Online Optimization with Predictions and Switching Costs: Fast Algorithms and Fundamental Limits

Yingying Li, Guannan Qu, and Na Li

***Abstract*—This paper studies an online optimization problem with switching costs and a finite prediction window. We propose a computationally efficient algorithm, Receding Horizon Gradient Descent (RHGD), which only requires a finite number of gradient evaluations at each time. We show that both the dynamic regret and the competitive ratio of the algorithm decay exponentially fast with the length of the prediction window. Moreover, we provide a fundamental lower bound on dynamic regret for general online algorithms with a finite prediction window. The lower bound matches the dynamic regret of our RHGD, meaning that the performance can not improve significantly even with more computation. Lastly, we present simulation results to test our algorithm numerically.**

I. Introduction

A classic online convex optimization (OCO) problem considers a decision maker interacting with an uncertain and even adversarial environment for T stages. At each time $t \in \{1, \dots, T\}$, the decision maker picks an action x_t from a convex set X. Then the environment reveals a convex cost $f_t(\cdot)$. As a result, the decision maker suffers the cost $f_t(x_t)$ based on the chosen action. The goal is to minimize the total cost in T stages. Classic OCO has been studied for decades, with an focus on improving online algorithm performance measured by regrets and/or competitive ratio [1]–[4].

Recent years have witnessed a growing interest in applying online optimization to real-world systems, e.g. economic dispatch [5]–[7], data center scheduling [8], [9], electric vehical charging [10], [11], video streaming [12], and thermal control [13]. However, there are two features of these scenarios that are generally not captured by classic OCO literature: time coupling effect and prediction about the future.

Time coupling effect: While classic OCO setup assumes that stage costs $f_t(x_t)$ are completely decoupled with time, in reality it is usually not the case. For example, to change actions from x_{t-1} to x_t, decision makers usually suffer a switching cost or ramp cost $d(x_t - x_{t-1})$ [8], [9], [14], [15]. In this way, the stage cost becomes time coupled and is defined as: $C_t(x_{t-1}, x_t) := f_t(x_t) + d(x_t - x_{t-1})$.

Prediction: Classic OCO often models environment as adversary and assumes that no information is available about future cost functions. However, in most applications, there is certain amount of prediction about the future, especially the near future. For example, in electricity system, the system operator can obtain a good prediction about the future demand and generation [16] [17].

Recently, there are some studies from OCO community exploring the effect of prediction, but most of them do not consider time coupled stage costs [18], [19].

In contrast, there have been many control algorithms, in particular, Model Predictive Control (MPC, aka Receding Horizon Control), developed for decades to handle both the prediction effect and the time coupling effect. One major focus of MPC is to design control rules to stabilize a dynamical system. Additional goals include minimizing the total stage costs, as studied in economic MPC [20]–[22]. However, the classic MPC approaches require solving a large optimization problem at each time, which is usually computationally expensive. Though there have been many recent efforts trying to reduce the computational overhead, e.g. inexact MPC and suboptimal MPC [23]–[26], there are limited results on the efficiency loss of these algorithms, such as bounds on dynamic regret. This is partially due to the complexity of the underlying system dynamics. Lastly, other similar online control algorithms, such as Averaging Fixed Horizon Control (AFHC) [8], [9], [27], [28], suffer from the same problem of high computational costs.

Contribution of this paper: In this paper we consider an OCO problem with switching costs and a prediction window W. To design online algorithms for this problem, we first study the structure of offline gradient-based algorithms, and use it to motivate our online algorithm Receding Horizon Gradient Descent (RHGD) and an accelerated version Receding Horizon Accelerated Gradient (RHAG). Our algorithms only require $W + 1$ gradient evaluations at each time step, which is more computationally efficient compared to optimization-based algorithms such as RHC and AFHC . Besides, there is a smooth interpolation between our algorithm and classic online methods in settings without prediction: when $W = 0$, our algorithm reduces to the classic online gradient descent.

We analyze the online performance of RHGD and RHAG by comparing algorithm outputs and offline optimal solution. The comparison is measured by both dynamic regret and competitive ratio. We show that the dynamic regret of RHGD and RHAG decays exponentially with W. Furthermore, under mild conditions on the costs, our online algorithms' competitive ratio are upper bounded by terms that also exponenially decay with W. This demonstrates hat our online algorithm performs much better with a small improvement on prediction.

Moreover, we study the fundamental performance limit of online algorithms given a prediction window W. We give a lower bound on the dynamic regret which also decays

This work was supported by NSF CAREER 1553407 and ARPA-E NODES. Y. Li, G. Qu and N. Li are with the School of Engineering and Applied Sciences, Harvard University, 33 Oxford Street, Cambridge, MA 02138, USA (email: yingyingli@g.harvard.edu, gqu@g.harvard.edu, nali@seas.harvard.edu).

exponentially with W. The decaying rate surprisingly matches the asymptotic property of RHAG's upper bound. This means that RHAG achieves the optimal utilization of prediction information only by using few computational efforts, which is much more efficient than optimization-based algorithms such as RHC and AFHC.

Lastly, we use the economic dispatch problem of electricity systems to numerically test our algorithm. The simulation is consistent with our theoretical results.

A. Notations

For vector $x \in \mathbb{R}^n$ and set $X \subset \mathbb{R}^n$, norm $\|x\|$ refers to the Euclidean norm, and $\Pi_X(x)$ denotes the projection of x onto set X. Besides, we denote the transpose of x as x'. The same applies to matrix transpose. In addition, X^T denotes the Cartesian product of T copies of set X. Moreover, we define $[T]$ as the set $\{1, \ldots, T\}$ for a positive integer T. Finally, we define the notation for gradient and partial gradient. Consider function $f(x, y)$ for $x \in \mathbb{R}^m$ and $\mathbb{R}^n$. Let $\nabla f(x, y) \in \mathbb{R}^{m+n}$ be the gradient of function $f(x, y)$, and $\frac{\partial f}{\partial x}(x, y) \in \mathbb{R}^m$ be the partial gradient with respect to x.

