]]-\frac{1}{2}\max\{\bar{P}^2,(P-\bar{P})^2\}-2VP\sqrt{N_T}(2B+\delta)\delta\\
\overset{(b)}{\geq}& VR^{\mathrm{opt}}-\frac{1}{2}\max\{\bar{P}^2,(P-\bar{P})^2\}-2VP\sqrt{N_T}(2B+\delta)\delta
\end{aligned}$$

where (a) follows by noting that $\mathbb{E}[Z(t)(\mathrm{tr}(\mathbf{Q}^*(t))-\bar{P})|Z(t)]$ is the expectation conditional on $Z(t)$ and the iterated law of expectations; and (b) follows from $\mathbb{E}[Z(t)\mathrm{tr}(\mathbf{Q}^*(t)-\bar{P})|Z(t)] = Z(t)\mathbb{E}[\mathrm{tr}(\mathbf{Q}^*(t))-\bar{P}]\leq 0$, where the identity follows because $\mathbf{Q}^*(t)$ only depends on $\mathbf{H}(t)$ and is independent of $Z(t)$, and the inequality follows because $Z(t)\geq 0$ and $\mathbb{E}[\mathrm{tr}(\mathbf{Q}^*(t))-\bar{P}]\leq 0, \forall t$.

Rearranging terms and dividing both sides by V yields $-\frac{1}{V}\mathbb{E}[\Delta(t)]+\mathbb{E}[R(t)]\geq R^{\mathrm{opt}}-\frac{\max\{\bar{P}^2,(P-\bar{P})^2\}}{2V}-2P\sqrt{N_T}(2B+\delta)\delta$.

8.7.4 Proof of Lemma 8.5

A problem similar to the problem (8.17)-(8.19) (with inequality constraint (8.18) replaced by the equality constraint $\mathrm{tr}(\mathbf{Q})=\bar{P}$) is considered in Lemma 14 in [SPB09]. The problem in [SPB09] is different from (8.17)-(8.19) since inequality constraint (8.18) is not necessarily tight at the optimal solution to (8.17)-(8.19). However, the proof flow of the current lemma is similar to [SPB09]. We shall first reduce the problem (8.17)-(8.19) to a simpler convex program with a real vector variable by characterizing the structure of its optimal solution. After that, we can derive an (almost) closed-form solution to the simpler convex program by studying its KKT conditions.

The details of the proof are as follows:

Claim 8.2. *If $\widehat{\mathbf{\Theta}}$ is an optimal solution to the following convex program:*

$$\min \quad \frac{1}{2}\|\mathbf{\Theta} - \mathbf{\Sigma}\|_F^2 \tag{8.30}$$

$$s.t. \quad tr(\mathbf{\Theta}) \leq \bar{P} \tag{8.31}$$

$$\mathbf{\Theta} \in \mathbb{S}_+^{N_T} \tag{8.32}$$

then $\widehat{\mathbf{Q}} = \mathbf{U}^{\mathsf{H}}\widehat{\mathbf{\Theta}}\mathbf{U}$ is an optimal solution to the problem (8.17)-(8.19).

Proof. This claim can be proven by contradiction. Let $\widehat{\mathbf{\Theta}}$ be an optimal solution to convex program (8.30)-(8.32) and define $\widehat{\mathbf{Q}} = \mathbf{U}^{\mathsf{H}}\widehat{\mathbf{\Theta}}\mathbf{U}$. Assume that there exists $\widetilde{\mathbf{Q}} \in \mathbb{S}_+^{N_T}$ such that $\widetilde{\mathbf{Q}} \neq \widehat{\mathbf{Q}}$ and is a feasible solution to the problem (8.17)-(8.19) that is strictly better than $\widehat{\mathbf{Q}}$. Consider $\widetilde{\mathbf{\Theta}} = \mathbf{U}\widetilde{\mathbf{Q}}\mathbf{U}^{\mathsf{H}}$ and reach a contradiction by showing $\widetilde{\mathbf{\Theta}}$ is strictly better than $\widehat{\mathbf{\Theta}}$ as follows:

Note that $\mathrm{tr}(\widetilde{\mathbf{\Theta}}) = \mathrm{tr}(\mathbf{U}\widetilde{\mathbf{Q}}\mathbf{U}^{\mathsf{H}}) = \mathrm{tr}(\widetilde{\mathbf{Q}}) \leq \bar{P}$, where the last inequality follows from the assumption that $\widetilde{\mathbf{Q}}$ is a feasible solution to the problem (8.17)-(8.19). Also note that $\widetilde{\mathbf{\Theta}} \in \mathbb{S}_+^{N_T}$ since $\widetilde{\mathbf{Q}} \in \mathbb{S}_+^{N_T}$. Thus, $\widetilde{\mathbf{\Theta}}$ is feasible to the problem (8.30)-(8.32).

Note that $\|\widetilde{\mathbf{\Theta}} - \mathbf{\Sigma}\|_F \overset{(a)}{=} \|\mathbf{U}^{\mathsf{H}}\widetilde{\mathbf{\Theta}}\mathbf{U} - \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}\|_F \overset{(b)}{=} \|\widetilde{\mathbf{Q}} - \mathbf{X}\|_F \overset{(c)}{<} \|\widehat{\mathbf{Q}} - \mathbf{X}\|_F \overset{(d)}{=} \|\mathbf{U}\widehat{\mathbf{Q}}\mathbf{U}^{\mathsf{H}} - \mathbf{U}\mathbf{X}\mathbf{U}^{\mathsf{H}}\|_F \overset{(e)}{=} \|\widehat{\mathbf{\Theta}} - \mathbf{\Sigma}\|_F$, where (a) and (d) follow from the fact that Frobenius norm is unitary invariant[9]; (b) follows from the fact that $\widetilde{\mathbf{\Theta}} = \mathbf{U}\widetilde{\mathbf{Q}}\mathbf{U}^{\mathsf{H}}$ and $\mathbf{X} = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$; (c) follows from the fact that $\widetilde{\mathbf{Q}}$ is strictly better than $\widehat{\mathbf{Q}}$; and (e) follows from the fact that $\widehat{\mathbf{Q}} = \mathbf{U}^{\mathsf{H}}\widehat{\mathbf{\Theta}}\mathbf{U}$ and $\mathbf{X} = \mathbf{U}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{U}$. Thus, $\widetilde{\mathbf{\Theta}}$ is strictly better than $\widehat{\mathbf{\Theta}}$. A contradiction! □

Claim 8.3. *The optimal solution to the problem* (8.30)-(8.32) *must be a diagonal matrix.*

Proof. This claim can be proven by contradiction. Assume that the problem (8.30)-(8.32) has an optimal solution $\widetilde{\mathbf{\Theta}}$ that is not diagonal. Since $\widetilde{\mathbf{\Theta}}$ is positive semidefinite, all the diagonal entries of $\widetilde{\mathbf{\Theta}}$ are non-negative. Define $\widehat{\mathbf{\Theta}}$ as a diagonal matrix whose the i-th diagonal entry is equal to the i-th diagonal entry of $\widetilde{\mathbf{\Theta}}$ for all $i \in \{1, 2, \ldots, N_T\}$. Note that $\mathrm{tr}(\widehat{\mathbf{\Theta}}) = \mathrm{tr}(\widetilde{\mathbf{\Theta}}) \leq \bar{P}$ and $\widehat{\mathbf{\Theta}} \in \mathbb{S}_+^n$. Thus, $\widehat{\mathbf{\Theta}}$ is feasible to the problem (8.30)-(8.32). Note that $\|\widehat{\mathbf{\Theta}} - \mathbf{\Sigma}\|_F < \|\widetilde{\mathbf{\Theta}} - \mathbf{\Sigma}\|_F$ since $\mathbf{\Sigma}$ is diagonal. Thus, $\widehat{\mathbf{\Theta}}$ is a solution strictly better than $\widetilde{\mathbf{\Theta}}$. A contradiction! So the optimal solution to the problem (8.30)-(8.32) must be a diagonal matrix. □

[9]That is $\|\mathbf{A}\mathbf{U}\|_F = \|\mathbf{A}\|_F$ for all $\mathbf{A} \in \mathbb{C}^{n\times n}$ and all unitary matrix $\mathbf{U}$.

By the above two claims, it suffices to assume that the optimal solution to the problem (8.17)-(8.19) has the structure $\hat{\mathbf{Q}} = \mathbf{U}^{\mathsf{H}}\mathbf{\Theta}\mathbf{U}$, where $\mathbf{\Theta}$ is a diagonal with non-negative entries $\theta_1, \ldots, \theta_{N_T}$. To solve the problem (8.17)-(8.19), it suffices to consider the following convex program.

$$\min \quad \frac{1}{2}\sum_{i=1}^{N_T}(\theta_i - \sigma_i)^2 \tag{8.33}$$

$$\text{s.t.} \quad \sum_{i=1}^{N_T}\theta_i \leq \bar{P} \tag{8.34}$$

$$\theta_i \geq 0, \forall i \in \{1, 2, \ldots, N_T\} \tag{8.35}$$

Note that the problem (8.33)-(8.35) satisfies Slater's condition. So the optimal solution to the problem (8.33)-(8.35) is characterized by KKT conditions [BV04]. Introducing Lagrange multipliers $\mu \in \mathbb{R}_+$ for inequality constraint $\sum_{i=1}^{N_T}\theta_i \leq \bar{P}$ and $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_{N_T}]^{\mathsf{T}} \in \mathbb{R}_+^{N_T}$ for inequality constraints $\theta_i \geq 0, i \in \{1, 2, \ldots, n\}$. Let $\boldsymbol{\theta}^* = [\theta_1^*, \ldots, \theta_{N_T}^*]^{\mathsf{T}}$ and $(\mu^*, \boldsymbol{\nu}^*)$ be any primal and dual pair with the zero duality gap. By KKT conditions, we have $\theta_i^* - \sigma_i + \mu^* - \nu_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}; \sum_{i=1}^{N_T}\theta_i^* \leq \bar{P}; \mu^* \geq 0; \mu^*\big(\sum_{i=1}^{N_T}\theta_i^* - \bar{P}\big) = 0; \theta_i^* \geq 0, \forall i \in \{1, 2, \ldots, N_T\}; \nu_i^* \geq 0, \forall i \in \{1, 2, \ldots, N_T\}; \nu_i^*\theta_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}$.

Eliminating $\nu_i^*, \forall i \in \{1, 2, \ldots, N_T\}$ in all equations yields $\mu^* \geq \sigma_i - \theta_i^*, i \in \{1, 2, \ldots, N_T\}$; $\sum_{i=1}^{N_T}\theta_i^* \leq \bar{P}$; $\mu^* \geq 0$; $\mu^*\big[\sum_{i=1}^{N_T}\theta_i^* - \bar{P}\big] = 0$; $\theta_i^* \geq 0, \forall i \in \{1, 2, \ldots, N_T\}$; $(\theta_i^* - \sigma_i + \mu^*)\theta_i^* = 0, \forall i \in \{1, 2, \ldots, N_T\}$.

For all $i \in \{1, 2, \ldots, N_T\}$, we consider $\mu^* < \sigma_i$ and $\mu^* \geq \sigma_i$ separately:

1. If $\mu^* < \sigma_i$, then $\mu^* \geq \sigma_i - \theta_i^*$ holds only when $\theta_i^* > 0$, which by $(\theta_i^* - \sigma_i + \mu^*)\theta_i^* = 0$ implies that $\theta_i^* = \sigma_i - \mu^*$.

2. If $\mu^* \geq \sigma_i$, then $\theta_i^* > 0$ is impossible, because $\theta_i^* > 0$ implies that $\theta_i^* - \sigma_i + \mu^* > 0$, which together with $\theta_i^* > 0$ contradicts the slackness condition $(\theta_i^* - \sigma_i + \mu^*)\theta_i^* = 0$. Thus, if $\mu^* \geq \sigma_i$, we must have $\theta_i^* = 0$.

Summarizing both cases, we have $\theta_i^* = \max\{0, \sigma_i - \mu^*\}, \forall i \in \{1, 2, \ldots, N_T\}$, where μ^* is chosen such that $\sum_{i=1}^{N_T}\theta_i^* \leq \bar{P}$, $\mu^* \geq 0$ and $\mu^*\big(\sum_{i=1}^{N_T}\theta_i^* - \bar{P}\big) = 0$.

To find such μ^*, we first check if $\mu^* = 0$. If $\mu^* = 0$ is true, the slackness condition $\mu^*\big(\sum_{i=1}^{N_T}\theta_i^* - \bar{P}\big)$ is guaranteed to hold and we need to further require $\sum_{i=1}^{N_T}\theta_i^* = \sum_{i=1}^{N_T}\max\{0, \sigma_i\} \leq \bar{P}$. Thus $\mu^* = 0$ if and only if $\sum_{i=1}^{n}\max\{0, \sigma_i\} \leq \bar{P}$. Note that Algorithm 8.4 checks if

$\sum_{i=1}^{N_T} \max\{0, \sigma_i\} \leq \bar{P}$ holds at the first step and if this is true, then we conclude $\mu^* = 0$ and we are done!

Otherwise, we know $\mu^* > 0$. By the slackness condition $\mu^*\big(\sum_{i=1}^{N_T} \theta_i^* - \bar{P}\big) = 0$, we must have $\sum_{i=1}^{N_T} \theta_i^* = \sum_{i=1}^{N_T} \max\{0, \sigma_i - \mu^*\} = \bar{P}$. To find $\mu^* > 0$ such that $\sum_{i=1}^{N_T} \max\{0, \sigma_i - \mu^*\} = \bar{P}$, we could apply a bisection search by noting that all θ_i^* are decreasing with respect to μ^*.

Another algorithm of finding μ^* is inspired by the observation that if $\sigma_j \geq \sigma_k, \forall j, k \in \{1, 2, \ldots, N_T\}$, then $\theta_j^* \geq \theta_k^*$. Thus, we first sort all σ_i in a decreasing order, say π is the permutation such that $\sigma_{\pi(1)} \geq \sigma_{\pi(2)} \geq \cdots \geq \sigma_{\pi(N_T)}$; and then sequentially check if $i \in \{1, 2, \ldots, N_T\}$ is the index such that $\sigma_{\pi(i)} - \mu^* \geq 0$ and $\sigma_{\pi(i+1)} - \mu^* < 0$. To check this, we first assume i is indeed such an index and solve the equation $\sum_{j=1}^{i} \big(\sigma_{\pi(j)} - \mu^*\big) = \bar{P}$ to obtain μ^*; (Note that in Algorithm 8.4, to avoid recalculating the partial sum $\sum_{j=1}^{i} \sigma_{\pi(j)}$ for each i, we introduce the parameter $S_i = \sum_{j=1}^{i} \sigma_{\pi(j)}$ and update S_i incrementally. By doing this, the complexity of each iteration in the loop is only $O(1)$.) then verify the assumption by checking if $\mu^* \geq 0$, $\sigma_{\pi(i)} - \mu^* \geq 0$ and $\sigma_{\pi(i+1)} - \mu^* \leq 0$. The algorithm is described in Algorithm 8.4 and has complexity $O(N_T \log(N_T))$. The overall complexity is dominated by the step of sorting all σ_i.

8.7.5 Proof of Lemma 8.6

Proof of Part 1:

The boundedness of $\mathbf{D}(t-1)$ can be shown as follows. $\|\mathbf{D}(t-1)\|_F = \|\mathbf{H}^{\mathsf{H}}(t-1)(\mathbf{I}_{N_R} + \mathbf{H}(t-1)\mathbf{Q}(t-1)\mathbf{H}^{\mathsf{H}}(t-1))^{-1}\mathbf{H}(t-1)\|_F \overset{(a)}{\leq} \|\mathbf{H}(t-1)\|_F^2 \|(\mathbf{I}_{N_R} + \mathbf{H}(t-1)\mathbf{Q}(t-1)\mathbf{H}^{\mathsf{H}}(t-1))^{-1}\|_F \overset{(b)}{\leq} \sqrt{N_R}B^2$, where (a) follows from Fact 8.1 and (b) follows from $\|\mathbf{H}(t-1)\|_F \leq B$ and Fact 8.6.

Proof of Part 2:

To simplify the notation, this part uses $\mathbf{H}$, $\widetilde{\mathbf{H}}$ and $\mathbf{Q}$ to represent $\mathbf{H}(t-1)$, $\widetilde{\mathbf{H}}(t-1)$ and $\mathbf{Q}(t-1)$, respectively.