II. Problem Formulation

In this paper we consider a variation of online convex optimization where the algorithm is subject to an additional switching cost on the change of actions from one time step to the next. If nothing is known about the future, one can expect that no online algorithm can perform well. Fortunately, in many practical applications, prediction with high precision is indeed available for the near future, e.g. wind generation and load forecast [16] [17]. Thus in this paper, we consider a situation where the algorithm has information about the current and a limited number of future cost functions.

Formally, we consider online convex optimization over a finite time horizon $t \in [T] := \{1, \ldots, T\}$. At each time step t, a sequence of costs $f_t, \ldots, f_{t+W-1}$ from a function class is revealed to the online algorithm. This W-lookahead window is perfect in the sense that these are the true cost functions the algorithm will experience in future time steps.[1] Given perfect $W-$lookahead, the decision maker needs to pick an action x_t from a compact convex action set $X \subset \mathbb{R}^n$. The goal is to minimize the total cost of online decisions in T stages $x = \{x_1, \ldots, x_T\} \in X^T \subseteq \mathbb{R}^{nT}$. The total cost is given by

$$C_1^T(x) = \sum_{t=1}^{T} \left(f_t(x_t) + \frac{\beta}{2}\|x_t - x_{t-1}\|^2 \right) \tag{1}$$

where x_0 is given and $\beta \geq 0$ is a weight parameter. Notice that here we consider quadratic switching cost functions, but the analysis can extend to other switching cost functions with properties such as monotonicity, convexity, and smoothness.

For the purpose of theoretical analysis, this paper considers a function class $\mathcal{F}_X(\alpha, l, G)$ which consists of functions defined on X that are α-strongly convex, differentiable and have l-Lipschitz continuous gradients. Mathematically, this means for every $t \in [T]$ and any $x_t, y_t \in X$, the following inequalities hold,

$$\begin{aligned} &i) \quad f_t(y_t) \geq f_t(x_t) + \langle \nabla f_t(x_t), y_t - x_t \rangle + \frac{\alpha}{2}\|y_t - x_t\|^2 \\ &ii) \quad \|\nabla f_t(y_t) - \nabla f_t(x_t)\| \leq l\|y_t - x_t\| \\ &iii) \quad \|\nabla f_t(x_t)\| \leq G \end{aligned}$$

Moreover, the action space X compact with diameter D:

$$\forall x, y \in X, \|x - y\| \leq D$$

Given the above properties, we can show the following nice properties hold for the total cost function $C_1^T(\cdot)$. The proof is deferred to Appendix A.

Lemma 1. *$C_1^T(x)$ is α-strongly convex and has L-Lipschitz continuous gradient, where $L = l + 4\beta$.*

The problem setup has natural applications in many areas. Here we briefly discuss two applications.

Example 1. *(Economic Dispatch in Power Systems.) Consider a power network with conventional generators and renewable energy supply. At time t, let $x_t = \{x_{t,i}\}_{i=1}^n$ be the outputs of n generators and X be the set of feasible outputs. The generation cost of generator i is $c^i(x_{t,i})$. The renewable supply is r_t and the power demand is d_t.*

At time t, the goal of economic dispatch is to reduce total generation cost while maintaining power balance: $\sum_{i=1}^n x_{t,i} + r_t = d_t$. Thus we incorporate imbalance penalty into the objective and consider the cost function

$$f_t(x_t) = \sum_{i=1}^{n} c^i(x_{t,i}) + \xi_t(\sum_{i=1}^{n} x_{t,i} - r_t - d_t)^2$$

where ξ_t is a penalty factor. In literature, $c^i(x_{t,i})$ is usually modeled as a quadratic function within capacity limit [5]. It is easy to see that $f_t(x_t)$ belongs to class $\mathcal{F}_X(\alpha, l, G)$.

In addition to the costs above, ramping process of conventional generators also incurs significant costs, e.g. maintenance and depreciation fee. In literature, such costs are referred as ramp costs and modeled as a quadratic function of ramping rate $\frac{\beta}{2}\|x_t - x_{t-1}\|^2 := \sum_{i=1}^n \frac{\beta}{2}\|x_{i,t} - x_{i,t-1}\|^2$ [14] [15]. As a result, the objective of economic dispatch for T stages is to minimize total costs including ramp costs

$$\min_{x_t} \sum_{t=1}^{T} \left(f_t(x_t) + \frac{\beta}{2}\|x_t - x_{t-1}\|^2 \right)$$

Although demand and renewable supply are random, prediction are available for a short time window [16] [17].

Example 2. *(Trajectory Tracking): Consider a simple dynamical system $x_{t+1} = x_t + u_t$, where x_t is the location of a robot, u_t is the control action (velocity of the robot). Let y_t be the location of the target at time t, and the tracking error is given by $f_t(x_t) = \frac{1}{2}\|x_t - y_t\|^2$. There will also be an energy loss for each control action, given by $\frac{\beta}{2}\|u_t\|^2 = \frac{\beta}{2}\|x_{t+1} - x_t\|^2$.*

[1] Although this assumption might be unrealistic, it is a good starting point to study the effect of prediction on online decision making. We will introduce prediction error in the future work.

The objective is to minimize the sum of the tracking error and the energy loss,

$$\min_{x_t} \sum_{t=0}^{T-1} \left(f_t(x_t) + \frac{\beta}{2}\|x_{t+1} - x_t\|^2 \right) + f_T(x_T).$$

In reality, there is usually a lookahead window W for the target trajectory y_t [29].

A. Performance Metrics

This paper studies online algorithm with two performance metrics: *dynamic regret* and *competitive ratio*. Both metrics are commonly used in literature [8], [19].

Before the formal definition, we introduce some notations and necessary concepts. Both dynamic regret and competitive ratio compare algorithm performance with optimal performance. Consider an online algorithm $\mathcal{A}$, the algorithm performance is defined as:

$$C_1^T(x^{\mathcal{A}}) = \sum_{t=1}^{T} \left(f_t(x_t^{\mathcal{A}}) + \frac{\beta}{2}\|x_t^{\mathcal{A}} - x_{t-1}^{\mathcal{A}}\|^2 \right)$$

where $x^{\mathcal{A}}$ denotes the outputs of algorithm $\mathcal{A}$ and $x_0^{\mathcal{A}} = x_0$. The optimal performance is defined by solving the optimization problem (1) in the offline setting, i.e. when all cost functions are known:

$$C_1^T(x^*) = \min_{x \in X^T} \sum_{t=1}^{T} \left(f_t(x_t) + \frac{\beta}{2}\|x_t - x_{t-1}\|^2 \right)$$

where x^* represents the optimal offline actions and $x_0^* = x_0$.

Moreover, it is well-known that online algorithm performance relies heavily on cost fluctuations over time (cite). To measure the cost fluctuation rate, we define *path length* of a cost function list $\{f_t\}_{t=1}^T$ by the accumulative absolute differences of stage optimizers in consecutive stages, i.e.

$$\sum_{t=1}^{T} \|\theta_t - \theta_{t-1}\| \tag{2}$$

where $\theta_t \in \arg\min_{x_t \in X} f_t(x_t)$ is the stage optimizer at stage t and $\theta_0 = x_0$. Path length is a commonly used way to model how fast cost function f_t changes over time [2][2].

In the following, we group function lists $\{f_t\}_{t=1}^T$ by their path lengths and define set $\mathcal{L}_T(L_T, \mathcal{F}_X(\alpha, l, G))$ as all function lists with path lengths bounded by L_T. Formally,

$$\begin{aligned}&\mathcal{L}_T(L_T, \mathcal{F}_X(\alpha, l, G)) \\ &:= \{\{f_t\}_{t=1}^T | \sum_{t=1}^{T} \|\theta_t - \theta_{t-1}\| \le L_T, f_t \in \mathcal{F}_X(\alpha, l, G)\}\end{aligned}$$

When $\mathcal{F}_X(\alpha, l, G)$ is clear from context, we write $\mathcal{L}_T(L_T)$.

Now, we are ready to define dynamic regret. Consider $f_t \in \mathcal{F}_X(\alpha, l, G)$, the dynamic regret of algorithm $\mathcal{A}$ is the maximal difference between algorithm performance and the optimal cost for any $\{f_t\}_{t=1}^T$ with path length at most L_T. More formally, the dynamic regret $\mathrm{Reg}(\mathcal{A}, L_T)$ is defined as

$$\mathrm{Reg}(\mathcal{A}, L_T) := \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \left(C_1^T(x^{\mathcal{A}}) - C_1^T(x^*) \right)$$

[2] [2] also gives an overview of other cost dynamics measures in literature.

Most literature, as well as this paper, looks for an algorithm that guarantees sublinear regret when path length is sublinear in T [2], [4], [19], [30].

The other metric used in this paper is competitive ratio, which is the maximal ratio between algorithm performance and the optimal performance:

$$\mathrm{CR}(\mathcal{A}, L_T(L_T)) := \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \frac{C_1^T(x^{\mathcal{A}})}{C_1^T(x^*)}.$$

To make this definition useful, an underlying assumption is that the cost function $f_t(x_t)$ is positive and not close to 0 [8]. Unlike dynamic regret, path length L_T and horizon T do not play a fundamental role in competitive ratio. This is because $C_1^T(x^*)$ also depends on L_T and T, so they can be cancelled by a well designed online algorithm. Therefore, both literature and this paper try to design algorithms that have a constant bound on competitive ratio for any L_T and T.

III. Classic Approaches

Before presenting our algorithm, we briefly introduces some classic algorithms in this section. For no-prediction setting, we introduce classic online gradient descent (OGD) and its theoretical performance. For prediction setting, we introduce a classic control algorithms Receding Horizon Control (RHC) and analyze its pros and cons.

A. Online Gradient Descent

Online Gradient Descent (OGD) is a classic online algorithm in no-prediction setting. The algorithm chooses actions by gradient update based on previous-step cost and action:

$$x_t = x_{t-1} - \frac{1}{\gamma}\nabla f_{t-1}(x_{t-1}) \tag{OGD}$$

Though OGD is well studied in literature [8] [2] [1], as far as we know, OGD's dynamic regret for OCO with switching cost has not been covered, thus we present it here.

Theorem 1. *For any function class $\mathcal{F}_X(\alpha, l, G)$ and any function list set $\mathcal{L}_T(L_T)$, the dynamic regret for online optimization with switching cost is upper bounded by:*

$$\mathrm{Reg}(OGD, L_T) \le (\beta/l + 1)\frac{G}{(1-\eta)}L_T \tag{3}$$

where $\eta = \sqrt{(1 - \frac{\alpha}{l})}$.

Proof. See Appendix. □

Moreover, in Section VI, we will show a general lower bound for online optimization with switching cost in Theorem 6. When $W = 0$, the lower bound matches OGD's regret upper bound up to a constant. Thus, when there is no prediction available, OGD is an optimal algorithm for online optimization with switching cost. This is quite surprising because OGD achieves the optimal regret only by taking one gradient evaluation at each time.