Note that

$$
\begin{aligned}
&\|\mathbf{D}(t-1)-\widetilde{\mathbf{D}}(t-1)\|_F \\
=&\|\mathbf{H}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\mathbf{H}-\widetilde{\mathbf{H}}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\big)^{-1}\widetilde{\mathbf{H}}\|_F \\
\leq&\|\mathbf{H}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\mathbf{H}-\widetilde{\mathbf{H}}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\mathbf{H}\|_F \\
&+\|\widetilde{\mathbf{H}}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\mathbf{H}-\widetilde{\mathbf{H}}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\widetilde{\mathbf{H}}\|_F \\
&+\|\widetilde{\mathbf{H}}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\widetilde{\mathbf{H}}-\widetilde{\mathbf{H}}^{\mathsf{H}}\big(\mathbf{I}_{N_R}+\widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\big)^{-1}\widetilde{\mathbf{H}}\|_F \\
\leq&\|\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\|_F\|\mathbf{H}\|_F\|\mathbf{H}-\widetilde{\mathbf{H}}\|_F+\|\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\|_F\|\widetilde{\mathbf{H}}\|_F\|\mathbf{H}-\widetilde{\mathbf{H}}\|_F \\
&+\|\widetilde{\mathbf{H}}\|_F^2\|\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}-\big(\mathbf{I}_{N_R}+\widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\big)^{-1}\|_F \qquad (8.36)
\end{aligned}
$$

where both inequalities follow from Fact 8.1.

Since $\|\mathbf{H}\|_F \leq B$ and $\|\widetilde{\mathbf{H}}-\mathbf{H}\| \leq \delta$, by Fact 8.1, we have $\|\widetilde{\mathbf{H}}\|_F \leq B+\delta$. By Fact 8.6, we have $\|\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\|_F \leq \sqrt{N_R}$. The following lemma from [SPB09] will be useful to bound $\|\big(\mathbf{I}_{N_R}+\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}-\big(\mathbf{I}_{N_R}+\widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\big)^{-1}\|_F$ from above.

Lemma 8.10 (Lemma 6 in [SPB09])**.** *Let $\mathbf{F}: \mathcal{D} \subseteq \mathbb{C}^{m\times n} \to \mathbb{C}^{p\times q}$ be a complex matrix-valued function defined on a convex set $\mathcal{D}$, assumed to be continuous on $\mathcal{D}$ and differentiable on the interior of $\mathcal{D}$, with Jacobian matrix*[10] *$\mathbf{D}_{\mathbf{X}}\mathbf{F}(\mathbf{X})$. Then, for any given $\mathbf{X},\mathbf{Y} \in \mathcal{D}$, there exists some $t \in (0,1)$ such that $\|F(\mathbf{Y})-F(\mathbf{X})\|_F \leq \|\mathbf{D}_{\mathbf{X}}\mathbf{F}(t\mathbf{Y}+(1-t)\mathbf{X})vec(\mathbf{Y}-\mathbf{X})\|_2 \leq \|\mathbf{D}_{\mathbf{X}}\mathbf{F}(t\mathbf{Y}+(1-t)\mathbf{X})\|_{2,mat}\|\mathbf{Y}-\mathbf{X}\|_F$, where $\|\mathbf{A}\|_{2,mat}$ denotes the spectral norm of $\mathbf{A}$, i.e., the largest singular value of $\mathbf{A}$.*

Lemma 8.10 is essentially a mean value theorem for complex matrix valued functions. The next corollary is the complex matrix version of elementary inequality $|\frac{1}{1+x}-\frac{1}{1+y}| \leq |x-y|, \forall x,y \geq 0$ and follows directly from Lemma 8.10.

Corollary 8.1. *Consider $\mathbf{F}: \mathbb{S}_+^n \to \mathbb{S}_+^n$ defined via $\mathbf{F}(\mathbf{X}) = (\mathbf{I}_n+\mathbf{X})^{-1}$. Then, $\|F(\mathbf{Y})-F(\mathbf{X})\|_F \leq n\|\mathbf{Y}-\mathbf{X}\|_F, \forall \mathbf{X},\mathbf{Y} \in \mathbb{S}_+^n$.*

Proof. By [HG07, SPB09], $d\mathbf{X}^{-1} = -\mathbf{X}^{-1}(d\mathbf{X})\mathbf{X}^{-1}$. Thus, $d(\mathbf{I}+\mathbf{X})^{-1} = -(\mathbf{I}+\mathbf{X})^{-1}(d\mathbf{X})(\mathbf{I}+\mathbf{X})^{-1}$. By identity $\mathrm{vec}(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})\mathrm{vec}(\mathbf{B})$, where $\otimes$ denotes the Kronecker product, we

[10]The Jacobian matrix is defined as the matrix $\mathbf{D}_{\mathbf{X}}\mathbf{F}(\mathbf{X})$ such that $d\mathrm{vec}(\mathbf{F}(\mathbf{x})) = \mathbf{D}_{\mathbf{X}}\mathbf{F}(\mathbf{X})\, d\mathrm{vec}(\mathbf{X})$. Note that the size of $\mathbf{D}_{\mathbf{X}}\mathbf{F}(\mathbf{X})$ is $pq \times mn$.

have $d\mathrm{vec}(\mathbf{F}(\mathbf{X})) = -\big(((\mathbf{I}+\mathbf{X})^{-1})^T \otimes (\mathbf{I}+\mathbf{X})^{-1}\big)\, d\mathrm{vec}(\mathbf{X})$. Thus, $\mathbf{D}_{\mathbf{X}}\mathbf{F}(\mathbf{X}) = -((\mathbf{I}+\mathbf{X})^{-1})^T \otimes (\mathbf{I}+\mathbf{X})^{-1}$. Note that for all $\mathbf{X} \in \mathbb{S}_+^n$, $\| -((\mathbf{I}+\mathbf{X})^{-1})^T \otimes (\mathbf{I}+\mathbf{X})^{-1}\|_{2,\mathrm{mat}} \leq \|((\mathbf{I}+\mathbf{X})^{-1})^T \otimes (\mathbf{I}+\mathbf{X})^{-1}\|_F \overset{(a)}{=} \|((\mathbf{I}+\mathbf{X})^{-1})^T\|_F \cdot \|(\mathbf{I}+\mathbf{X})^{-1}\|_F = \|(\mathbf{I}+\mathbf{X})^{-1}\|_F^2 \overset{(b)}{\leq} n$, where (a) follows from the fact that $\|\mathbf{A}\otimes\mathbf{B}\|_F = \|\mathbf{A}\|_F \cdot \|\mathbf{B}\|_F, \forall \mathbf{A} \in \mathbb{C}^{m\times n}, \mathbf{B} \in \mathbb{C}^{n\times l}$ (see Exercise 28, page 253 in [HJ91]); and (b) follows Fact 8.6. Applying Lemma 8.10 yields $\|F(\mathbf{Y}) - F(\mathbf{X})\|_F \leq n\|\mathbf{Y}-\mathbf{X}\|_F, \forall \mathbf{X}, \mathbf{Y} \in \mathbb{S}_+^n$. □

Applying the above corollary yields

$$
\begin{aligned}
&\|\big(\mathbf{I}_{N_R} + \mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1} - \big(\mathbf{I}_{N_R} + \widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\big)^{-1}\|_F \\
\overset{(a)}{\leq}& N_R\|\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\|_F \\
=& N_R\|\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}\mathbf{Q}\mathbf{H}^{\mathsf{H}} + \widetilde{\mathbf{H}}\mathbf{Q}\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\|_F \\
\overset{(b)}{\leq}& N_R\big(\|\mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\|_F + \|\widetilde{\mathbf{H}}\mathbf{Q}\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\|_F\big) \\
\overset{(c)}{\leq}& N_R\big(\|\mathbf{Q}\|_F\|\mathbf{H}^{\mathsf{H}}\|_F\|\mathbf{H}-\widetilde{\mathbf{H}}\|_F + \|\widetilde{\mathbf{H}}\|_F\|\mathbf{Q}\|_F\|\mathbf{H}^{\mathsf{H}} - \widetilde{\mathbf{H}}^{\mathsf{H}}\|_F\big) \\
\overset{(d)}{\leq}& N_R\bar{P}(2B+\delta)\delta
\end{aligned}
$$

where (a) follows from Corollary 8.1; (b) and (c) follows from Fact 8.1; and (d) follows from the fact that $\|\mathbf{H}\|_F \leq B$ and $\|\widetilde{\mathbf{H}} - \mathbf{H}\|_F \leq \delta$, $\|\widetilde{\mathbf{H}}\|_F \leq B+\delta$, and the fact that $\|\mathbf{Q}\|_F \leq \mathrm{tr}(\mathbf{Q}) \leq \bar{P}$, which is implied by Fact 8.2 and $\mathbf{Q} \in \widetilde{\mathcal{Q}}$.

Plugging equations $\|\widetilde{\mathbf{H}}\|_F \leq B+\delta$, $\|\big(\mathbf{I}_{N_R} + \mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1}\|_F \leq \sqrt{N_R}$ and $\|\big(\mathbf{I}_{N_R} + \mathbf{H}\mathbf{Q}\mathbf{H}^{\mathsf{H}}\big)^{-1} - \big(\mathbf{I}_{N_R} + \widetilde{\mathbf{H}}\mathbf{Q}\widetilde{\mathbf{H}}^{\mathsf{H}}\big)^{-1}\|_F \leq N_R\bar{P}(2B+\delta)\delta$ into equation (8.36) yields $\|\mathbf{D}(t-1) - \widetilde{\mathbf{D}}(t-1)\|_F \leq \sqrt{N_R}B\delta + \sqrt{N_R}(B+\delta)\delta + (B+\delta)^2 N_R\bar{P}(2B+\delta)\delta = \big(\sqrt{N_R}B + \sqrt{N_R}(B+\delta) + (B+\delta)^2 N_R\bar{P}(2B+\delta)\big)\delta$.

Proof of Part 3:

This part follows from $\|\widetilde{\mathbf{D}}(t-1)\|_F \leq \|\widetilde{\mathbf{D}}(t-1) - \mathbf{D}(t-1)\|_F + \|\mathbf{D}(t-1)\|_F$.

Chapter 9

Duality Codes and the Integrality Gap Bound for Index Coding

Consider a noiseless wireless system with N receivers, W independent packets of the same size, and a single broadcast station. The broadcast station has all packets. Each receiver has a subset of the packets as side information, but desires another (disjoint) subset of the packets. The broadcast station must deliver the packets to their intended receivers. To this end, it makes a sequence of (possibly coded) transmissions that are overheard by all receivers. The goal is to find a coding scheme with the minimum number of transmissions (clearance time) such that each user is able to decode its demanded packets. This problem was introduced by Birk and Kol in [BK98, BK06] and is known as the index coding problem.

The formulation of the index coding problem is simple, elegant and captures the essence of broadcasting with side information. It also relates directly to multi-hop network coding problems. Specifically, work in [RSG10] shows that an index coding problem can be reduced to a network coding problem. A partial converse of this result is also shown in [RSG10], in that linear versions of network coding can be redued to linear index coding (see [ERL15] for extended results in this direction). However, the index coding problem still seems to be intractable. The first index coding problem investigated by Birk and Kol considers only the case of unicast packets and can be represented as a directed side information graph. Work by Bar-Yossef et. al. in [BYBJK11] shows that the performance of the best *scalar linear code* is equal to the graph parameter *minrank* of the side information graph. However, computing the minrank of a given graph is NP-hard [Pee96]. Further, it is known that restricting to scalar linear codes is generally sub-optimal [ALS$^+$08, LS09].

One branch of research on index coding aims to find tight performance bounds. Work in [BYBJK11] shows that if the index coding problem has an *undirected* side information graph (such as when it has symmetric demands) then the minrank is lower-bounded by the independence number of the graph, and upper-bounded by the clique cover number. For the unicast index coding problem, work in [BYBJK11] shows that the optimal clearance time (with respect to any scalar, vector or non-linear code) is lower-bounded by the maximum acyclic subgraph of the side information graph. Work in [NTZ13] generalizes this to the multicast/groupcast case using a directed bipartite graph. It shows that the optimum of the general problem is lower-bounded by the maximum acyclic subgraph induced by deletions of packet vertices, user-vertices and packet-to-user arcs. In [BKL10], a sequence of linear programs is proposed to bound the optimal clearance time.

Another branch of research on index coding focuses on studying the performance of specific codes and specific graph structures. Work in [ALS$^+$08] shows that vector linear codes can have strictly better performance compared with scalar linear codes. Work in [LS09] demonstrates that non-linear codes can outperform both scalar and vector linear codes. Instead of finding the minimum clearance time, Chaudhry et. al. in [CASL11] consider the problem of maximizing the total number of saved transmissions by exploiting a specific code structure together with graph theory algorithms. Ong et. al. in [OH12] find the optimal index code in the *single uniprior* case, where each user only has a single uniprior packet as side information.

This chapter studies index coding from a perspective of optimization and duality. The results in this chapter are originally developed in our papers [YN13, YN14]. This chapter illustrates the inherent duality between the information theoretical lower bound in [BYBJK11][NTZ13] and the performance of specific codes. Section 9.1 extends the bipartite digraph representation of the problem in [NTZ13] to a weighted bipartite digraph. Section 9.2 uses this new graph structure to develop an integer linear program that finds the maximum acyclic subgraph. Section 9.3 considers the linear programming (LP) relaxation of the integer program, and shows that the dual problem of this relaxation corresponds to a simple form of vector linear codes, called vector cyclic codes. It follows that the information theoretic optimum is bounded by the *integrality gap* between the integer program and its LP relaxation. Section 9.4 shows that in the special case when the bipartite digraph is planar, the integrality gap is zero. In this case, optimality is achieved by a scalar cyclic code. Section 9.5 considers a different representation of the original

integer program that yields a smaller integrality gap. The dual problem of its LP relaxation leads to a more sophisticated *partial clique coding strategy* that time-shares between maximum distance separable (MDS) codes. The smaller integrality gap ensures that these codes are closer to the lower bound. These results provide new insight into the index coding problem and suggest that good codes can be found by exploring LP relaxations of the maximum acyclic subgraph problem.

9.1 Weighted Bipartite Digraph

There are N receivers, also called *users*. Let $\mathcal{U} = \{u_1, \ldots, u_N\}$ be the set of users. Assume there are W total packets, all of the same size, labeled $\{q_1, \ldots, q_W\}$. For each $m \in \{1, \ldots, W\}$, define $\mathcal{S}_m$ as the set of users in $\mathcal{U}$ that already have packet q_m as side information, and define $\mathcal{D}_m$ as the set of users in $\mathcal{U}$ that demand packet q_m. Without loss of generality, assume that each packet is demanded by at least one user (else, that packet can be eliminated). Thus, the demand set $\mathcal{D}_m$ is non-empty for all $m \in \{1, \ldots, W\}$. On the other hand, the side information sets $\mathcal{S}_m$ can be empty. Indeed, the set $\mathcal{S}_m$ is empty if and only if no user has packet q_m as side information. It is reasonable to assume that the set of users that demand a packet is disjoint from the set of users that already have that packet as side information, so that $\mathcal{S}_m \cap \mathcal{D}_m = \emptyset$ for all $m \in \{1, \ldots, W\}$.

This index coding problem is represented by a bipartite directed graph in [NTZ13][TDN12], where user vertices are on the left of the graph, packet vertices are on the right, and the $\mathcal{S}_m$ and $\mathcal{D}_m$ sets are represented by directed arcs. A directed graph is also called a *digraph*. It is useful to extend this representation to a *weighted* bipartite digraph as follows: Two packets q_k and q_m are said to have the same *type* if $\mathcal{S}_k = \mathcal{S}_m$ and $\mathcal{D}_k = \mathcal{D}_m$. That is, two packets have the same type if they have the same side information and demand sets. Types arise naturally when users desire multi-packet files, since packets of the same file typically have the same type.

Let M be the number of packet types, and let $\mathcal{P} = \{p_1, \ldots, p_M\}$ be the set of types. The index coding problem can be represented by a weighted bipartite digraph $\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{A}, \mathcal{W}_{\mathcal{P}})$ as follows: Let $\mathcal{U}$ be the set of vertices on the left side of the graph and let $\mathcal{P}$ be the set of vertices on the right side of the graph (see Fig. 9.1). The arc set $\mathcal{A}$ has a user-to-packet arc (u_n, p_m) if and only if user $u_n \in \mathcal{U}$ has all packets of type p_m. The arc set $\mathcal{A}$ has a packet-to-user arc

(p_m, u_n) if and only if user $u_n \in \mathcal{U}$ demands all packets of type p_m. Finally, define $\mathcal{W}_\mathcal{P}$ as the set of integer weights associated with packet vertices in $\mathcal{P}$. The weight $w_{p_m} \in \mathcal{W}_\mathcal{P}$ of packet vertex $p_m \in \mathcal{P}$ is equal to the number of packets of type p_m. Thus, the total number of packets W satisfies $W = \sum_{m=1}^{M} w_{p_m}$.