B. Receding Horizon Control and Its Variants

When there is a constant prediction window size W, Receding Horizon Control is a commonly used algorithm. At each time s, RHC solves a W-stage optimization problem:

$$\min_{X^W} C_s^{s+W-1}(x_s, \dots, x_{s+W-1}) + T_{s+W}(x_{s+W-1}) \quad (4)$$

where

$$C_s^{s+W-1}(\cdot) = \sum_{t=s}^{s+W-1} \left(f_t(x_t) + \frac{\beta}{2}\|x_t - x_{t-1}\|^2 \right), \quad (5)$$

x_{s-1} is determined by previous iterations and $T_{s+W}(x_{s+W-1})$ is a terminal cost function. Let $\{x_s^s, \dots, x_{s+W-1}^s\}$ denote the solution to (4). The output of RHC is x_s^s at time s.

Though RHC enjoys much better performance than OGD thanks to prediction information, one major drawback of RHC is the high computational burden at each time. Considering that OGD achieves optimal regret for $W = 0$ by using gradient updates, a natural question is whether we can utilize prediction effectively also by gradient updates. To answer this question is one major goal of this paper.

In the rest of this paper, we will introduce a gradient-based algorithm Receding Horizon Accelerated Gradient (RHAG) and show that it optimally utilizes the prediction information.

Before going to our algorithm design, we would like to comment on previous efforts on reducing computational complexity of RHC. In particular, the control community has proposed several methods, e.g. inexact MPC and suboptimal MPC, and studied properties of stability and transient performance for trajectories converging to steady state. However, in online optimization, optimal solutions generally do not converge. Thus, current theoretical results cannot be applied to the problem considered in this paper.

IV. Fast Algorithms

In this section, we will introduce two online algorithms: Receding Horizon Gradient Descent (RHGD) and Receding Horizon Accelerated Gradient (RHAG). Both algorithms are adapted from offline gradient based algorithms: gradient descent (GD) and Nesterov's accelerated gradient (NAG) respectively. Our online algorithms only evaluate gradients for $W+1$ times at each time, so they are much faster than existing optimization-based algorithms.

A. Receding Horizon Gradient Descent

Before introducing RHGD, we first discuss offline optimization problem, and the classic GD algorithm. We will analyze the special structure of our offline problem, which results in the special structure of GD updates. This special updating scheme motivates our online algorithm RHGD.

1) Offline Problem and Gradient Descent: Given cost functions $f_1, \dots, f_T$, the offline optimization problem is

$$\min_{x \in X^T} \; C_1^T(x) = \sum_{t=1}^{T} \left(f_t(x_t) + \frac{\beta}{2}\|x_t - x_{t-1}\|^2 \right) \quad (6)$$

Apply Gradient Descent to solve (6):

$$x^{(k)} = \Pi_{X^T} \left(x^{(k-1)} - \eta \nabla C_1^T(x^{(k-1)}) \right) \quad (7)$$

where $\eta > 0$ is the stepsize, $x^{(k)}$ denotes the kth update of x and the initial value $x^{(0)}$ is given. Considering the update of each x_t, we can rewrite the updating rule (7) as

$$x_t^{(k+1)} = \Pi_X \left(x_t^{(k)} - \eta g_t(x_{t-1}^{(k)}, x_t^{(k)}, x_{t+1}^{(k)}) \right), \; t \in [T] \quad (8)$$

where $g_t(\cdot)$ denotes the partial gradient of $C_1^T(\cdot)$ with respect to x_t, i.e. $g_t(\cdot) = \frac{\partial C_1^T}{\partial x_t}$. Moreover, due to the special structure of offline optimization (6), $g_t(\cdot)$ only depends on neighboring actions x_{t-1}, x_t, x_{t+1} and has explicit expression:

$$g_t(x_{t-1}, x_t, x_{t+1}) = \nabla f_t(x_t) + \beta(2x_t - x_{t-1} - x_{t+1}), \; t < T$$
$$g_T(x_{T-1}, x_T, x_{T+1}) = \nabla f_T(x_T) + \beta(x_T - x_{T-1})$$

To ease notation, we write $g_T(x_{T-1}, x_T, x_{T+1})$ even though $g_T(\cdot)$ does not depend on x_{T+1}.

Now let us think online scenario. The major difficulty of online optimization is the lack of future information. However, thanks to the special structure of our problem, rule (8) only needs one-step-forward information x_{t+1} to update x_t. Thus, given W-prediction, we are able to implement (8) in the online fashion, which motivates our design of RHGD.

Algorithm 1 Receding Horizon Gradient Descent

Inputs: x_0 (action at time 0), W, stepsizes γ, η
$x_1^{1-W} \leftarrow x_0$
for $s = 2 - W$ to T **do**
  I) Initialize value for x_{s+W}.
  if $s + W \le T$ **then**
    $x_{s+W}^s \leftarrow \Pi_X \left(x_{s+W-1}^{s-1} - \gamma \nabla f_{s+W-1}(x_{s+W-1}^{s-1}) \right)$
  II) Update value for $x_s, \dots, x_{s+W-1}$ *backwards.*
  for $t = \min(s + W - 1, T) : -1 : \max(s, 1)$ **do**
    $x_t^s \leftarrow \Pi_X \left(x_t^{s-1} - \eta g_t(x_{t-1}^{s-2}, x_t^{s-1}, x_{t+1}^s) \right)$
Ouputs: x_t^t at time $t = 1, \dots, T$.

2) Online Algorithm RHGD: Roughly speaking, RHGD uses OGD outputs as initial actions and updates actions based on GD-inspired rules. The pseudocode is given in Algorithm ??. An intuitive explanation is given below.

First of all, we introduce the notation. Remember that x_t is the action taken at time t. To determine x_t, RHGD starts computing it W steps before time t, i.e. for $s = t - W, \dots, t$. We let x_t^s denote the value of x_t computed at time s.

Based on the new notation, we briefly overview the major steps taken by RHGD to determine each action x_t. At time $t - W$, x_t is computed for the first time, thus being known as the initial value of x_t. Then, RHGD updates x_t once at each step for $s = t - W - 1, \dots, t$, so x_t^s is the kth update of x_t when $s = t - W + k$. Eventually, at time $s = t$, RHGD takes action x_t^t, which is the Wth update after initialization. To sum up, for any $t \in [T]$, we have

I): Initial value: x_t^{t-W}
II): kth update: x_t^s, where $s = t - W + k, \; k \in [W]$

Now, we explain the details of how RHGD initializes and updates x_t for each $t \in [T]$. First is the initializing rule. Remember that the prediction window at time $t - W$ is $[t - W, (t - W) + W - 1]$, so $f_t(\cdot)$ is unknown. As a result,

we have to initialize x_t based on previous-step initial action $x_{t-1}^{(t-1)-W}$ and previous-step cost function $f_{t-1}(\cdot)$:

$$x_t^{t-W} = \Pi_X \left(x_{t-1}^{(t-1)-W} - \gamma \nabla f_{t-1}(x_{t-1}^{(t-1)-W}) \right) \quad (9)$$

where $\gamma > 0$ is the stepsize and $x_1^{1-W} = x_0$. The initializing rule follows the same idea as OGD.

Next is the updating rule, which follows by rewritting (8). Remember that x_t^s is the kth update of x_t at time $s = t - W + k$. Similarly, $x_{t-1}^{s-2}, x_t^{s-1}, x_{t+1}^s$ are the $(k-1)$th update of x_{t-1}, x_t, x_{t+1} respectively. Therefore, for $s = t-W+1, \ldots, t$, we can rewrite (8) as

$$x_t^s = \Pi_X \left(x_t^{s-1} - \eta g_t(x_{t-1}^{s-2}, x_t^{s-1}, x_{t+1}^s) \right) \quad (10)$$

The only thing left is to show that (10) only uses available information. $g_t(\cdot)$ is available because $f_t(\cdot)$ is predictable at time $s = t - W + 1, \ldots, t$. In addition, x_{t-1}^{s-2}, x_t^{s-1} are available because they are computed before time s. The only tricky part is x_{t+1}^s which is also computed at time s. To deal with this, RHGD is designed to update x_{t+1}^s before x_t^s. More specifically, at time s, RHGD computes $x_s, \ldots, x_{s+W}$ backwards [3]. In this way, x_{t+1}^s is available when we compute x_t^s for $t = s, \ldots, s + W - 1$.

Now, it is straightforward to see that RHGD and offline GD have identical updating rules. This relation is formally stated below and crucial to our theoretical analysis in Section ??.

Theorem 2. *Given the same stepsize η, let $x_t^{(k)}$ denote the kth update according to GD, and x_t^s denote the update at time s. Notice that x_t^s is the kth update when $s = t - W + k$.*

If GD and RHGD shares the same initial values

$$x_t^{(0)} = x_t^{t-W}, \forall t \in [T]$$

then the output of RHGD is the same as offline GD after W iterations:

$$x_t^{(W)} = x_t^t, \forall t \in [T].$$

Proof. The main idea of the proof has already been discussed above. We omit the details due to space limit. □

The above explanation focuses on initializing and updating of a single action x_t. In the following, we summarize RHGD's computation at each time step s:

I) Initialize x_{s+W} according to (9),

ii) Update $x_s, \ldots, x_{s+W-1}$ backwards according to (10)

Both (9) and (10) only evaluate gradient for once, so at time s, RHGD in total only evaluates gradients for $W + 1$ times, which is much faster than solving optimization at each time.

B. Receding Horizon Accelerated Gradient

Receding Horizon Accelerated Gradient (RHAG) is similar to RHGD except that RHAG's updating rule is based on Nesterov's accelerated gradient method (NAG). Thus, we will first introduce NAG's updating rule for offline optimization, based on which we will design RHAG.

[3]See Line 7 and 8 in Algorithm 1

1) Offline Problem and Nesterov's Accelerated Gradient: Nesterov's Accelerated Gradient (NAG) is famous for being the optimal first order algorithm. It is more complicated than GD but enjoys faster convergence rate.

Here, we apply NAG to our offline problem (6) and write the updating rule for each action x_t for $t \in [T]$:

$$\begin{aligned} x_t^{(k)} &= \Pi_X \left(y_t^{(k-1)} - \eta g_t(y_{t-1}^{(k-1)}, y_t^{(k-1)} y_{t+1}^{(k-1)}) \right) \\ y_t^{(k)} &= (1+\lambda) x_t^{(k)} - \lambda x_t^{(k-1)} \end{aligned} \quad (11)$$

where $\lambda = \frac{1-\sqrt{\alpha\eta}}{1+\sqrt{\alpha\eta}}$ and $y_t^{(0)} = x_t^{(0)}$ is given.

Notice that NAG's updating rule (11) only needs one-step forward information y_{t+1} to compute x_t and y_t. This information dependency pattern is similar to that of GD's updating rule. Therefore, we can use the same trick to design RHAG based on NAG.

Algorithm 2 Receding Horizon Accelerated Gradient

Inputs: x_0 (action at time 0), W, α, stepsizes γ, η
$y_1^{1-W} \leftarrow x_0$, $x_1^{1-W} \leftarrow x_0$
$\lambda \leftarrow \frac{1-\sqrt{\alpha\eta}}{1+\sqrt{\alpha\eta}}$
for $s = 2 - W$ to T **do**
  I) Initialize value for x_{s+W}, y_{s+W}.
  if $s + W \le T$ **then**
    $x_{s+W}^s \leftarrow \Pi_X \left(x_{s+W-1}^{s-1} - \gamma \nabla f_{s+W-1}(x_{s+W-1}^{s-1}) \right)$
    $y_{s+W}^s \leftarrow x_{s+W}^s$
  II) Update $(x_s, y_s), \ldots, (x_{s+W-1}, y_{s+W-1})$ backwards
  for $t = \min(s + W - 1, T) : -1 : \max(s, 1)$ **do**
    $x_t^s \leftarrow \Pi_X \left(y_t^{s-1} - \eta g_t(y_{t-1}^{s-2}, y_t^{s-1}, y_{t+1}^s) \right)$
    $y_t^s = (1+\lambda) x_t^s - \lambda x_t^{s-1}$
Ouputs: x_t^t at time $t = 1, \ldots, T$.