A packet is said to be a *unicast packet* if it is demanded by only one user, and is said to be a *groupcast packet* if it is demanded by two or more users. An index coding problem is said to be *unicast* if all packets are unicast packets. The index coding problems treated in [BK98][BYBJK11] are unicast problems. The current chapter also focuses exclusively on the unicast case. However, rather than use the graph structure of [BYBJK11], for our purposes it is more efficient to use a weighted bipartite digraph.[1] Figure 9.1 shows an example of the weighted bipartite digraph representation for a unicast index coding problem with 3 user vertices and 3 packet types. In this example, packet types p_1, p_2, p_3 are demanded by users u_1, u_2, u_3, respectively, so that $\mathcal{D}_1 = \{u_1\}$, $\mathcal{D}_2 = \{u_2\}$, $\mathcal{D}_3 = \{u_3\}$. Furthermore, the side information sets are as follows:

- Packets of type p_1 are contained as side information by users in the set $\mathcal{S}_1 = \{u_2, u_3\}$.
- Packets of type p_2 are contained as side information by the user in the set $\mathcal{S}_2 = \{u_3\}$.
- Packets of type p_3 are contained as side information by the user in the set $\mathcal{S}_3 = \{u_1\}$.

9.2 Acyclic Subgraph Bound and its LP Relaxation

The following definitions from graph theory are useful. A sequence of vertices $\{s_1, s_2, \ldots, s_K\}$ of a general digraph is defined as a *cycle* if $(s_i, s_{i+1}) \in \mathcal{A}$ for all $i \in \{1, 2, \ldots, K-1\}$, all vertices in $\{s_1, s_2, \ldots, s_{K-1}\}$ are distinct, and $s_1 = s_K$. A digraph is *acyclic* if it contains no cycle. A set of vertices is called a *feedback vertex set* if the removal of vertices in this set leaves an acyclic digraph. In a vertex-weighted digraph, the feedback vertex set with the minimum sum weight is called the *minimum feedback vertex set*.

For the weighted bipartite digraph $\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{A}, \mathcal{W}_\mathcal{P})$ (as defined in the previous section), there exists a subset $\mathcal{P}_{\text{fd}} \subseteq \mathcal{P}$ such that the removal of vertices in $\mathcal{P}_{\text{fd}}$ and all the associated

[1] The unicast problem can be represented by the graph structure in [BYBJK11] by changing each user that desires more than one packet into multiple *virtual users* that each want a single packet. This can significantly expand the size of the graph, particularly when users want large multi-packet files. The graph structure in the current chapter does not expand the number of users; this is conceptually simpler and is useful for proving optimality in some cases (see Corollary 9.2).

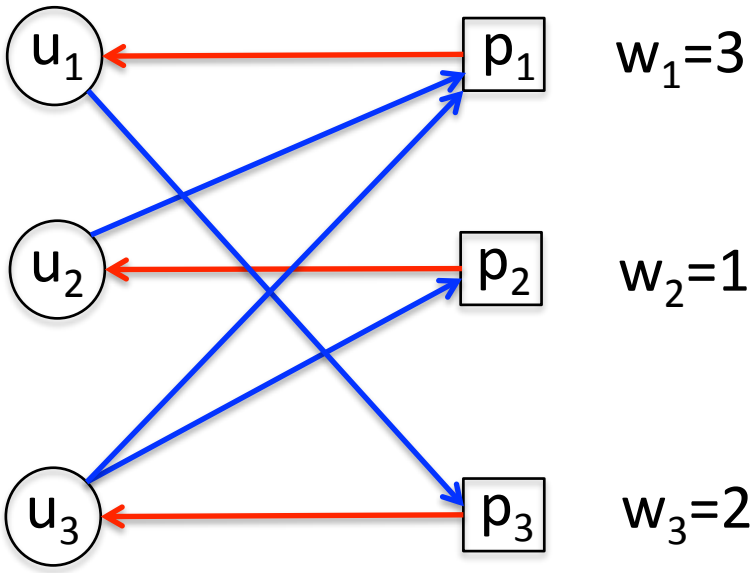

Figure 9.1: The bipartite digraph representation of a unicast index coding problem with 3 user vertices and 3 packet type vertices.

packet-to-user arcs and user-to-packet arcs leaves an acyclic subgraph. In this case, $\mathcal{P}_{\text{fd}}$ is called a *feedback packet vertex set*. A trivial feedback packet vertex set is $\mathcal{P}_{\text{fd}} = \mathcal{P}$ and the corresponding acyclic subgraph has no packet vertex. This trivial feedback packet vertex set has *weight* W, since the sum weight of all packet vertices is W. It is often possible to find a feedback packet vertex set with sum weight smaller than W. The feedback packet vertex set with the minimum sum weight is called the *minimum feedback packet vertex set*. The acyclic subgraph induced by the deletion of the minimum feedback packet vertex set is called the *maximum acyclic subgraph(MAS)*.

Assume that each transmission from the base station sends a number of bits equal to the number of bits in each of the fixed length packets. It is trivial to satisfy all demands with W transmissions, where each of the W packets is successively transmitted without coding. However, coding can often be used to reduce the number of transmissions. Let $T_{\min}(\mathcal{G})$ represent the minimum number of transmissions required to deliver all packets to their intended users for an index coding problem defined by the weighted bipartite digraph $\mathcal{G}$. The value $T_{\min}(\mathcal{G})$ considers all possible coding strategies. A theorem in [NTZ13] provides an information theoretic lower bound on $T_{\min}(\mathcal{G})$.

Theorem 9.1 (Theorem 1 and Lemma 1 in [NTZ13])**.** *Consider an index coding problem* $\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{A}, \mathcal{W}_{\mathcal{P}})$*. Let* $\mathcal{P}_{fd} \subseteq \mathcal{P}$ *be a feedback packet vertex set and let* $\mathcal{G}'$ *be the acyclic subgraph*

induced by the deletion of $\mathcal{P}_{fd}$*. If* $\sum_{p_m \in \mathcal{G}'} w_{p_m} = W'$*, then* $T_{min}(\mathcal{G}) \geq W'$*.*

While the above theorem holds for general (possibly groupcast) index coding problems, this chapter uses it in the unicast case. For unicast problems, Theorem 9.1 reduces to an earlier result on acyclic subgraphs in [BYBJK11] after a suitable transformation of the graph structure.

Suppose the largest cycle in digraph $\mathcal{G}$ involves L packet vertices. Define the set of all cycles in $\mathcal{G}$ as $\mathcal{C} = \bigcup_{i=1}^{L} \mathcal{C}_i$, where $\mathcal{C}_i, i = 2, \ldots, L$ is the set of all cycles involving i packet vertices. These cycles can possibly overlap, i.e., some of them can share common vertices. The tightest lower bound provided by Theorem 9.1 is referred to as the maximum acyclic subgraph (MAS) bound and can be formulated as a linear integer program (IP) as below:

Maximum Acyclic Subgraph IP (P1):

$$\begin{aligned} \max_{x_m} \quad & \sum_{m=1}^{M} x_m w_{p_m} \\ \text{s.t.} \quad & \sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in C_i\}} \leq i-1, \quad \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\ & x_m \in \{0,1\}, \quad m = 1, \ldots, M \end{aligned}$$

where $x_m \in \{0,1\}, m = 1, \ldots, M$ indicates if packet vertex p_m remains in the acyclic subgraph, objective function $\sum_{m=1}^{M} x_m w_{p_m}$ is the sum weight of the acyclic subgraph, $\mathbf{1}_{\{p_m \in C_i\}}$ is the indicator function which equals one if and only if packet vertex p_m participates in cycle C_i, and $\sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in C_i\}} \leq i-1$ is the constraint that for each cycle $C_i \in \mathcal{C}_i$, at most $i-1$ packet vertices remain in the acyclic subgraph. This problem finds the MAS bound formed by packet vertex deletion.

The integer constraints of the above problem can be convexified to form the following *linear*

programming (LP) relaxation:

$$\begin{aligned} & \text{Maximum Acyclic Subgraph LP (P1}'\text{):} \\ \max_{x_m} \quad & \sum_{m=1}^{M} x_m w_{p_m} \\ \text{s.t.} \quad & \sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in C_i\}} \leq i-1, \quad \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\ & 0 \leq x_m \leq 1, \quad m = 1, \ldots, M \end{aligned}$$

The only difference between problem (P1) and its relaxation (P1′) is that the constraints $x_m \in \{0, 1\}$ are changed to $0 \leq x_m \leq 1$. The relaxed problem (P1′) can be solved with standard linear programming techniques. The number of constraints depends on the number of cycles in the graph. However, the number of cycles in general graphs can grow exponentially with the number of vertices, and so (P1′) can be difficult to solve when the graph is large[2].

One might not expect the relaxed problem (P1′) to have a physical meaning. Remarkably, this chapter proves that it does. Indeed, the next section shows that any solution to the relaxed problem leads to a coding strategy. The clearance time of the coding strategy is equal to the optimal objective function value of the relaxed problem. Hence, this value is an *upper bound* on $T_{min}(\mathcal{G})$. This is surprising because the original integer program (P1) provides a *lower bound* on $T_{min}(\mathcal{G})$ and does not suggest *any* particular coding strategy.

Define val(P1) as the optimal objective function value of problem (P1), being the size of the maximum acyclic subgraph. Theorem 9.1 implies that $val(\text{P1}) \leq T_{\min}(\mathcal{G})$. The optimal objective function value for the relaxation (P1′) can be written as $val(\text{P1}') = val(\text{P1}) + gap(\text{P1}', \text{P1})$, where $gap(\text{P1}', \text{P1}) = val(\text{P1}') - val(\text{P1})$ is the *integrality gap* between the LP relaxation (P1′) and the integer program (P1). Since the relaxation (P1′) has less restrictive constraints, the value of $gap(\text{P1}', \text{P1})$ is always non-negative. The next section proves constructively that:

$$val(\text{P1}) \leq T_{\min}(\mathcal{G}) \leq val(\text{P1}) + gap(\text{P1}', \text{P1})$$

Thus, the difference between the minimum clearance time and the maximum acyclic subgraph

[2]In fact, a linear program with an exponential number of constraints can still be solved in polynomial time via the ellipsoid method as long as it has an efficient separation oracle [GLS93]. It can be shown that the maximum acyclic subgraph LP (P1′) has an efficient separation oracle and hence can be solved in polynomial time via the ellipsoid method.

bound is bounded by the integrality gap $gap(\mathrm{P1}', \mathrm{P1})$. Furthermore, Section 9.4 shows that $gap(\mathrm{P1}', \mathrm{P1}) = 0$ in special cases when the digraph $\mathcal{G}$ is planar.

9.3 Cyclic Codes and Linear Programming Duality

Inspired by the observation that the lower bound in Theorem 9.1 is closely connected with cycles in graph $\mathcal{G}$, this section considers *cyclic codes* that exploit cycles in $\mathcal{G}$. It is shown that the problem of finding the optimal cyclic code is the dual problem of the LP relaxation (P1′). Thus, the performance gap between the optimal cyclic code and the optimal index code is ultimately bounded by the integrality gap $gap(\mathrm{P1}', \mathrm{P1})$.

9.3.1 Cyclic Codes

Suppose there exists a cycle in $\mathcal{G}$ that involves K user vertices $\{u_1, u_2, \ldots, u_K\}$ and K packet vertices $\{p_1, p_2, \ldots, p_K\}$. In this cycle, user u_1 has p_K as side information and demands p_1, user u_2 has p_1 as side information and demands p_2, user u_3 has p_2 as side information and demands p_3, and so on. If the weight of each packet vertex is identically one, a *K-cycle coding action* can deliver all K packets by transmitting $Z_i = p_i + p_{i+1}, i = 1, \ldots, K-1$ with $K-1$ transmissions, where addition is the mod-2 summation of each bit in both packets. After transmissions, user $u_i \in \{u_2, \ldots, u_K\}$ can decode packet p_i by performing $p_{i-1} + Z_{i-1} = p_{i-1} + (p_{i-1} + p_i) = p_i$. At the same time, user u_1 can decode packet p_1 by performing:

$$\begin{aligned} & Z_1 + \ldots + Z_{K-1} + q_K \\ =& (p_1 + p_2) + (p_2 + p_3) + \ldots + (p_{K-1} + p_K) + p_K \\ =& p_1. \end{aligned}$$

A linear index code is said to be a cyclic code if it uses a sequence of coding actions that involve only cyclic coding actions and direct broadcasts without coding. Linear codes can be further categorized into *scalar linear codes* and *vector linear codes* according to whether the transmitted message is a linear combination of the original packets or the subpackets obtained by subdivisions. In scalar linear codes, each packet is considered as an element of a finite field and the transmitted message is a linear combination of packets over that field. In vector linear codes,

each packet is assumed to be sufficiently large and can be divided into many smaller subpackets and the transmitted message is a linear combination of these subpackets instead of the original packets. The problem of finding the optimal scalar cyclic code to clear $\mathcal{G}$ can be formulated as an IP as below:

$$\begin{aligned}
&\text{Cyclic Code IP (P2):}\\
\min_{y_{C_i}, y_m} \quad & \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i}(i-1) + \sum_{m=1}^{M} y_m \\
\text{s.t.} \quad & y_m + \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m \in C_i\}} \geq w_{p_m},\ m = 1, \ldots, M \\
& y_{C_i} \text{ non-negative integer}, \quad \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\
& y_m \text{ non-negative integer}, \quad m = 1, \ldots, M
\end{aligned}$$

where y_{C_i} is the number of cyclic coding actions over each cycle $C_i, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$, y_m is the number of direct broadcasts over each packet vertex $p_m, m = 1, \ldots, M$, objective function $\sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i}(i-1) + \sum_{m=1}^{M} y_m$ is the total number of transmissions, and $y_m + \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m \in C_i\}} \geq w_{p_m}$ is the constraint that all the w_{p_m} packets represented by packet vertex p_m are cleared by either cyclic codes or direct broadcasts.

The LP relaxation of the cyclic code IP (P2) is as below:

$$\begin{aligned}
&\text{Cyclic Code LP (P2}'\text{):}\\
\min_{y_{C_i}, y_m} & \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i}(i-1) + \sum_{m=1}^{M} y_m \\
\text{s.t.} \quad & y_m + \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m \in C_i\}} \geq w_{p_m},\ m = 1, \ldots, M \\
& y_{C_i} \geq 0, \quad \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\
& y_m \geq 0, \quad m = 1, \ldots, M
\end{aligned}$$

The only difference between the above problem and the cyclic code IP (P2) is that the constraints that y_{C_i} and y_m are non-negative integers are replaced by the relaxed constraints that $y_{C_i} \geq 0$ and $y_m \geq 0$. This gives rise to the optimal vector cyclic code. The optimal vector cyclic code can be viewed as a scheme for time-sharing of cyclic coding actions over overlapping cycles. With this interpretation, y_{C_i} is proportional to the fraction of time used for cyclic coding actions over

cycle C_i.

Since all the coefficients in the linear constraints of the cyclic code LP (P2′) are integers, an optimal solution can be found that has all variables equal to rational numbers. Let an optimal solution of cyclic code LP (P2′) be $y^*_{C_i}, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L; y^*_m, m = 1, \ldots, M$, and assume these values are all rational numbers. The optimal vector cyclic code can be constructed as follows. First, one can find an integer θ such that $\theta y^*_{C_i}, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L; \theta y^*_m, m = 1, \ldots, M$ are all integers. Next, divide each packet into θ subpackets. After the subdivision, a single cyclic coding action over a cycle C_i is no longer a linear combination of packets but a linear combination of subpackets. Further, a single (uncoded) direct broadcast from a packet vertex p_m is no longer the broadcast of one packet but one subpacket. Then, the optimal vector cyclic code performs $\theta y^*_{C_i}$ cyclic coding actions over each cycle $C_i, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$ and broadcasts θy^*_m subpackets over each packet vertex $p_m, m = 1, \ldots, M$. To apply the above vector cyclic code, the number of bits in each packet must be an integer multiple of θ. This is a reasonable assumption when the packet size is large. Indeed, if the original packet size is B, each packet can be expanded to have size $\tilde{B} = B + r_B$, where $\tilde{B}$ is the smallest multiple of θ that is greater than or equal to B, and $r_B \in \{0, 1, \ldots, \theta - 1\}$. The expansion ratio is $(B + r_B)/B$, which converges to 1 as $B \to \infty$.

Define $gap(\text{P2}, \text{P2}')$ as the integrality gap between the cyclic code IP (P2) and its LP relaxation (P2′). Since the relaxation (P2′) has less restrictive constraints, the value of $gap(\text{P2}, \text{P2}')$ is always non-negative. Let $val(\text{P2})$ and $val(\text{P2}')$ be the optimal objective function values for problems (P2) and (P2′), respectively. Thus, $val(\text{P2})$ and $val(\text{P2}')$ are the clearance times attained by the optimal scalar cyclic code and vector cyclic code, respectively, and:

$$val(\text{P2}) = val(\text{P2}') + gap(\text{P2}, \text{P2}') \tag{9.1}$$

9.3.2 Duality Between Information Theoretical Lower Bounds and Cyclic Codes

The duality between the maximum acyclic subgraph lower bound given by Theorem 9.1 and the optimal cyclic code is formally stated in the following lemma.