2) Receding Horizon Accelerated Gradient: We continue with our online notation: let x_t^s denote the value of x_t computed at time s, and y_t^s for value of y_t computed at time s.

Similar to RHGD, RHAG also initialize actions at W steps beforehand, and uses OGD outputs as initial actions. The only difference lies in updating rule, which follows from NAG's updating rule (11):

$$\begin{aligned} x_t^s &= \Pi_X \left(y_t^{s-1} - \eta g_t(y_{t-1}^{s-2}, y_t^{s-1}, y_{t+1}^s) \right) \\ y_t^s &= (1+\lambda) x_t^s - \lambda x_t^{s-1} \end{aligned}$$

Remember that x_t^s, y_t^s are kth update; $y_{t-1}^{s-2}, y_t^{s-1}, y_{t+1}^s$ and x_t^{s-1} are all the $k-1$th update for $k = s - t + W$, so this updating rule is identical to offline (11). To make y_{t+1}^s available, we apply the same trick: at each time s, update $y_s, \ldots, y_{s+W}$ backwards.

This relation between NAG and RHAG is summarized in Theorem 3.

Theorem 3. *Given the same stepsize η, let $x_t^{(k)}$ denote the kth update according to NAG, and x_t^s denote the update at time s of RHAG. Notice that x_t^s is the kth update when $s = t-W+k$. If NAG and RHAG shares the same initial values*

$$x_s^{(0)} = x_s^{s-W}, \forall s \in [T]$$

then the output of RHAG is the same as offline NAG after W iterations:

$$x_s^{(W)} = x_s^s, \forall s \in [T].$$

Proof. The main idea of the proof has already been discussed above. We omit the details due to space limit. □

Similar to RHGD, RHAG also evaluates $W+1$ gradients at each time, so it is also much faster than optimization-based algorithms such as RHC.

Remark 1. *The initializing rule in both RHGD and RHAG does not have to be OGD, but we have good theoretical results for OGD and OGD is easy to implement. Generally speaking, any fast online algorithm for prediction-free problem with good theoretical results can be used as initial points.*

V. Performance Guarantee

In this section, we provide upper bounds on dynamic regret and competitive ratio of RHGD and RHAG. We will show that both algorithms' performance improve exponentially with W by each metric. Moreover, RHAG enjoys much better performance than RHGD. To ease analysis, we let $x_0 = 0$ without loss of generality.

A. Dynamic Regret

The theorem below provides upper bounds on RHGD and RHAG's dynamic regret.

Theorem 4. *For any function class $\mathcal{F}_X(\alpha, l, G)$ and any $L_T \in [0, DT]$, given stepsizes $\gamma = 1/l$, $\eta = 1/L$, the dynamic regret of RHGD and RHAG are upper bounded by*

$$\text{Reg}(RHGD, L_T) \leq Q_f \delta (1 - \frac{1}{Q_f})^W L_T \tag{12}$$

$$\text{Reg}(RHAG, L_T) \leq 2\delta (1 - \frac{1}{\sqrt{Q_f}})^W L_T \tag{13}$$

where $\delta = (\beta/l + 1)\frac{G}{(1-\kappa)}$, $\kappa = \sqrt{(1 - \frac{\alpha}{l})}$, $Q_f = \frac{L}{\alpha}$.

Before the proof, we make a few comments on the bounds.

Firstly, notice that the bound in (12) depends linearly on L_T. Thus, when cost functions fluctuate sublinearly, both RHGD and RHAG achieve sublinear regret. Moreover, in Section VI we will show that this dependence is optimal: given a finite lookahead windown W, any online algorithm's dynamic regret is lower bounded by $O(L_T)$.

Secondly, the upper bound decays exponentially fast with the prediction window W. Thus, our online algorithms' performance improves significantly by increasing the lookahead window. This means that our algorithm uses the prediction information effectively.

Finally, since $Q_f > 1$, we have

$$1 - \frac{1}{Q_f} \geq 1 - \frac{1}{\sqrt{Q_f}}$$

so RHAG's dynamic regret decays much faster than RHGD's, especially when Q_f is large. This means that RHAG uses prediction information much more efficiently. We will further show that RHAG provides an optimal way to exploit prediction information in Section VI.

Now we are ready to prove Theorem 4.

Proof of Theorem 4: Let's first prove the bound for RHGD. Applying Theorem 2, we can convert the dynamic regret of RHGD to the objective error of offline GD after W iterations

$$\text{Reg}(RHGD, L_T) = \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} C_1^T(x^{(W)}) - C_1^T(x^*)$$

where $x^{(W)} = \{x_1^{(W)}, \ldots, x_T^{(W)}\}$ are GD outputs after W iterations.

According to convergence rate of offline gradient descent for strongly convex and smooth functions, we have

$$\begin{aligned}&\text{Reg}(RHGD, L_T)\\ &\leq \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \big(C_1^T(x^{(0)}) - C_1^T(x^*)\big) Q_f (1 - \frac{1}{Q_f})^W\end{aligned}$$

In addition, the initial values $x_1^{(0)}, \ldots, x_T^{(0)}$ are the outputs of OGD. As a result,

$$\text{Reg}(OGD, L_T) = \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \big(C_1^T(x^{(0)}) - C_1^T(x^*)\big)$$

Thus, by applying Lemma 1 we have the upper bound for RHGD.