Lemma 9.1. *The maximum acyclic subgraph LP (P1′) and the cyclic code LP (P2′) form a primal-dual linear programming pair. In particular, the vector cyclic code associated with problem*

(P2′) achieves a clearance time val(P2′) that satisfies:

$$val(P2') = val(P1) + gap(P1', P1) \tag{9.2}$$

Proof. The Lagrangian function of the cyclic code LP (P2′) can be written as

$$\begin{aligned}
&L(y_{C_i}, y_m, \lambda_m, \mu_{C_i}, \mu_m) \\
=&\sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i}(i-1) + \sum_{m=1}^{M} y_m + \sum_{m=1}^{M} \lambda_m \big[w_{p_m} - y_m - \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m \in C_i\}} \big] \\
&- \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} \mu_{C_i} y_{C_i} - \sum_{m=1}^{M} \mu_m y_m \\
=&\sum_{m=1}^{M} \lambda_m w_{p_m} + \sum_{m=1}^{M} y_m [1 - \lambda_m - \mu_m] + \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \big[(i-1) - \sum_{m=1}^{M} \lambda_m \mathbf{1}_{\{p_m \in C_i\}} - \mu_{C_i} \big]
\end{aligned}$$

where $\lambda_m \geq 0, m = 1, \ldots, M$; $\mu_{C_i} \geq 0, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$ and $\mu_m \geq 0, m = 1, \ldots, M$. The dual problem of (P2′) is defined as:

$$\max_{\substack{\lambda_m \geq 0 \\ \mu_{C_i} \geq 0 \\ \mu_m \geq 0}} \min_{\substack{y_{C_i} \in \mathbb{R} \\ y_m \in \mathbb{R}}} L(y_{C_i}, y_m, \lambda_m, \mu_{C_i}, \mu_m)$$

Note that,

$$\min_{\substack{y_{C_i} \in \mathbb{R} \\ y_m \in \mathbb{R}}} L(y_{C_i}, y_m, \lambda_m, \mu_{C_i}, \mu_m) = \begin{cases} \sum_{m=1}^{M} \lambda_m w_{p_m} & \text{if } \begin{smallmatrix} (i-1) - \sum_{m=1}^{M} \lambda_m \mathbf{1}_{\{p_m \in C_i\}} - \mu_{C_i} = 0, \\ \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\ 1 - \lambda_m - \mu_m = 0, m = 1, \ldots, M \end{smallmatrix} \\ -\infty & \text{otherwise} \end{cases}$$

Then, the dual problem of (P2′) can be written as,

$$\begin{aligned}
\max_{\lambda_m,\mu_{C_i},\mu_m} \quad & \sum_{m=1}^{M} \lambda_m w_{p_m} \\
\text{s.t.} \quad & (i-1) - \sum_{m=1}^{M} \lambda_m \mathbf{1}_{\{p_m \in C_i\}} - \mu_{C_i} = 0, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\
& 1 - \lambda_m - \mu_m = 0, \quad m = 1, \ldots, M \\
& \lambda_m \geq 0, \quad m = 1, \ldots, M \\
& \mu_{C_i} \geq 0, \quad \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\
& \mu_m \geq 0, \quad m = 1, \ldots, M
\end{aligned}$$

Eliminating variables $\mu_{C_i}, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$ and $\mu_m, m = 1, \ldots, M$, we obtain

$$\begin{aligned}
\max_{\lambda_m} \quad & \sum_{m=1}^{M} \lambda_m w_{p_m} \\
\text{s.t.} \quad & \sum_{m=1}^{M} \lambda_m \mathbf{1}_{\{p_m \in C_i\}} \leq (i-1), \quad \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\
& 0 \leq \lambda_m \leq 1, \quad m = 1, \ldots, M
\end{aligned}$$

The above problem is the same as (P1′). Thus, the clearance time of the vector cyclic code associated with problem (P2′) is equal to the value of the optimal objective function in problem (P1′), which is $val(\text{P1}) + gap(\text{P1}', \text{P1})$. □

Thus far, we have proven the following lower and upper bound for the minimum clearance time of an index coding problem.

$$val(\text{P1}) \leq T_{\min}(\mathcal{G}) \leq val(\text{P1}) + gap(\text{P1}', \text{P1}) \tag{9.3}$$

where the first inequality follows from Theorem 9.1 and the second inequality follows from Lemma 9.1. Hence, the performance gap between the optimal index code and the optimal vector cyclic code is ultimately bounded by the integrality gap between the maximum acyclic subgraph IP (P1) and its LP relaxation (P1′).

There are various techniques for bounding the integrality gaps of integer linear programs, such as the random rounding methods in [RT87, Rag88]. Rather than explore this direction, the

next section provides a special case where the gap is equal to zero. This is motivated as follows. Adding the non-negative value $gap(\mathrm{P2}, \mathrm{P2}')$ to the right-hand-side of (9.3) gives:

$$\begin{aligned} val(\mathrm{P1}) \leq T_{\min}(\mathcal{G}) &\leq val(\mathrm{P1}) + gap(\mathrm{P1}', \mathrm{P1}) + gap(\mathrm{P2}, \mathrm{P2}') \\ &= val(\mathrm{P2}) \end{aligned}$$

where the final equality uses (9.1)-(9.2). In the special case when $val(\mathrm{P1}) = val(\mathrm{P2})$, one has $gap(\mathrm{P1}', \mathrm{P1}) = gap(\mathrm{P2}, \mathrm{P2}') = 0$ and $val(\mathrm{P2}) = T_{min}(\mathcal{G})$, so that the scalar cyclic code given by the cyclic code IP (P2) is an optimal index code.

9.4 Optimality of Cyclic Codes in Planar Bipartite Graphs

In graph theory, a *planar graph* is a graph that can be drawn as a picture on a 2-dimensional plane in a way so that no two arcs meet at a point other than a common vertex. The main result in this section is the following theorem:

Theorem 9.2. *If the bipartite digraph $\mathcal{G}$ for a (unicast) index coding problem is planar, then $val(P1) = val(P2)$, i.e., $gap(P1', P1) = 0$ and $gap(P2, P2') = 0$. Hence, the (scalar) cyclic code given by the cyclic code IP (P2) is an optimal index code.*

The proof of Theorem 9.2 relies on the cycle-packing and feedback arc set duality in arc-weighted planar graphs, which is summarized in the following theorem.

Theorem 9.3 (Theorem 2.1 in [GT11b] originally proven in [LY78])**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \mathcal{W}_{\mathcal{A}})$ be an arc-weighted planar digraph where $\mathcal{V}$ is the set of vertices, $\mathcal{A}$ is the set of arcs and $\mathcal{W}_{\mathcal{A}}$ is an integer arc weight assignment which assigns each arc $a \in \mathcal{A}$ a non-negative integer weight $w_a \in \mathbb{Z}^+$. Let $\mathcal{C}$ be the set of cycles in $\mathcal{G}$. Then we have*

$$\begin{aligned} &\min \Big\{ \sum_{a \in \mathcal{A}} x_a w_a : \sum_{a \in \mathcal{A}} x_a \mathbf{1}_{\{a \in C\}} \geq 1, \forall C \in \mathcal{C}; x_a \in \{0,1\}, \forall a \in \mathcal{A} \Big\} \\ =& \max \Big\{ \sum_{C \in \mathcal{C}} y_C : \sum_{C \in \mathcal{C}} y_C \mathbf{1}_{\{a \in C\}} \leq w_a, \forall a \in \mathcal{A}; y_C \in \mathbb{Z}^+, \forall C \in \mathcal{C} \Big\}. \end{aligned} \tag{9.4}$$

The integer program on the left-hand-side of (9.4) is a minimum feedback arc set problem, while the integer program on the right-hand-side of (9.4) is a cycle packing problem. Both

problems are associated with *arc-weighted digraphs*. However, our graph is *vertex-weighted* rather than *arc-weighted*. To apply this theorem, we modify the bipartite digraph $\mathcal{G}$ to produce an arc-weighted digraph $\mathcal{G}^s$, which is planar if and only if $\mathcal{G}$ is planar. We then show that the minimum feedback packet vertex set problem and the cycle packing problem in $\mathcal{G}$ can be reduced to the minimum feedback arc set problem and the cycle packing problem in $\mathcal{G}^s$, respectively. The following subsections develop the proof of Theorem 9.2 and provide some additional consequences.

9.4.1 Complementary Problems

The maximum acyclic subgraph IP (P1) finds the packet weighted maximum acyclic subgraph. This is equivalent to finding the minimum feedback packet vertex set. Indeed, this is the set of packets whose deletion induce the packet weighted maximum acyclic subgraph. Thus, an equivalent problem to the maximum acyclic subgraph IP (P1) is:

$$\text{(P3)} \quad \begin{aligned} \min_{x_m} \quad & \sum_{m=1}^{M} x_m w_{p_m} \\ \text{s.t.} \quad & \sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in C_i\}} \geq 1, \ \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L \\ & x_m \in \{0,1\}, \quad m = 1, \ldots, M \end{aligned}$$

where $x_m \in \{0,1\}, m = 1, \ldots, M$ indicates if packet vertex p_m is selected into the feedback packet vertex set, objective function $\sum_{m=1}^{M} x_m w_{p_m}$ is the sum weight of the feedback packet vertex set, $\mathbf{1}_{\{p_m \in \mathcal{C}_i\}}$ is the indicator function which equals one if and only if packet vertex p_m participates in cycle C_i, and $\sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in C_i\}} \geq 1$ is the constraint that at least one packet vertex in each cycle is selected into the feedback packet vertex set. If $x_m^*, m = 1, \ldots, M$ is the optimal solution of (P3) and attains the optimal value W_0, then $\overline{x}_m^* = 1 - x_m^*, m = 1, \ldots, M$ is the optimal solution of (P1) and attains the optimal value $W - W_0$.

Now consider the integer program related to cyclic coding. It is now useful to write the complementary problem to the cyclic code IP (P2). In [CASL11], Chaudhry et. al. introduced the concept of complementary index coding problems. Instead of trying to find the minimum number of transmissions to clear the problem, the complementary index coding problem is formulated to maximize the number of saved transmissions by exploiting a specific code structure. Recall

that any K-cycle code can deliver K packets in $K-1$ transmissions and hence one transmission is saved in each K-cycle code. If the weight of each packet is not identically one, then K-cycle coding actions can be performed $w_{\min} = \min\{w_{p_1}, \ldots, w_{p_K}\}$ times on the same cycle. By performing K-cycle coding actions w_{min} times and then directly broadcasting the remaining packets (uncoded), the base station can deliver $w_{\text{total}} = \sum_{k=1}^{K} w_{p_k}$ packets with $w_{\text{total}} - w_{\min}$ transmissions.Thus, $w_{\min}$ transmissions are saved.

The complementary index coding problem which aims to maximize the number of saved transmissions by exploiting scalar cycles in $\mathcal{G}$ is formulated as a linear integer program below:

$$
\text{(P4)} \quad
\begin{aligned}
\max_{y_{C_i}} \quad & \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \\
\text{s.t.} \quad & \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m \in C_i\}} \leq w_{p_m}, \ m = 1, \ldots, M \\
& y_{C_i} \text{ non-negative integer}, \ \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L
\end{aligned}
$$

where y_{C_i} is the number of cyclic coding actions over each cycle $C_i \in \mathcal{C}_i, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$, objective function $\sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i}$ is the total number of cyclic coding actions, i.e., total number of saved transmissions, and $\sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m \in C_i\}} \leq w_{p_m}$ is the constraint that each packet vertex p_m can participate in at most w_{p_m} cyclic coding actions. This is important because if packet vertex p_m has already participated w_{p_m} times in cyclic coding actions, then all of its packets have been delivered and new cyclic coding actions that involve this packet vertex can no longer save any transmissions. If the optimal solution of (P4) is $y^*_{C_i}, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$ and attains the optimal value W_0, then the optimal solution of the cyclic code IP (P2) is $\overline{y}^*_{C_i} = y^*_{C_i}, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L, \overline{y}^*_m = w_{p_m} - \sum_{i=2}^{L} \sum_{C_i \in \mathcal{C}_i} y^*_{C_i} \mathbf{1}_{\{p_m \in C_i\}}, m = 1, \ldots, M$ and attains the optimal value $W - W_0$.

9.4.2 Packet Split Digraph

Definition 9.1 (Packet Split Digraph)**.** *Given a graph $\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{A}, \mathcal{W}_{\mathcal{P}})$, we construct the corresponding packet split digraph $\mathcal{G}^s = (\mathcal{V}^s, \mathcal{A}^s, \mathcal{W}^s)$ as follows:*

1. *For each packet vertex $p_m \in \mathcal{P}, m = 1, \ldots, M$, we create two packet vertices p_m^{in} and p_m^{out}. Let $\mathcal{V}^s = \mathcal{U} \cup \{p_1^{in}, p_1^{out}, p_2^{in}, p_2^{out}, \ldots, p_M^{in}, p_M^{out}\}$.*

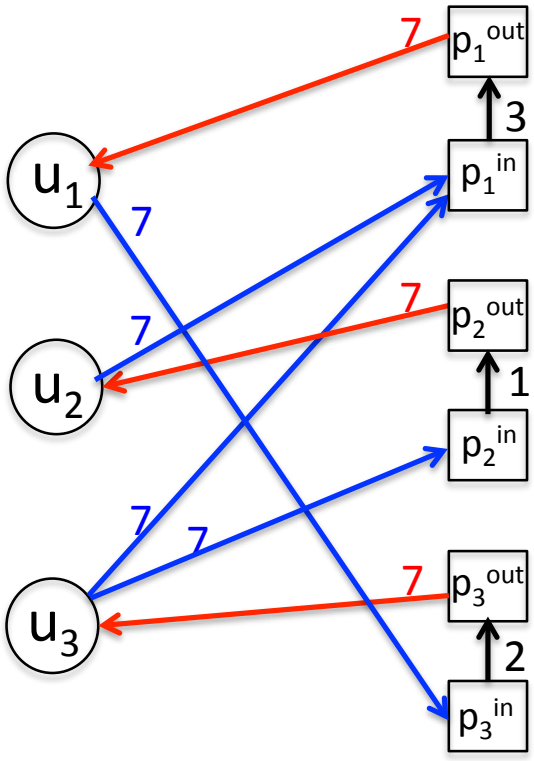

Figure 9.2: The packet split digraph constructed from the bipartite digraph given in Figure 9.1

2. *For each packet vertex* $p_m \in \mathcal{P}, m = 1, \ldots, M$*, we create a packet-to-packet arc* (p_m^{in}, p_m^{out}) *in* $\mathcal{A}^s$*. For each arc* $(u_n, p_m) \in \mathcal{A}$*, we create a user-to-packet arc* (u_n, p_m^{in}) *in* $\mathcal{A}^s$*. For each arc* $(p_m, u_n) \in \mathcal{A}$*, we create a packet-to-user arc* (p_m^{out}, u_n) *in* $\mathcal{A}^s$*.*

3. *For each arc* (p_m^{in}, p_m^{out}) *in* $\mathcal{A}^s$*, we assign a weight which is equal to* $w_{p_m} \in \mathcal{W}_{\mathcal{P}}$*. For each arc* (u_n, p_m^{in}) *or* (p_m^{out}, u_n) *in* $\mathcal{A}^s$*, we assign an integer weight which is larger than* $\sum_{m=1}^{M} w_{p_m}$*.*

For any bipartite digraph $\mathcal{G}$, the packet split digraph $\mathcal{G}^s$, which is an arc-weighted digraph, can always be constructed. Figure 9.2 shows the packet split digraph constructed from the bipartite digraph in Figure 9.1. In any digraph, a set of arcs is called a feedback arc set if the removal of arcs in this set leaves an acyclic digraph. If the digraph is arc-weighted, the feedback arc set with the minimum sum weight is called the minimum feedback arc set.

The following facts summarize the connections between the packet split digraph and the original digraph.

Fact 9.1. *There is a bijection between* $\mathcal{G}$ *and* $\mathcal{G}^s$*. This bijection maps user vertices, user-to-packet arcs, packet vertices, and packet-to-user arcs in* $\mathcal{G}$ *to user vertices, user-to-packet arcs,*

packet-to-packet arcs, and packet-to-user arcs in $\mathcal{G}^s$, respectively. Thus, this bijection also maps cycles in $\mathcal{G}$ to cycles in $\mathcal{G}^s$.

Proof. The bijection can be easily identified according to the construction rule of the packet split digraph. □

Fact 9.2. *Every minimum feedback arc set of packet split graph $\mathcal{G}^s$ contains only packet-to-packet arcs and no packet-to-user arcs or user-to-packet arcs.*

Proof. In digraph $\mathcal{G}$, each cycle contains at least one packet vertex. By Fact 1, each cycle $\mathcal{G}^s$ contains at least one packet-to-packet arc. As such, the arc set composed of all packet-to-packet arcs is a feedback arc set of $\mathcal{G}^s$ and this feedback arc set contains no packet-to-user arcs or user-to-packet arcs. Note that the sum weight of this arc set is strictly less than the weight of any single packet-to-user or user-to-packet arc. Any feedback arc set with a packet-to-user arc or user-to-packet arc has a sum weight strictly larger than that of this one and hence cannot be a minimum feedback arc set. □

Fact 9.3. *If $\mathcal{A}^s_{fd} \subseteq \mathcal{A}^s$ is a minimum feedback arc set of the packet split digraph $\mathcal{G}^s$, then a minimum feedback packet vertex set $\mathcal{P}_{fd} \subseteq \mathcal{P}$ of $\mathcal{G}$ is immediate. In addition, the sum weight of $\mathcal{P}_{fd}$ is equal to the sum weight of $\mathcal{A}^s_{fd}$.*

Proof. Let $\mathcal{A}^{\mathrm{s}}_{\mathrm{fd}}$ be a minimum feedback arc set of $\mathcal{G}^s$ and the sum weight of $\mathcal{A}^{\mathrm{s}}_{\mathrm{fd}}$ be W_{fd}. By Fact 2, $\mathcal{A}^{\mathrm{s}}_{\mathrm{fd}}$ contains only packet-to-packet arcs. By Fact 1, the packet vertex set $\mathcal{P}_{\mathrm{fd}} \subseteq \mathcal{P}$ composed by packet vertices corresponding to arcs in $\mathcal{A}^{\mathrm{s}}_{\mathrm{fd}}$ is a feedback packet vertex set of $\mathcal{G}$ and the sum weight of $\mathcal{P}_{\mathrm{fd}}$ is equal to W_{fd}. If $\mathcal{P}_{\mathrm{fd}}$ is not a minimum feedback packet vertex set, there must exist a minimum feedback packet vertex set, say $\mathcal{P}'_{\mathrm{fd}}$, whose sum weight $W'_{\mathrm{fd}} < W_{\mathrm{fd}}$. By Fact 1, the counterpart of $\mathcal{P}'_{\mathrm{fd}}$ in $\mathcal{G}^s$ is a feedback arc set and the sum weight of this feedback arc set is equal to W'_{fd}. Denote this feedback arc set as $\mathcal{A}^{s,\prime}_{\mathrm{fd}}$, then $\mathcal{A}^{s,\prime}_{\mathrm{fd}}$ has a sum weight strictly less than W_{fd}. This contradicts the fact that $\mathcal{A}^{\mathrm{s}}_{\mathrm{fd}}$ is a minimum feedback arc set of $\mathcal{G}^s$. Hence, $\mathcal{P}_{\mathrm{fd}}$ must be a minimum feedback packet set of $\mathcal{G}$. □

9.4.3 Optimality of Cyclic Codes in Planar Graphs

The planarity of a digraph is not affected by arc directions, so that a digraph is planar if and only if its undirected counterpart, where all directed arcs are turned into undirected edges,

is planar. The following definitions are useful in characterizing the planarity of an (undirected) graph.

Definition 9.2 (Page 21 in [Bol98]). *Given an edge $e = (v_1, v_2)$ of a graph $\mathcal{G}$, subdividing the edge e is the operation of replacing the edge $e = (v_1, v_2)$ by the path (v_1, v_0, v_2) of length 2 (see Figure 9.3a).*

Definition 9.3 (Page 24 in [Bol98]). *Given an edge $e = (v_1, v_2)$ of a graph $\mathcal{G}$, contracting the edge e is the operation of merging the vertices v_1 and v_2 and deleting all resulting loops and duplicate edges (see Figure 9.3b).*

Definition 9.4 (Page 24 in [Bol98]). *A graph $\mathcal{H}$ is a minor of a graph $\mathcal{G}$ if $\mathcal{H}$ is a subgraph of a graph obtained from $\mathcal{G}$ by a sequence of edge contractions.*

Note that if a graph $\mathcal{G}$ is planar, edge subdivisions and contractions preserve the planarity. Two simplest non-planar graphs are the complete graph with 5 vertices, which is denoted as K_5, and the complete bipartite graph with 3 vertices on one side and 3 vertices on the other side, which is denoted as $K_{3,3}$. Both of them are drawn in Figure 9.4. The following theorem provides a sufficient and necessary condition for the planarity of an undirected graph.

Theorem 9.4 (Page 24 in [Bol98]). *A graph $\mathcal{G}$ is planar if and only if $\mathcal{G}$ contains neither K_5 nor $K_{3,3}$ as a minor.*

In the index coding problem, a packet is said to be a *uniprior packet* [OH12] if it is contained as side information by only one user. The following lemma is proposed to characterize the planarity of the packet split graph $\mathcal{G}^s$.

Lemma 9.2. *Let $\mathcal{G}$ be an index coding problem where each packet vertex is either unicast or uniprior and let $\mathcal{G}^s$ be the packet split digraph of $\mathcal{G}$. $\mathcal{G}^s$ is planar if and only if $\mathcal{G}$ is planar.*

Proof.

- $\mathcal{G}^s$ planar $\Rightarrow$ $\mathcal{G}$ planar: This part is relatively easy. Assume $\mathcal{G}^{\mathrm{s}}$ is planar and is drawn in a plane. A planar drawing of $\mathcal{G}$ can be obtained by contracting all the packet-to-packet arcs of $\mathcal{G}^{\mathrm{s}}$ into packet vertices. This part holds for any $\mathcal{G}$ even if some packet vertex is neither unicast nor uniprior.

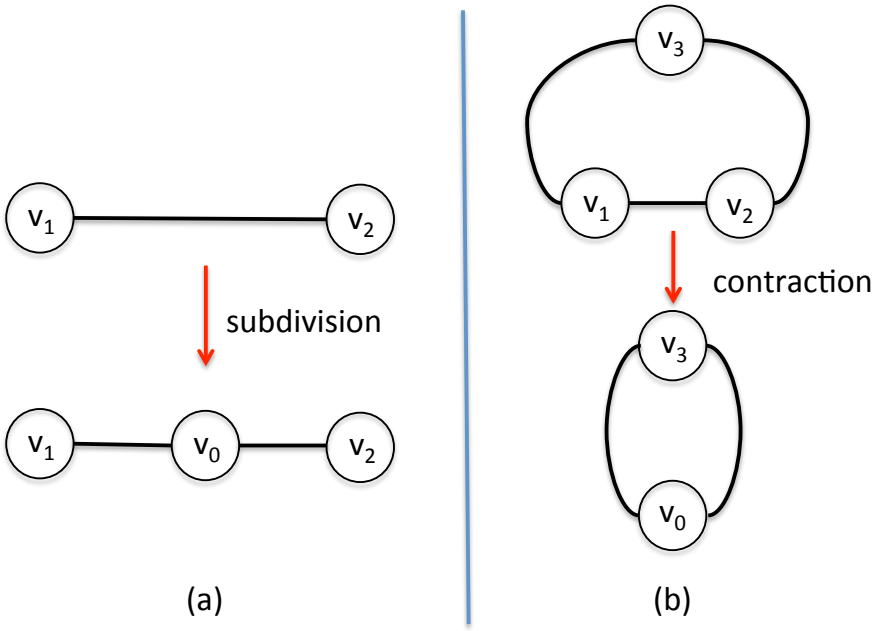

Figure 9.3: (a) Subdivision of edge (v_1, v_2). (b) Contraction of edge (v_1, v_2).

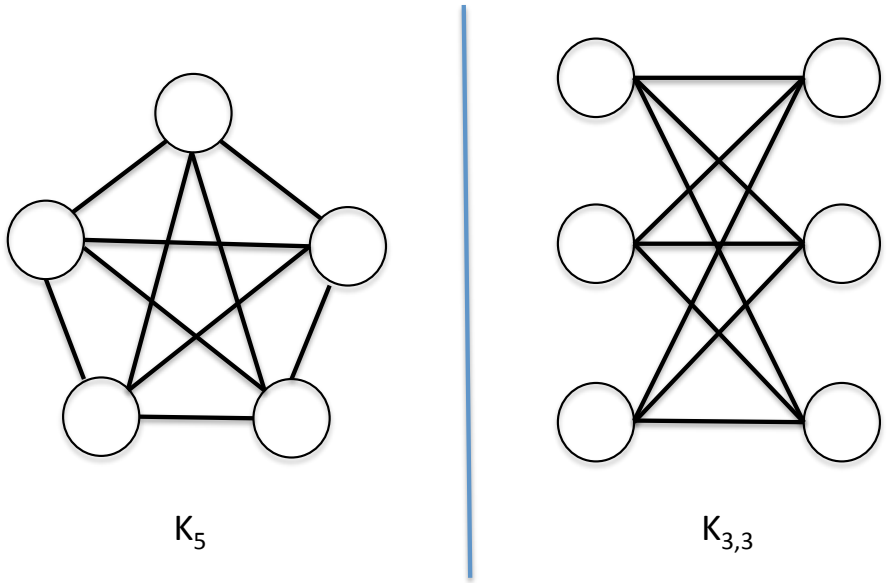

Figure 9.4: K_5 and $K_{3,3}$

- $\mathcal{G}$ planar $\Rightarrow$ $\mathcal{G}^s$ planar: Assume $\mathcal{G}$ is planar and is drawn in a plane. A planar drawing of $\mathcal{G}^{\rm s}$ can be obtained by subdividing packet-to-user arcs and user-to-packet arcs in $\mathcal{G}$. A crucial property is that each packet vertex in $\mathcal{G}$ has either one outgoing arc (unicast) or one incoming arc (uniprior). For each packet vertex p_m with only one outgoing arc (unicast), we can subdivide the outgoing arc into two parts; add a new vertex $p_m^{\rm out}$ in the middle and reindex the vertex p_m as $p_m^{\rm in}$. This preserves planarity, and the newly created vertex $p_m^{\rm out}$ indeed acts as the corresponding outgoing vertex for packet vertex p_m in the desired packet split digraph $\mathcal{G}^s$ (since that packet vertex has only one outgoing arc). The remaining packet vertices p_m that have not participated in these subdivisions must be uniprior and hence have just one *incoming* arc. We can subdivide this incoming arc into two parts; add a new vertex $p_m^{\rm in}$ in the middle and reindex the vertex p_m as $p_m^{\rm out}$. The subdivision operations as above yield a planar drawing of $\mathcal{G}^{\rm s}$.

□

Corollary 9.1. *For any unicast index coding problem $\mathcal{G}$, $\mathcal{G}^s$ is planar if and only if $\mathcal{G}$ is planar.*

Now we are ready to present the main result in this section.

Theorem 2: (Restated) If the bipartite digraph $\mathcal{G}$ for a (unicast) index coding problem is planar, then $val(\text{P1}) = val(\text{P2})$, i.e., $gap(\text{P1}', \text{P1}) = 0$ and $gap(\text{P2}, \text{P2}') = 0$. Hence, the cyclic code given by (P2) is an optimal index code.

Proof. Since $\mathcal{G}$ is a planar graph and this is a unicast index coding problem, $\mathcal{G}^s$ is also a planar graph by Corollary 9.1. Let $\mathcal{G}^s = (\mathcal{V}^{\rm s}, \mathcal{A}^{\rm s}, \mathcal{W}^{\rm s})$ be the packet spit digraph of $\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{A}, \mathcal{W}_{\mathcal{P}})$. Let $\mathcal{C}^s$ be the set of cycles in $\mathcal{G}^s$. The minimum feedback arc set problem in $\mathcal{G}^s$ can be formulated as a linear IP as follows:

$$
\begin{aligned}
& \min_{x_a} && \sum_{a \in \mathcal{A}}^{M} x_a w_a \\
(\text{P3}^*) \quad & \text{s.t.} && \sum_{a \in \mathcal{A}} x_a \mathbf{1}_{\{a \in C\}} \geq 1, \quad \forall C \in \mathcal{C}^s \\
& && x_a \in \{0,1\}, \quad a \in \mathcal{A}
\end{aligned}
$$

Similarly, the cycle-packing problem in $\mathcal{G}^s$ can formulated as another linear IP as follows:

$$\begin{aligned} & \max_{y_C} && \sum_{C\in\mathcal{C}^s} y_C \\ (\text{P4}^*) \quad & \text{s.t.} && \sum_{C\in\mathcal{C}^s}^{L} y_C \mathbf{1}_{\{a\in C\}} \le w_a, \quad \forall a \in \mathcal{A}^s \\ & && y_C \text{ non-negative integer}, \forall C \in \mathcal{C}^s \end{aligned}$$

By Theorem 9.3, if $\mathcal{G}^s$ is a planar graph, then (P3*) and (P4*) have the same optimal value. In what follows, we show that the optimal value of (P3) is equal to that of (P3*) and the optimal value of (P4) is equal to that of (P4*).

- **(P3) and (P3*) have the same optimal value:** By Fact 3, the minimum feedback arc set corresponding to the solution of (P3*) can be converted to a minimum feedback packet set solution of (P3) which attains the same optimal objective function value as that of (P3*). On the other hand, by Fact 1, the optimal solution of (P3) can be converted to a solution of (P3*) which attains the same objective value as that of (P3).

- **(P4) and (P4*) have the same optimal value:** By Fact 1, there is a bijection from $\mathcal{C}$ to $\mathcal{C}^s$. This is equivalent to say, there is a bijection from variables in (P4) to those in (P4*). Let $\mathcal{A}_1^s$ be the set of packet-to-packet arcs and $\mathcal{A}_2^s$ be the set of packet-to-user and user-to-packet arcs. So $\mathcal{A}_1^s \cup \mathcal{A}_2^s = \mathcal{A}^s$ and $\mathcal{A}_1^s \cap \mathcal{A}_2^s = \emptyset$. The constraints $\sum_{C\in\mathcal{C}^s} y_C \mathbf{1}_{\{a\in C\}} \le w_a, \forall a \in \mathcal{A}_1^s$ in (P4*) are essentially the same as the constraints $\sum_{i=2}^{L} \sum_{C_i\in\mathcal{C}_i} y_{C_i} \mathbf{1}_{\{p_m\in C_i\}} \le w_{p_m}, m = 1,\ldots,M$ in (P4). The other inequality constraints $\sum_{C\in\mathcal{C}^s} y_C \mathbf{1}_{\{a\in C\}} \le w_a$ over $a \in \mathcal{A}_2^s$ can be shown to be redundant as follows. Let $y_C, C \in \mathcal{C}^s$ be an arbitrary non-negative integer vector which satisfies all the constraints $\sum_{C\in\mathcal{C}^s} y_C \mathbf{1}_{\{a\in C\}} \le w_a$ over $a \in \mathcal{A}_1^s$. Due to the bipartite property, each cycle in $\mathcal{G}$ contains at least one packet vertex. By Fact 1, each cycle in $\mathcal{G}^s$ contains at least one packet-to-packet arc. Thus, for any $C \in \mathcal{C}^s$, there exists

some $a \in \mathcal{A}_1^s$ such that $\mathbf{1}_{\{a \in \mathcal{C}\}} = 1$. Then, for any $\bar{a} \in \mathcal{A}_2^s$ we have,

$$\begin{aligned}
\sum_{C \in \mathcal{C}^s} y_C \mathbf{1}_{\{\bar{a} \in C\}} &\leq \sum_{C \in \mathcal{C}^s} y_C \\
&\leq \sum_{C \in \mathcal{C}^s} \big[y_C \cdot \sum_{a \in \mathcal{A}_1^s} \mathbf{1}_{\{a \in \mathcal{C}\}} \big] \\
&= \sum_{a \in \mathcal{A}_1^s} \big[\sum_{C \in \mathcal{C}^s} y_C \mathbf{1}_{\{a \in \mathcal{C}\}} \big] \\
&\leq \sum_{a \in \mathcal{A}_1^s} w_a \\
&< w_{\bar{a}}
\end{aligned}$$

where the first inequality follows from the fact that $0 \leq \mathbf{1}_{\{\bar{a} \in C\}} \leq 1$; the second inequality follows from the fact that for any $C \in \mathcal{C}^s$ there exists some $a \in \mathcal{A}_1^s$ such that $\mathbf{1}_{\{a \in \mathcal{C}\}} = 1$; the third inequality follows from the fact that all the constraints $\sum_{C \in \mathcal{C}^s} y_C \mathbf{1}_{\{a \in C\}} \leq w_a$ over $a \in \mathcal{A}_1^s$ are satisfied; and the last inequality follows from the fact that the weight of any packet-to-user arc or user-to-packet-arc is strictly larger than the sum weight of all packet-to-packet arcs. This is to say the constraint $\sum_{C \in \mathcal{C}^s} y_C \mathbf{1}_{\{a \in C\}} \leq w_a$ over any $a \in \mathcal{A}_2^s$ is automatically satisfied and hence redundant. Hence, (P4) and (P4*) are two equivalent optimization problems.