Similarly, for RHAG, the dynamic regret can be bounded by the error bound of offline NAG after W iterations:

$$\begin{aligned}&\text{Reg}(RHAG, L_T)\\ &\leq 2 \sup_{f_t{}_{t=1}^T \in \mathcal{L}_T(L_T)} \big(C_1^T(x^{(0)}) - C_1^T(x^*)\big) (1 - \frac{1}{\sqrt{Q_f}})^W\end{aligned}$$

Apply OGD's regret bound, we prove the upper bound of RHAG's dynamic regret.

B. Upper Bound on Competitive Ratio

The following will show that our online algorithms have constant competitive ratio with respect to T and L_T under a mild assumption on stage costs. Moreover, the bounds decay exponentially with prediction window length W. Similar to dynamic regret, RHAG enjoys much better performance with respect to competitive ratio.

Notice that competitive ratio only makes sense when $f_t(x_t)$ is positive and not close to zero. These requirements are generally true in practice because there is usually a start-up cost or installation cost at each stage. The following assumption is on the existence of start-up costs.

Assumption 1. *For any t, there exists a start-up cost $e > 0$ for each f_t, i.e. $f_t(x_t) \geq e$ for all $t \in [T]$.*

Given Assumption 1, we can show the following bounds on competitive ratio of RHGD's and RHAG.

Theorem 5. *Given Assumption 1, for any function class $\mathcal{F}_X(\alpha, l, G)$ and any $L_T \in [0, DT]$, given stepsizes $\gamma = 1/l$, $\eta = 1/L$, we have*

$$\text{CR}(RHGD, L_T) \leq 1 + \frac{DQ_f\delta}{e}(1 - \frac{1}{Q_f})^W$$

$$\text{CR}(RHAG, L_T) \leq 1 + \frac{2D\delta}{e}(1 - \frac{1}{\sqrt{Q_f}})^W$$

where $\delta = (\beta/l + 1)\frac{G}{(1-\kappa)}$, $\kappa = \sqrt{(1 - \frac{\alpha}{l})}$, $Q_f = \frac{L}{\alpha}$ *and* e *is defined in Assumption 1.*

We make some remarks before the proof. First of all, the upper bounds do not depend on T and L_T, which means that RHGD and RHAG provides a near optimal solution for arbitrarily long horizon and large cost fluctuations.

Moreover, with longer prediction window, the competitive ratio bounds decreases exponentially with W. This demonstrates that our online algorithms utilize prediction effectively.

In addition, it is easy to see that RHAG enjoys better performance than RHGD does, because

$$(1 - \frac{1}{Q_f})^W \geq (1 - \frac{1}{\sqrt{Q_f}})^W$$

since $Q_f \geq 1$.

Next, we give the proof which is based on the results of dynamic regret.

Proof.

$$\begin{aligned}
\text{CR}(RHGD, L_T) &= \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \frac{C_1^T(x^{(W)}) - C_1^T(x^*)}{C_1^T(x^*)} + 1 \\
&\leq \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \frac{Q_f\delta(1 - \frac{1}{Q_f})^W L_T}{C_1^T(x^*)} + 1 \\
&\leq \sup_{\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T)} \frac{Q_f\delta(1 - \frac{1}{Q_f})^W DT}{eT} + 1 \\
&= \frac{Q_f\delta(1 - \frac{1}{Q_f})^W D}{e} + 1
\end{aligned}$$

The first inequality is by taking supremum of numerator and using Theorem 4. The second one is because $L_T \leq DT$ due to bounded action space and Assumption 1.

Similarly, we can prove the bound of $\text{CR}(RHAG, L_T)$. □

VI. Lower Bound

This section studies the performance limit of online algorithms given perfect prediction in windows of size W. For any online algorithm, we show that i) the dynamic regret decays at most exponentially fast with respect to the prediction window W; ii) it is impossible to achieve sublinear regret in our setting given constant W when $L_T \sim O(T)$, which aligns with results in other scenarios [3], [30].

First, we formally define online information and online algorithms[4]. Let I_t denote online information at time t. I_t consists of all past information and future cost prediction:

$$I_t = \{x_0, f_1(\cdot), \ldots, f_{t-1}(\cdot), f_t(\cdot), \ldots, f_{t+W-1}(\cdot)\}$$

Notice that I_t implicitly contains all iterations at time before t, because these iterations can be fully determined by x_0 and cost functions in I_t.

Any online algorithm $\mathcal{A}$ can be characterized by a series of maps $\{\mathcal{A}_t\}_{t=1}^T$ from information sets to X. Specifically, algorithm $\mathcal{A}$ computes an output $x_t^{\mathcal{A}}$ based on map $\mathcal{A}_t$ and current online information I_t for all t:

$$x_t^{\mathcal{A}} = \mathcal{A}_t(I_t), \quad \forall t \in [T] \tag{14}$$

In the following, when we say $\mathcal{A}$ is an online algorithm, we mean it satisfies (14).

Notice that the only requirement imposed by (14) is that it only uses past information and predictable information to compute outputs. This is a constraint that is generally satisfied by any online algorithm, including optimization-based algorithms such as RHC, and gradient-based algorithms proposed in this paper. What we are going to show is that for such a general class, there is a fundamental limit in online performance, and the limit performance is reached by our gradient-based algorithm RHAG.

Theorem 6. *For any* $Q_f > 1$*, let* $\rho = \frac{\sqrt{Q}_f - 1}{\sqrt{Q}_f + 1}$*, when* $T \geq \max(\log_\rho \frac{\sqrt{1-\rho^2}}{2}, 2W)$*, for any* $L_T \in [0, DT]$*, there exists* $\{f_t\}_{t=1}^T \in \mathcal{L}_T(L_T, \mathcal{F}(\alpha, \alpha, \alpha D))$*, where* $\alpha = \frac{4\beta}{Q_f - 1}$*, such that for any* $\mathcal{A}$ *satisfying* (14)*, we have*

$$\text{Reg}(L_T, \mathcal{A}) \geq \frac{\alpha D(1-\rho^2)^2}{128 Q_f}(\frac{\sqrt{Q}_f - 1}{\sqrt{Q}_f + 1})^{2W} L_T$$

We save the proof to appendix and only discuss the interpretation of this lower bound.