Combining the above facts, we can conclude that the optimal value of (P3) is equal to that of (P4). Denote this value as W_0. According to Theorem 9.1, $W - W_0$ is a lower bound on the clearance time of the index coding problem $\mathcal{G}$. On the other hand, $W - W_0$ is the clearance time achieved by the scalar cyclic code corresponding to the solution of (P4), or equivalently the cyclic code IP (P2). Hence, we can conclude that the cyclic code given by the cyclic code IP (P2) is the optimal index code. □

9.4.4 Optimality of Cyclic Codes in the Unicast-Uniprior Index Coding Problem

In this subsection, we consider the unicast-uniprior index coding problem where each packet is demanded by one single user and can be contained as side information by at most one single user. This problem is motivated by the broadcast relay problem [NTZ13] where multiple users

exchange their individual data through a broadcast relay.

A strong corollary of Theorem 9.2 on the unicast-uniprior index coding problem is presented as below. This corollary is also an enhancement of the conclusion in Section III.C of [NTZ13] where the cyclic code is proven to be the optimal index code in the unicast-uniprior index coding problem with less than or equal to 3 users.

Corollary 9.2. *If the number of users in the unicast-uniprior index coding problem is less than or equal to* 4*, then cyclic codes are optimal.*

Proof. Let $\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{A}, \mathcal{W}_{\mathcal{P}})$ be a unicast-uniprior index coding problem where each packet vertex has one single outgoing link and one single incoming link and $|\mathcal{U}| \leq 4$. Let the underlying undirected graph of $\mathcal{G}$ be $U(\mathcal{G})$. The degree of a vertex in an undirected graph is defined as the number of its adjacent edges. At most 4 vertices in $U(\mathcal{G})$ can have a degree larger than 2. That is because each packet vertex must have a degree of 2 and only a user vertex can have a degree larger than 2.

By Theorem 9.4, if $U(\mathcal{G})$ is nonplanar, there must exist a subgraph of $U(\mathcal{G})$ which can be converted to either K_5 or $K_{3,3}$ after several contracting operations. Note that K_5 has 5 nodes with identical degree of 4 and $K_{3,3}$ has 6 nodes with identical degree of 3. Also note that no matter a user-to-packet edge or a packet-to-user edge in $U(\mathcal{G})$ is contracted, one user vertex and one packet vertex are replaced by one new vertex whose degree is equal to the degree of the user vertex. As a result, contracting operations performed over $U(\mathcal{G})$ can not generate new nodes with degree larger than 2. Thus, there doesn't exist a subgraph of $U(\mathcal{G})$ which has K_5 or $K_{3,3}$ as minor. So graph $U(\mathcal{G})$ must be planar. By Theorem 9.2, the cyclic code is optimal in $\mathcal{G}$. □

9.5 Partial Clique Codes: a Duality Perspective

Section 9.3 shows the inherent duality between the maximum acyclic subgraph bound given by Theorem 9.1 and the optimal cyclic code. In fact, this is not an isolated case. In this section, a different code structure involving *partial clique codes* is considered. Partial clique codes are more sophisticated but often lead to performance improvements over cyclic codes. It is shown that the problem of finding the optimal partial clique code is the dual problem of another LP relaxation of the maximum acyclic subgraph IP (P1). The new relaxation is different from (P1′) and results in a smaller integrality gap.

9.5.1 Partial Clique Codes

Let $\mathcal{P}_0 \subseteq \mathcal{P}$ be a subset of k ($1 \leq k \leq M$) packet vertices and $\mathcal{N}_{\text{out}}(\mathcal{P}_0) = \bigcup_{p \in \mathcal{P}_0} \mathcal{N}_{\text{out}}(p)$ be the outgoing neighborhood of $\mathcal{P}_0$, i.e., the subset of users who demand packets in $\mathcal{P}_0$. If each user in $\mathcal{N}_{\text{out}}(P_0)$ has at least d ($0 \leq d \leq k-1$) packet vertices in $\mathcal{P}_0$ as side information, and at least one such user has exactly d, then the subgraph of $\mathcal{G}$ induced by $\mathcal{P}_0$ and $\mathcal{N}_{\text{out}}(\mathcal{P}_0)$ is called a (k,d)-*partial clique*. A (k,d)-partial clique where the weight of each packet vertex is identically 1 can be cleared with $k-d$ transmissions using $k-d$ independent linear combinations of the packets (such as using systematic maximum distance separable (MDS) codes in [BK98, TDN12] or random codes in [HMK$^+$06]). For example, the digraph $\mathcal{G}$ in Figure 9.1 itself is a $(3,1)$-partial clique. If the weight of each packet vertex is identically one, then this graph can be cleared by transmitting 2 linear combinations in the form $Z_1 = \alpha_1 p_1 + \alpha_2 p_2 + \alpha_3 p_3$ and $Z_2 = \beta_1 p_1 + \beta_2 p_2 + \beta_3 p_3$, where the α_i and β_i values are taken from a finite field $\mathbb{F}$. If the finite field $\mathbb{F}$ is large enough, we are able to find 2 linear combinations that, together with any one known value of p_1, p_2 or p_3, are linearly independent. Thus, each user $u_i, i = 1,2,3$ can decode p_i by solving a system of 2 linear equations and 2 unknowns.

The linear index code of $\mathcal{G}$ is said to be a partial clique code if it uses a sequence of coding actions that involve only partial clique coding actions. Note that the subgraph induced by a single packet vertex and the user vertex demanding it is by definition a $(1,0)$-partial clique. Let $\mathcal{T}_{k,d}, k = 1,\ldots,M, d = 0,\ldots,k-1$ be the set of all (k,d)-partial cliques in $\mathcal{G}$. The problem of finding the optimal scalar partial clique code can be formulated as an IP as below:

Partial Clique Code IP (P5):

$$\min_{y_{T_{k,d}}} \sum_{k=1}^{M} \sum_{d=0}^{k-1} \sum_{T_{k,d} \in \mathcal{T}_{k,d}} y_{T_{k,d}}(k-d)$$

$$\text{s.t.} \sum_{k=1}^{M} \sum_{d=0}^{k-1} \sum_{T_{k,d} \in \mathcal{T}_{k,d}} y_{T_{k,d}} \mathbf{1}_{\{p_m \in T_{k,d}\}} \geq w_{p_m},\ m = 1,\ldots,M$$

$$y_{T_{k,d}} \text{ non-negative integer}, \quad \substack{\forall T_{k,d} \in \mathcal{T}_{k,d}, \\ k=1,\ldots,M, d=0,\ldots,k-1}$$

where $y_{T_{k,d}}$ is the number of partial clique coding actions over each partial clique $T_{k,d}, \forall T_{k,d} \in \mathcal{T}_{k,d}, k = 1,\ldots,M, d = 0,\ldots,k-1$, objective function $\sum_{k=1}^{M} \sum_{d=0}^{k-1} \sum_{T_{k,d} \in \mathcal{T}_{k,d}} y_{T_{k,d}}(k-d)$ is the total

number of transmissions, and $\sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} y_{T_{k,d}}\mathbf{1}_{\{p_m\in T_{k,d}\}} \geq w_{p_m}$ is the constraint that all the w_{p_m} packets represented by packet vertex p_m are cleared by partial cliques involving it.

The problem of finding the optimal vector partial clique code can be formulated as a LP problem as below:

Partial Clique Code LP (P5′):

$$\begin{aligned}
\min_{y_{T_{k,d}}} \quad & \sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} y_{T_{k,d}}(k-d) \\
\text{s.t.} \quad & \sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} y_{T_{k,d}}\mathbf{1}_{\{p_m\in T_{k,d}\}} \geq w_{p_m}, m=1,\ldots,M \\
& y_{T_{k,d}} \geq 0, \qquad \substack{\forall T_{k,d}\in\mathcal{T}_{k,d},\\ k=1,\ldots,M, d=0,\ldots,k-1}
\end{aligned}$$

Similar to cyclic codes, the partial clique code LP (P5′) is the LP relaxation of the partial clique code IP (P5).

The structure of partial clique codes is much more sophisticated than that of cyclic codes. Typically, partial clique codes have to be implemented over a large enough finite field while cyclic codes can always be implemented over the binary field. On the other hand, the performance of partial clique codes in general is better (no worse) than that of cyclic codes. This is summarized in the following lemma.

Lemma 9.3. *In any (unicast) index coding problem, the optimal clearance time attained by scalar cyclic codes is no less than that attained by scalar partial clique codes. Similarly, the optimal clearance time attained by vector cyclic codes is no less than that attained by vector partial clique codes.*

Proof. This lemma is proven for scalar codes. However, all the arguments can be carried over to vector codes after each packet is divided into subpackets. Recall that in any K-cycle, each user vertex has at least one packet vertex as side information. So each K-cycle code can be equivalently replaced by a $(K,1)$-partial clique code. This uses partial clique coding to achieve the same clearance time. Thus, the best partial clique coding strategy achieves a clearance time that is less than or equal to that of the best cyclic coding strategy. □

Figure 9.5 shows an example of the index coding problem with 3 users and 3 packets. The bipartite digraph of this problem is not planar. (In fact, this example is the only unicast index

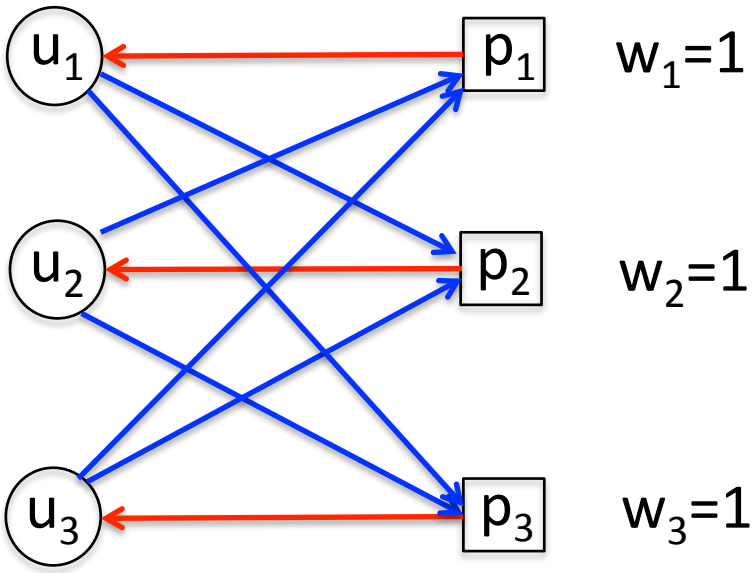

Figure 9.5: An example with 3 users and 3 packets where the partial clique code is strictly better than the cyclic code.

coding problem with 3 users and 3 packets for which the bipartite digraph is non-planar.) It can be verified that the optimal scalar cyclic code can clear this problem with 2 transmissions. On the other hand, the bipartite digraph itself is a $(3, 2)$-partial clique and hence the scalar partial clique code can clear it with one single transmission. The scalar partial clique code simply transmits $Z = p_1 + p_2 + p_3$. In this simple example, the scalar partial clique code is strictly better than the scalar cyclic code. However, the following theorem shows that partial clique codes have no performance advantage over cyclic codes in the unicast-uniprior index coding problem.

Theorem 9.5. *In any unicast-uniprior index coding problem, the optimal clearance time attained by scalar cyclic codes is equal to that attained by scalar partial cliques. Similarly, the optimal clearance time attained by vector cyclic codes is equal to that attained by vector partial cliques.*

Proof. This theorem is proven for scalar codes. However, all the arguments can be carried over to vector codes after each packet is divided into subpackets.

- Claim 1: The optimal clearance time attained by cyclic codes is larger than or equal to that attained by partial clique codes. This is Lemma 9.3.
- Claim 2: The optimal clearance time attained by cyclic codes is less than or equal to that attained by partial clique codes. For any partial clique $T_{k,d}$ $(d \geq 1)$ utilized in the optimal

partial clique code, k packets are cleared with $k-d$ transmissions. By definition of partial cliques, each user vertex in this $T_{k,d}$ has at least d arcs outgoing to packet vertices in it. So we are able to find a cycle in it. To find a cycle, we start at any vertex, traverse a path from vertex to vertex using any outgoing link and discover a cycle when we revisit a vertex. Denote this cycle as C_1 and delete all the packet vertices and the associated outgoing and incoming arcs from $T_{k,d}$. Note that each packet vertex has at most one outgoing arc and at most one incoming arc in a unicast-uniprior index coding problem. Hence, no two packet vertices in C_1 share the same outgoing neighbor or incoming neighbor. So after the deletion of the packet vertices and the associated outgoing and incoming arcs, the number of outgoing arcs of the user vertices involved in C_1 decreases by one while the number of outgoing arcs of the user vertices not involved in C_1 does not change. So in the remaining part of this $T_{k,d}$, each user vertex has at least $d-1$ outgoing arcs. Repeat the above process again and again. In the end, we have d cycles and no two cycles share the same packet vertex. So by performing a cycle code over each cycle $C_i, i = 1, \ldots, d$, we can save d transmissions in total. Hence, this $T_{k,d}$ can be cleared with $k-d$ transmissions by applying cyclic codes. As a result, cyclic codes are no worse than partial clique codes in the unicast-uniprior index coding problem.

□

9.5.2 Duality Between Information Theoretical Lower Bounds and Partial Clique Codes

Define an IP as below:

$$\text{IP (P6)} \quad \begin{aligned} \max_{x_m} \ & \sum_{m=1}^{M} x_m w_{p_m} \\ \text{s.t.} \ & \sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in T_{k,d}\}} \leq k-d, \ \substack{\forall T_{k,d} \in \mathcal{T}_{k,d}, \\ k=1,\ldots,M, d=0,\ldots,k-1} \\ & x_m \in \{0,1\}, \quad m = 1, \ldots, M \end{aligned}$$

The physical meaning of IP (P6) is to find the maximum packet weighted subgraph of $\mathcal{G}$ formed by packet vertex deletions such that at least d packet vertices are deleted in each (k,d) partial clique.

Lemma 9.4. *The partial clique code LP (P5′) and the LP relaxation of IP (P6) are a primal-dual linear programming pair.*

Proof. The Lagrangian function of the partial clique code LP (P5′) can be written as

$$\begin{aligned}
L(y_{T_{k,d}}, \lambda_m, \mu_{T_{k,d}}) =& \sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} y_{T_{k,d}}(k-d) + \sum_{m=1}^{M}\lambda_m\big[w_{p_m} - \sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} y_{T_{k,d}}\mathbf{1}_{\{p_m\in T_{k,d}\}}\big] \\
& - \sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} \mu_{T_{k,d}} y_{T_{k,d}} \\
=& \sum_{k=1}^{M}\sum_{d=0}^{k-1}\sum_{T_{k,d}\in\mathcal{T}_{k,d}} y_{T_{k,d}}\big[(k-d) - \sum_{m=1}^{M}\lambda_m\mathbf{1}_{\{p_m\in T_{k,d}\}} - \mu_{T_{k,d}}\big] + \sum_{m=1}^{M}\lambda_m w_{p_m}
\end{aligned}$$

where $\lambda_m \geq 0, m = 1, \ldots, M$ and $\mu_{T_{k,d}} \geq 0, \forall T_{k,d} \in \mathcal{T}_{k,d}, k = 1, \ldots, M, d = 0, \ldots, k-1$. The dual problem of the partial clique code LP (P5′) is defined as:

$$\max_{\substack{\lambda_m\geq 0\\ \mu_{T_{k,d}}\geq 0}} \min_{y_{T_{k,d}}\in\mathbb{R}} L(y_{T_{k,d}}, \lambda_m, \mu_{T_{k,d}})$$

Note that,

$$\min_{y_{T_{k,d}}\in\mathbb{R}} L(y_{T_{k,d}}, \lambda_m, \mu_{T_{k,d}}) = \begin{cases} \sum_{m=1}^{M}\lambda_m w_{p_m} & \begin{smallmatrix}(k-d)-\sum_{m=1}^{M}\lambda_m\mathbf{1}_{\{p_m\in T_{k,d}\}}-\mu_{T_{k,d}}=0\\ \forall T_{k,d}\in\mathcal{T}_{k,d}, k=1,\ldots,M, d=0,\ldots,k-1\end{smallmatrix} \\ -\infty & \text{otherwise} \end{cases}$$

Then, the dual problem of the partial clique code LP (P5′) can be written as,

$$\begin{aligned}
\max_{\lambda_m,\mu_{T_{k,d}}} & \sum_{m=1}^{M}\lambda_m w_{p_m} \\
\text{s.t.} \quad & (k-d) - \sum_{m=1}^{M}\lambda_m\mathbf{1}_{\{p_m\in T_{k,d}\}} - \mu_{T_{k,d}} = 0, \forall T_{k,d}\in\mathcal{T}_{k,d}, k = 1, \ldots, M, d = 0, \ldots, k-1 \\
& \lambda_m \geq 0, \quad m = 1, \ldots, M \\
& \mu_{T_{k,d}} \geq 0, \quad \begin{smallmatrix}\forall T_{k,d}\in\mathcal{T}_{k,d}\\ k=1,\ldots,M, d=0,\ldots,k-1,\end{smallmatrix}
\end{aligned}$$

Eliminating variables $\mu_{T_{k,d}}, \forall T_{k,d} \in \mathcal{T}_{k,d}, k = 1, \ldots, M, d = 0, \ldots, k-1$, we obtain

$$\begin{aligned} \max_{\lambda_m} \quad & \sum_{m=1}^{M} \lambda_m w_{p_m} \\ \text{s.t.} \quad & \sum_{m=1}^{M} \lambda_m \mathbf{1}_{\{p_m \in T_{k,d}\}} \leq (k-d), \quad \substack{\forall T_{k,d} \in \mathcal{T}_{k,d}, \\ \forall k=1,\ldots,M, d=0,\ldots,k-1} \\ & \lambda_m \geq 0, \quad m = 1, \ldots, M \end{aligned}$$

Now consider all the M packet vertices, i.e., all $T_{1,0} \in \mathcal{T}_{1,0}$. The corresponding constraints $\sum_{m=1}^{M} \lambda_m \mathbf{1}_{\{p_m \in T_{k,d}\}} \leq (k-d), \forall T_{1,0} \in \mathcal{T}_{1,0}$ can be simplified as $\lambda_m \leq 1, m = 1, \ldots, M$. Hence, the above linear programming problem is the LP relaxation of (P6). □

IP (P6) seems quite different from the maximum acyclic subgraph IP (P1) and it seems that there exists no duality between the optimal partial clique code and the MAS lower bound. However, the following lemma shows that the maximum acyclic subgraph IP (P1) and IP (P6) are two equivalent problems.

Lemma 9.5. *For any (unicast) index coding problem $\mathcal{G}$, the maximum acyclic subgraph IP (P1) and IP (P6) are two equivalent problems.*

Proof. Note that the objective function in problem (P1) is the same as that in problem (P6). To prove problems (P1) and (P6) are equivalent, we show that $x_m \in \{0, 1\}, m = 1, \ldots, M$ is feasible to problem (P1) if and only if it is feasible to problem (P6).

- Feasible to (P6) ⇒ feasible to (P1): Assume $x_m \in \{0, 1\}, m = 1, \ldots, M$ is feasible to (P6). For any cycle $C_i, \forall C_i \in \mathcal{C}_i, i = 2, \ldots, L$ involving i packet vertices in $\mathcal{G}$, let us consider the partial clique $T_{i,d}$ formed by the i packet vertices and i user vertices in this i-cycle. By the definition of a cycle, each user vertex has at least one packet vertex among these i packet vertices as side information. So $d \geq 1$. Since $x_m \in \{0, 1\}, m = 1, \ldots, M$ satisfies the inequality constraints in (P6), at least d packet vertices among these i packet vertices are deleted. Since at least one packet vertex of the cycle is deleted, cycle C_i cannot be complete. Hence, $x_m, m = 1, \ldots, M$ yields an acyclic subgraph of $\mathcal{G}$.

- Feasible to (P1) ⇒ feasible to (P6): Assume $x_m \in \{0, 1\}, m = 1, \ldots, M$ is feasible to (P1). For any partial clique $T_{k,d}$, if $d = 0$, then constraint $\sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in T_{k,d}\}} \leq k - d$ is trivially satisfied. Without loss of generality, assume $1 \leq d \leq k-1$. Then, in this

partial clique $T_{k,d}$, each user vertex has at least d outgoing arcs. So we can find a cycle in this partial clique. (To find a cycle, we start at any vertex, traverse a path from vertex to vertex using any outgoing link and discover a cycle when we revisit a vertex.) Since $x_m \in \{0,1\}, m = 1,\ldots,M$ is feasible to (P1), at least one packet vertex in this cycle is deleted. Assume d_1 packet vertices are deleted. These deleted packet vertices are also vertices in partial clique $T_{k,d}$. If $d_1 \geq d$, then the constraint over $T_{k,d}$ is satisfied. If $d_1 < d$, then we continue to consider the remaining part of $T_{k,d}$ after deleting these d_1 packet vertices. In the remaining part, each user vertex has at least $d-d_1$ outgoing arcs. A similar argument as above shows that we are still able to find a new cycle in the remaining part and at least one packet vertex in the cycle is deleted. Assume d_2 packet vertices in the new cycle are deleted. If $d_1 + d_2 < d$, we can repeat this process until at least d packet vertices are shown to be deleted. That is to say, constraint $\sum_{m=1}^{M} x_m \mathbf{1}_{\{p_m \in T_{k,d}\}} \leq k - d$ over all $T_{k,d}$ is satisfied. Hence, $x_m \in \{0,1\}, m = 1,\ldots,M$ satisfies the constraints of (P6).

□

The above lemma indicates that IP (P6) is another representation of the maximum acyclic subgraph IP (P1). However, this new representation is non-trivial. The LP relaxations of IP (P6) and the maximum acyclic subgraph IP (P1) correspond to partial clique codes and cyclic codes, respectively. Lemma 9.3 demonstrates that codes associated with IP (P6) in general have better performance than codes associated with the maximum acyclic subgraph IP (P1).

9.5.3 Discussion

The integer linear programs (P1) and (P6) are two different representations of the same problem of finding the maximum acyclic subgraph bound. Different representations of an integer linear program can yield LP relaxations with different integrality gaps. In Section 9.3 and this section, we show that the LP relaxation of (P1) is the (dual) problem of finding the optimal vector cyclic code, and the LP relaxation of (P6) is the (dual) problem of finding the optimal vector partial clique code. The performance of partial clique codes is no worse than that of cyclic codes. Hence, the integrality gap of the LP relaxation of (P6) is no larger than that of the LP relaxation of (P1). The relations between various problems in this chapter are illustrated in Figure 9.6. Note that (P2′) and (P5′) require a large packet size, while (P5) and (P5′) require

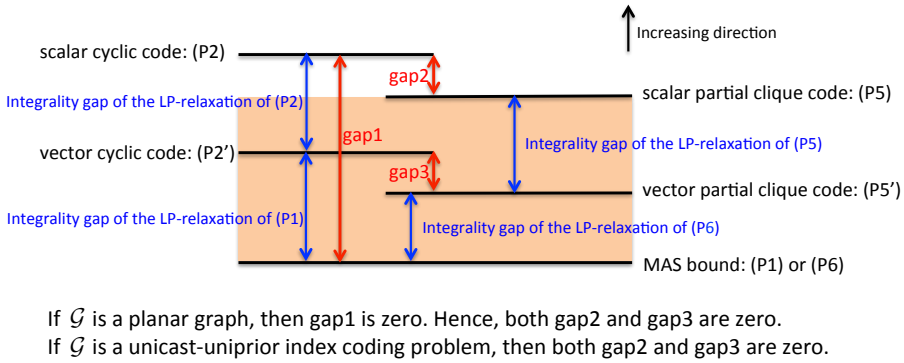

Figure 9.6: The relations between various problems in Chapter 9.

encoding in a large finite field. The graph parameter *minrank* is known to be optimal over scalar linear codes [BYBJK11], and hence lies somewhere in the shaded region between (P5) and the MAS bound.

Since there are various techniques for obtaining tight LP relaxations of an integer linear program [NW88], a potential approach to design good code structures for the index coding problem is to explore different representations of the maximum acyclic subgraph IP (P1) for which the LP relaxations have small integrality gaps. If the dual problem of such an LP relaxation can be interpreted as a code, then this is a good code for the index coding problem.

9.6 Chapter Summary

This chapter studies index coding from a perspective of optimization and duality. It illustrates the inherent duality between the information theoretic maximum acyclic subgraph (MAS) lower bound and the optimal cyclic codes and partial clique codes. The performance of both codes is bounded by the respective integrality gap of two different LP relaxations of the integer program that defines the MAS bound. In the special case when the index coding problem has a planar digraph representation, the integrality gap associated with cyclic coding is shown to be zero. So the exact optimality is achieved by cyclic coding. For general (non-planar) problems, the LP-relaxation associated with partial clique coding provides an integrality gap that is no worse, and often better, than the previous gap. These results provide new insight into the index coding problem and suggest that good codes can be found by exploring different relaxations of the MAS bound problem.

Chapter 10

Conclusions

In this thesis, we develop new Lagrangian methods for constrained convex programs with complicated functional constraints. Existing Lagrangian methods either have a slow $O(\frac{1}{\epsilon^2})$ convergence time or can only solve problems with linear equality constraint functions. The new methods developed in this thesis are proven to have a faster $O(\frac{1}{\epsilon})$ convergence time and can be implemented in parallel in most cases of interest. The per-iteration complexity of the new methods is also as small as that of existing Lagrangian methods. The design intuition and performance analysis of our new methods is different from conventional analysis techniques for existing Lagrangian methods and is based on a drift-plus-penalty type analysis, which is originally proposed for stochastic network optimization in dynamic queueing networks.

Most existing backpressure algorithms for joint rate control and routing in data networks can be interpreted as distributive applications of certain Lagrangian methods for a multi-commodity network flow formulation. By adapting our new Lagrangian methods, we are able to develop new backpressure algorithms that have the best utility and queue length tradeoff among all known backpressure algorithms.

The new Lagrangian methods are further adapted to develop new learning algorithms for online convex optimization with constraints. The two developed learning algorithms are proven to achieve the best regret and constraint violations for online convex optimization with stochastic constraints and online convex optimization with long term constraints, respectively. Power control for energy harvesting devices with outdated state information is closely related to online convex optimization with stochastic constraints but is restricted to a more stringent energy availability constraint. For this problem, we develop dynamic power control policy to achieve an $O(\epsilon)$ optimal utility by using a battery with capacity $O(1/\epsilon)$.

In this thesis, we also extend existing stochastic constrained convex optimization techniques and utilize Lagrangian duality theory for constrained convex optimization to study two other important problems in wireless communication and network coding. In the first problem, we adapted the conventional drift-plus-penalty technique for stochastic optimization and Zinkevich's projected online gradient descent for online convex optimization to develop new dynamic transmit covariance design policies for MIMO fading systems with unknown channel distributions and inaccurate channel state information. In the second problem, we study the index coding problem and characterize the optimality of two representative linear codes by studying the integrality gap between the integer linear program from an information theoretical lower bound and its linear programming relaxations and the Lagrangian duality between various linear programming relaxations and their dual problems.

Bibliography

[ALS+08] N. Alon, E. Lubetzky, U. Stav, A. Weinstein, and A. Hassidim. Broadcasting with side information. In *Proceedings of IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 823–832, 2008.

[BA03] H. Bai and M. Atiquzzaman. Error modeling schemes for fading channels in wireless communications: A survey. *IEEE Communications Surveys & Tutorials*, 5(2):2–9, 2003.

[BDH+08] P. L. Bartlett, V. Dani, T. Hayes, S. Kakade, A. Rakhlin, and A. Tewari. High-probability regret bounds for bandit online linear optimization. In *Proceedings of Conference on Learning Theory (COLT)*, 2008.

[Ber99] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

[BG02] R. A. Berry and R. G. Gallager. Communication over fading channels with delay constraints. *IEEE Transactions on Information Theory*, 48(5):1135–1149, 2002.

[BGD13] P. Blasco, D. Gunduz, and M. Dohler. A learning theoretic approach to energy harvesting communication system optimization. *IEEE Transactions on Wireless Communications*, 12(4):1872–1882, 2013.

[BK98] Y. Birk and T. Kol. Informed-source coding-on-demand (ISCOD) over broadcast channels. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 1998.

[BK06] Y. Birk and T. Kol. Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients. *IEEE Transactions on Information Theory*, 52(6):2825 – 2830, June 2006.

[BKL10] A. Blasiak, R. Kleinberg, and E. Lubetzky. Index coding via linear programming. *arXiv:1004.1379*, 2010.

[BKVH07] S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

[BLEM+09] N. Buchbinder, L. Lewin-Eytan, I. Menache, J. S. Naor, and A. Orda. Dynamic power allocation under arbitrary varying channels : An online approach. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2009.

[BLEM+10] N. Buchbinder, L. Lewin-Eytan, I. Menache, J. S. Naor, and A. Orda. Dynamic power allocation under arbitrary varying channels : The multi-user case. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2010.

[BNO03] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

[Bol98] B. Bollobás. *Modern Graph Theory*, volume 184. Springer, 1998.

[BPC+11] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[BSS06] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, 2006.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[BX05] S. Boyd and L. Xiao. Least-squares covariance matrix adjustment. *SIAM Journal on Matrix Analysis and Applications*, 27(2):532–546, 2005.

[BYBJK11] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol. Index coding with side information. *IEEE Transactions on Information Theory*, 57(3):1479–1494, March 2011.

[CASL11] M. A. R. Chaudhry, Z. Asad, A. Sprintson, and M. Langberg. On the complementary index coding problem. In *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2011.

[CBL06] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[CBLW96] N. Cesa-Bianchi, P. M. Long, and M. K. Warmuth. Worst-case quadratic loss bounds for prediction using linear functions and gradient descent. *IEEE Transactions on Neural Networks*, 7(3):604–619, 1996.

[CGP15] A. Cotter, M. Gupta, and J. Pfeifer. A light touch for heavily constrained SGD. In *Proceedings of Conference on Learning Theory (COLT)*, 2015.

[Dev77] L. P. Devroye. A uniform bound for the deviation of empirical distribution functions. *Journal of Multivariate Analysis*, 7(4):594–597, 1977.

[Doo53] J. L. Doob. *Stochastic processes*. Wiley New York, 1953.

[DSSSC08] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l_1-ball for learning in high dimensions. In *Proceedings of International Conference on Machine Learning (ICML)*, 2008.

[Dur10] R. Durrett. *Probability: Theory and Examples*. Cambridge University Press, 2010.

[DY16] W. Deng and W. Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *Journal of Scientific Computing*, 66(3):889–916, 2016.

[ERL15] M. Effros, S. E. Rouayheb, and M. Langberg. An equivalence between network coding and index coding. *IEEE Transactions on Information Theory*, 61(5):2478–2487, 2015.

[ES06] A. Eryilmaz and R. Srikant. Joint congestion control, routing, and mac for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(8):1514–1524, 2006.

[ES12] A. Eryilmaz and R. Srikant. Asymptotically tight steady-state queue length bounds implied by drift conditions. *Queueing Systems*, 72:311–359, 2012.

[ETW12] U. Erez, M. D. Trott, and G. W. Wornell. Rateless coding for Gaussian channels. *IEEE Transactions on Information Theory*, 58(2):530–547, 2012.

[FHM07] A. Feiten, S. Hanly, and R. Mathar. Derivatives of mutual information in Gaussian vector channels with applications. In *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2007.

[FLEP10] Y. Fan, L. Lai, E. Erkip, and H. V. Poor. Rateless coding for MIMO fading channels: performance limits and code construction. *IEEE Transactions on Wireless Communications*, 9(4):1288–1292, 2010.

[GGT10] M. Gatzianas, L. Georgiadis, and L. Tassiulas. Control of wireless networks with rechargeable batteries. *IEEE Transactions on Wireless Communications*, 9(2):581–593, 2010.

[GLS93] M. Grotschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimizations*. Springer-Verlag, 1993.

[GMPV09] A. Ghosh, P. McAfee, K. Papineni, and S. Vassilvitskii. Bidding for representative allocations for display advertising. In *Proceedings of International Workshop on Internet and Network Economics (WINE)*, 2009.

[GNT06] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, 2006.

[Gol05] A. Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.

[Gor99] G. J. Gordon. Regret bounds for prediction problems. In *Proceeding of Conference on Learning Theory (COLT)*, 1999.

[GPS15] E. Gustavsson, M. Patriksson, and A.-B. Strömberg. Primal convergence from dual subgradient methods for convex optimization. *Mathematical programming*, 150(2):365–390, 2015.

[GT11a] A. Goldfarb and C. Tucker. Online display advertising: targeting and obtrusiveness. *Marketing Science*, 30(3):389–404, 2011.

[GT11b] B. Guenin and R. Thomas. Packing directed circuits exactly. *Combinatorica*, 31(4):397– 421, 2011.

[Haj82] B. Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 14(3):502–525, 1982.

[HAK07] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2007.

[Haz16] E. Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3–4):157–325, 2016.

[HG07] A. Hjørungnes and D. Gesbert. Complex-valued matrix differentiation: Techniques and key results. *IEEE Transactions on Signal Processing*, 55(6):2740–2746, 2007.

[HH15] E. Hossain and M. Hasan. 5G cellular: key enabling technologies and research challenges. *IEEE Instrumentation & Measurement Magazine*, 18(3):11–21, 2015.

[HJ85] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[HJ91] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

[Hjø11] A. Hjørungnes. *Complex-valued Matrix Derivatives: with Applications in Signal Processing and Communications*. Cambridge University Press, 2011.

[HL16] M. Hong and Z.-Q. Luo. On the linear convergence of the alternating direction method of multipliers. *Mathematical Programming*, 2016.

[HMK+06] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, October 2006.

[HMNK13] L. Huang, S. Moeller, M. J. Neely, and B. Krishnamachari. LIFO-backpressure achieves near-optimal utility-delay tradeoff. *IEEE/ACM Transactions on Networking*, 21(3):831–844, 2013.

[HN11] L. Huang and M. J. Neely. Delay reduction via lagrange multipliers in stochastic network optimization. *IEEE Transactions on Automatic Control*, 56(4):842–857, 2011.

[HN13] L. Huang and M. J. Neely. Utility optimal scheduling in energy-harvesting networks. *IEEE/ACM Transactions on Networking*, 21(4):1117–1130, 2013.

[HR11] Y. Hu and A. Ribeiro. Adaptive distributed algorithms for optimal random access channels. *IEEE Transactions on Wireless Communications*, 10(8):2703–2715, 2011.

[HUL01] J.-B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer, 2001.

[HY12] B. He and X. Yuan. On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012.

[JB04] E. Jorswieck and H. Boche. Channel capacity and capacity-range of beamforming in MIMO wireless systems under correlated fading with covariance feedback. *IEEE Transactions on Wireless Communications*, 3(5):1543–1553, 2004.

[JHA16] R. Jenatton, J. Huang, and C. Archambeau. Adaptive algorithms for online convex optimization with long-term constraints. In *Proceedings of International Conference on Machine Learning (ICML)*, 2016.

[JP03] S. K. Jayaweera and H. V. Poor. Capacity of multiple-antenna systems with both receiver and transmitter channel state information. *IEEE Transactions on Information Theory*, 49(10):2697–2709, 2003.

[JVG01] S. A. Jafar, S. Vishwanath, and A. Goldsmith. Channel capacity and beamforming for multiple transmit and receive antennas with covariance feedback. In *Proceedings of IEEE International Conference on Communications (ICC)*, 2001.

[Kel97] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8(1):33–37, 1997.

[KHZS07] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems*, 6(4), 2007.

[KMS+15] P. Kamalinejad, C. Mahapatra, Z. Sheng, S. Mirabbasi, V. C. Leung, and Y. L. Guan. Wireless energy harvesting for the internet of things. *IEEE Communications Magazine*, 53(6):102–108, 2015.

[KMT98] F. P. Kelly, A. K. Maulloo, and D. K. Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, 1998.

[KW97] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

[LHSS09] J. Liu, Y. T. Hou, Y. Shi, and H. D. Sherali. On performance optimization for multi-carrier MIMO Ad-Hoc networks. In *Proceedings of ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, 2009.

[LK06] V. K. Lau and Y.-K. R. Kwok. *Channel-Adaptive Technologies and Cross-Layer Designs for Wireless Systems with Multiple Antennas: Theory and Applications*. John Wiley & Sons, 2006.

[LL97] T. Larsson and Z. Liu. A Lagrangean relaxation scheme for structured linear programs with application to multicommodity network flows. *Optimization*, 40(3):247–284, 1997.

[LL99] S. H. Low and D. E. Lapsley. Optimization flow control—I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, 1999.

[LMS06] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff. Opportunistic power scheduling for dynamic multi-server wireless systems. *IEEE Transactions on Wireless Communications*, 5(6):1506–1515, 2006.

[LMZ15] T.-Y. Lin, S.-Q. Ma, and S.-Z. Zhang. On the sublinear convergence rate of multi-block ADMM. *Journal of the Operations Research Society of China*, 3(3):251–274, 2015.

[Low03] S. H. Low. A duality model of TCP and queue management algorithms. *IEEE/ACM Transactions on Networking*, 11(4):525–536, 2003.

[LPS99] T. Larsson, M. Patriksson, and A.-B. Strömberg. Ergodic, primal convergence in dual subgradient schemes for convex programming. *Mathematical programming*, 86(2):283–312, 1999.

[LS04] X. Lin and N. B. Shroff. Joint rate control and scheduling in multihop wireless networks. In *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2004.

[LS06] X. Lin and N. B. Shroff. Utility maximization for communication networks with multipath routing. *IEEE Transactions on Automatic Control*, 51(5):766–781, 2006.

[LS09] E. Lubetzky and U. Stav. Nonlinear index coding outperforming the linear optimum. *IEEE Transactions on Information Theory*, 55(8):3544–3551, August 2009.

[LSXS15] J. Liu, N. B. Shroff, C. H. Xia, and H. D. Sherali. Joint congestion control and routing optimization: An efficient second-order distributed approach. *IEEE/ACM Transactions on Networking*, 24(3):1404–1420, 2015.

[LTCS16] B. Li, D. Tse, K. Chen, and H. Shen. Capacity-achieving rateless polar codes. In *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2016.

[LY78] C. Lucchesi and D. Younger. A minimax theorem for directed graphs. *J. London Math. Soc.(2)*, 17(3):369 – 374, 1978.

[LZ16] G. Lan and Z. Zhou. Algorithms for stochastic optimization with expectation constraints. *arXiv:1604.03887*, 2016.

[MB16] P. Mertikopoulos and E. V. Belmega. Learning to be green: Robust energy efficiency maximization in dynamic MIMO–OFDM systems. *IEEE Journal on Selected Areas in Communications*, 34(4):743–757, 2016.

[MJY12] M. Mahdavi, R. Jin, and T. Yang. Trading regret for efficiency: online convex optimization with long term constraints. *Journal of Machine Learning Research*, 13(1):2503–2528, 2012.

[MM16] P. Mertikopoulos and A. L. Moustakas. Learning in an uncertain world: MIMO covariance matrix optimization with imperfect feedback. *IEEE Transactions on Signal Processing*, 64(1):5–18, 2016.

[MSZ13] N. Michelusi, K. Stamatiou, and M. Zorzi. Transmission policies for energy harvesting sensors with time-correlated energy supply. *IEEE Transactions on Communications*, 61(7):2988–3001, 2013.

[MTY09] S. Mannor, J. N. Tsitsiklis, and J. Y. Yu. Online learning with sample path constraints. *Journal of Machine Learning Research*, 10:569–590, March 2009.

[MYJ13] M. Mahdavi, T. Yang, and R. Jin. Stochastic convex optimization with multiple objectives. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[Nee03] M. J. Neely. *Dynamic Power Allocation and Routing for Satellite and Wireless Networks with Time Varying Channels*. PhD thesis, Massachusetts Institute of Technology, 2003.

[Nee05] 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*, 2005.

[Nee06] M. J. Neely. Super-fast delay tradeoffs for utility optimal fair scheduling in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(8):1489–1501, 2006.

[Nee07] M. J. Neely. Optimal energy and delay tradeoffs for multiuser wireless downlinks. *IEEE Transactions on Information Theory*, 53(9):3095–3113, 2007.

[Nee10] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool Publishers, 2010.

[Nee14] M. J. Neely. A simple convergence time analysis of drift-plus-penalty for stochastic optimization and convex programs. *arXiv:1412.0791*, 2014.

[Nee15] M. J. Neely. Energy-aware wireless scheduling with near optimal backlog and convergence time tradeoffs. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2015.

[Nee16] M. J. Neely. Energy-aware wireless scheduling with near-optimal backlog and convergence time tradeoffs. *IEEE/ACM Transactions on Networking*, 24(4):2223–2236, 2016.

[Nes04] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Science & Business Media, 2004.

[NMR05] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic power allocation and routing for time-varying wireless networks. *IEEE Journal on Selected Areas in Communications*, 23(1):89–103, 2005.

[NN14] I. Necoara and V. Nedelcu. Rate analysis of inexact dual first-order methods application to dual decomposition. *IEEE Transactions on Automatic Control*, 59(5):1232–1243, May 2014.

[NNG15] I. Necoara, Y. Nesterov, and F. Glineur. Linear convergence of first order methods for non-strongly convex optimization. *arXiv preprint arXiv:1504.06298v4*, 2015.

[NO09a] A. Nedić and A. Ozdaglar. Approximate primal solutions and rate analysis for dual subgradient methods. *SIAM Journal on Optimization*, 19(4):1757–1780, 2009.

[NO09b] A. Nedić and A. Ozdaglar. Subgradient methods for saddle-point problems. *Journal of Optimization Theory and Applications*, 142(1):205–228, 2009.

[NP16] I. Necoara and A. Patrascu. Iteration complexity analyisis of dual first order methods for conic convex programming. *Optimization Method and Software*, 31(3):645–678, 2016.

[NPN15] I. Necoara, A. Patrascu, and A. Nedić. Complexity certifications of first-order inexact lagrangian methods for general convex programming: Application to real-time mpc. In *Developments in Model-Based Optimization and Control*, pages 3–26. Springer, 2015.

[NTZ13] M. J. Neely, A. S. Tehrani, and Z. Zhang. Dynamic index coding for wireless broadcast networks. *IEEE Transactions on Information Theory*, 59(11):7525–7540, November 2013.

[NW88] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*, volume 18. Wiley New York, 1988.

[NW06] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Science & Business Media, 2006.

[OH12] L. Ong and C. K. Ho. Optimal index codes for a class of multicast networks with receiver side information. In *Proceedings of IEEE International Conference on Communications (ICC)*, 2012.

[ÖLR14] B. Özbek and D. Le Ruyet. *Feedback Strategies for Wireless Communication*. Springer, 2014.

[PB13] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013.

[PC06] D. P. Palomar and M. Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8):1439–1451, 2006.

[PCL03] D. P. Palomar, J. M. Cioffi, and M. A. Lagunas. Uniform power allocation in MIMO channels: a game-theoretic approach. *IEEE Transactions on Information Theory*, 49(7):1707–1727, July 2003.

[Pee96] R. Peeters. Orthogonal representations over finite fields and the chromatic number of graphs. *Combinatorica*, 16(3):417–431, 1996.

[PL03] D. P. Palomar and M. A. Lagunas. Joint transmit-receive space-time equalization in spatially correlated MIMO channels: A beamforming approach. *IEEE Journal on Selected Areas in Communications*, 21(5):730–743, 2003.

[PS05] J. A. Paradiso and T. Starner. Energy scavenging for mobile and wireless electronics. *IEEE Pervasive Computing*, 4(1):18–27, 2005.

[Rag88] P. Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2):130–143, 1988.

[Rib10] A. Ribeiro. Ergodic stochastic optimization algorithms for wireless communication and networking. *IEEE Transactions on Signal Processing*, 58(12):6369–6386, 2010.

[RSG10] S. E. Rouayheb, A. Sprintson, and C. Georghiades. On the index coding problem and its relation to network coding and matroid theory. *IEEE Transactions on Information Theory*, 56(7):3187–3195, July 2010.

[RT87] P. Raghavan and C. D. Tompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[SB94] C. P. Simon and L. Blume. *Mathematics for Economists*. Norton New York, 1994.

[SC96] H. D. Sherali and G. Choi. Recovery of primal solutions when using subgradient optimization methods to solve lagrangian duals of linear programs. *Operations Research Letters*, 19(3):105–113, 1996.

[Ser09] R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. John Wiley & Sons, 2009.

[SHN14] S. Supittayapornpong, L. Huang, and M. J. Neely. Time-average optimization with nonconvex decision set and its convergence. In *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2014.

[Sho85] N. Z. Shor. *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag, 1985.

[SK11] S. Sudevalayam and P. Kulkarni. Energy harvesting sensor nodes: Survey and implications. *IEEE Communications Surveys & Tutorials*, 13(3):443–461, 2011.

[SMT15] I. Stiakogiannakis, P. Mertikopoulos, and C. Touati. Adaptive power allocation and control in time-varying multi-carrier MIMO networks. *arXiv:1503.02155*, 2015.

[SPB09] G. Scutari, D. P. Palomar, and S. Barbarossa. The MIMO iterative waterfilling algorithm. *IEEE Transactions on Signal Processing*, 57(5):1917–1935, 2009.

[SS11] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

[Sto05] A. L. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, 50(4):401–457, 2005.

[TDN12] A. S. Tehrani, A. G. Dimakis, and M. J. Neely. Bipartite index coding. In *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2012.

[TE92] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, 1992.

[Tel99] I. E. Telatar. Capacity of multi-antenna Gaussian channels. *European Transactions on Telecommunications*, 10(6):585–596, 1999.

[TTM11] H. Terelius, U. Topcu, and R. M. Murray. Decentralized multi-agent optimization via dual decomposition. In *IFAC World Congress*, 2011.

[TV05] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[TV15] T. Tao and V. Vu. Random matrices: universality of local spectral statistics of non-hermitian matrices. *The Annals of Probability*, 43(2):782–874, 2015.

[TY12] K. Tutuncuoglu and A. Yener. Optimum transmission policies for battery limited energy harvesting nodes. *IEEE Transactions on Wireless Communications*, 11(3):1180–1189, 2012.

[UUNS11] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam. Optimal power cost management using stored energy in data centers. *Proceedings of ACM SIGMETRICS*, 2011.

[UYE+15] S. Ulukus, A. Yener, E. Erkip, O. Simeone, M. Zorzi, P. Grover, and K. Huang. Energy harvesting wireless communications: A review of recent advances. *IEEE Journal on Selected Areas in Communications*, 33(3):360–381, 2015.

[VLS05] V. V. Veeravalli, Y. Liang, and A. M. Sayeed. Correlated MIMO wireless channels: capacity, optimal signaling, and asymptotics. *IEEE Transactions on Information Theory*, 51(6):2058–2072, 2005.

[VP07] M. Vu and A. Paulraj. On the capacity of MIMO wireless channels with dynamic CSIT. *IEEE Journal on Selected Areas in Communications*, 25(7):1269–1283, 2007.

[Vu02] V. Vu. Concentration of non-lipschitz functions and applications. *Random Structures & Algorithms*, 20(3):262–316, 2002.

[WO13] E. Wei and A. Ozdaglar. On the $O(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers. In *Proceedings of IEEE Global Conference on Signal and Information Processing*, 2013.

[WOJ13] E. Wei, A. Ozdaglar, and A. Jadbabaie. A distributed Newton method for network utility maximization–I: algorithm. *IEEE Transactions on Automatic Control*, 58(9):2162–2175, 2013.

[WWW+17] W. Wu, J. Wang, X. Wang, F. Shan, and J. Luo. Online throughput maximization for energy harvesting communication systems with battery overflow. *IEEE Transactions on Mobile Computing*, 16(1):185–197, 2017.

[WYN15] X. Wei, H. Yu, and M. J. Neely. A sample path convergence time analysis of drift-plus-penalty for stochastic optimization. *arXiv:1510.02973v2*, 2015.

[YN13] H. Yu and M. J. Neely. Duality codes and the integrality gap bound for index coding. In *Proceedings of 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2013.

[YN14] H. Yu and M. J. Neely. Duality codes and the integrality gap bound for index coding. *IEEE Transactions on Information Theory*, 60(11):7256–7268, 2014.

[YN15] H. Yu and M. J. Neely. On the convergence time of the drift-plus-penalty algorithm for strongly convex programs. In *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2015.

[YN16a] H. Yu and M. J. Neely. Dynamic power allocation in MIMO fading systems without channel distribution information. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2016.

[YN16b] H. Yu and M. J. Neely. A low complexity algorithm with $O(\sqrt{T})$ regret and finite constraint violations for online convex optimization with long term constraints. *arXiv:1604.02218*, 2016.

[YN16c] 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.

[YN17a] H. Yu and M. J. Neely. Dynamic transmit covariance design in MIMO fading systems with unknown channel distributions and inaccurate channel state information. *IEEE Transactions on Wireless Communications*, 16(6):pp.3996–4008, 2017.

[YN17b] H. Yu and M. J. Neely. A new backpressure algorithm for joint rate control and routing with vanishing utility optimality gaps and finite queue lengths. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[YN17c] H. Yu and M. J. Neely. A new backpressure algorithm for joint rate control and routing with vanishing utility optimality gaps and finite queue lengths. *arXiv:1701.04519*, 2017.

[YN17d] H. Yu and M. J. Neely. A primal-dual parallel method with $O(1/\epsilon)$ convergence for constrained composite convex programs. *arXiv:1708.00322*, 2017.

[YN17e] 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(2):pp.759–783, 2017.

[YN18a] H. Yu and M. J. Neely. Learning aided optimization for energy harvesting devices with outdated state information. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2018.

[YN18b] H. Yu and M. J. Neely. On the convergence time of dual subgradient methods for strongly convex programs. *IEEE Transacations on Automatic Control*, to appear, 2018.

[YNW17] H. Yu, M. J. Neely, and X. Wei. Online convex optimization with stochastic constraints. In *Advances in Neural Information Processing Systems*, pages 1427–1437, 2017.

[YU12] J. Yang and S. Ulukus. Optimal packet scheduling in an energy harvesting communication system. *IEEE Transactions on Communications*, 60(1):220–230, 2012.

[Zin03] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of International Conference on Machine Learning (ICML)*, 2003.

[ZRJ13] M. Zargham, A. Ribeiro, and A. Jadbabaie. Accelerated backpressure algorithm. In *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, 2013.