First of all, Theorem 6 shows that dynamic regret decays at most exponentially with W, even with arbitrarily large computational power. Compared with Theorem 4, we show that the exponential decay can be achieved by RHGD and RHAG with few computational efforts.

At a closer look, the necessary prediction window size W for RHAG to reach a certain regret R is

$$W \leq O(\sqrt{Q_f}\log(L_T/R))$$

and the lower bound can also be converted to

$$W \geq O((\sqrt{Q_f} - 1)\log(L_T/R))$$

Therefore, RHAG uses prediction information as efficiently as any other algorithms with more computation. In other words, by computing $W + 1$ times, the prediction information is optimally exploited by RHAG.

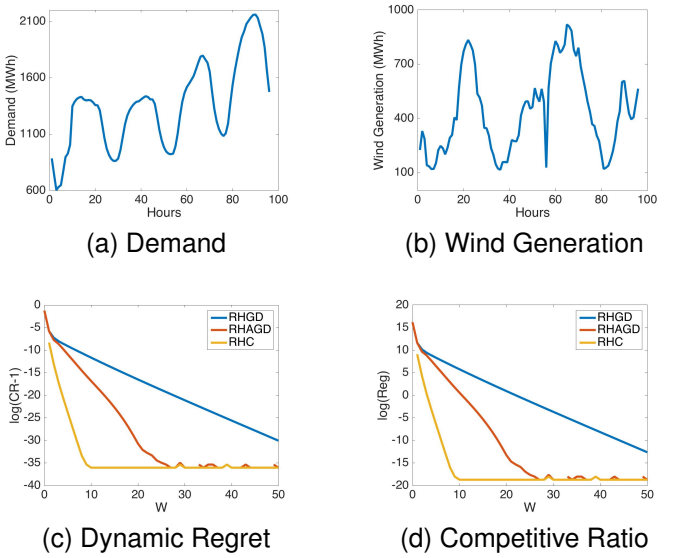
Besides, the lower bound depends linearly on L_T, which is the same as upper bound. This means that when L_T is linear, it is impossible to get sublinear dynamic regret. Similar results have already been discovered in other settings [3], [30]. This also means that RHAG achieves optimal dependency with respect to L_T.

Finally, notice that when $W = 0$, this lower bound reduces to a general lower bound for online setting without prediction. As far as we know, this is also the first result on the fundamental limit for online optimization with switching cost in prediction-free case.

VII. A Numerical Study: Economic Dispatch

This section presents a numerical experiment of RHGD, an economic dispatch problem as introduced in Example 1.

[4]Here we only consider deterministic algorithms, but the results can be easily generalized to stochastic algorithms.



(a) Demand (b) Wind Generation (c) Dynamic Regret (d) Competitive Ratio

Fig. 1. (a) and (b) are demand and wind generation profile from New England ISO. (c) and (d) depicts the dynamic regret and competitive ratio of RHGD and suboptimal MPC in log scale.

TABLE I
RUNNING TIME OF RHGD, RHAG AND RHC

$\mathcal{A}$ \ W	5	15
RHGD	0.0361	0.1038
RHAG	0.0372	0.1060
RHC	33.07	37.21

In the simulation, we consider three conventional generators with quadratic costs given below.

$$c^1(x_{t,1}) = (x_{t,1})^2 + 15x_{t,1} + 10$$
$$c^2(x_{t,2}) = 1.2(x_{t,2})^2 + 10x_{t,2} + 27$$
$$c^3(x_{t,3}) = 1.4(x_{t,3})^2 + 6x_{t,3} + 21$$

Besides, we consider a high-penetration of wind supply as shown in Figure 1 (b) where the data is from [31]. Figure 1 (a) depicts the load profile of New Hampshire from June 9 to June 12 in 2017 [32]. For simplicity, we let $\xi_t = \xi = 1.2$ and $\beta = 1$. In the simulation, each t corresponds to an hour.

Figure 1 (c) and (d) respectively present the dynamic regret and the competitive ratio of RHGD RHAG and RHC in log scale as a function of prediction window W. Notice that when $W = 0$, i.e. without prediction, RHGD and RHAG reduces to classic OGD. When W increase, all three algorithms decay linearly in log scale, which demonstrates exponential decay rate. Moreover, RHAG and RHC enjoys higher decay rate than RHGD, which matches our theoretical upper bounds. Finally, to reach the same dynamic regret, RHAG takes about 2 times larger W than that RHC does, which matches our analysis that RHAG exploits prediction information optimally.

Table I compares the computational time of RHGD RHAG and RHC for $W = 5, 15$. Notice that RHGD and RHAG are significantly faster than RHC, which is intuitive because RHGD and RHAG are gradient based while RHC is optimization based.

VIII. CONCLUSION

In this paper we study online convex optimization problems with switching costs and propose two computational efficient online algorithms, RHGD and RHAG. Our online algorithms only use W steps of prediction and only needs $W + 1$ steps of gradient evaluation at each time step. We show that the dynamic regret and the competitive ratio of RHGD and RHAG decay exponentially fast with the prediction window W. Moreover, RHAG's decaying rate matches that of a lower bound of general online algorithms, meaning that RHAG exploits prediction information optimally by using few computational efforts.

There are many interesting future directions, such as i) generalizing the method to handle imperfect prediction, and ii) designing and studying other computational efficient online algorithms such as suboptimal MPC.

APPENDIX

A. Proof of Lemma 1

Proof. Remember that $C_1^T(x) = \sum_t f_t(x_t) + \frac{\beta}{2}\|x_t - x_{t-1}\|^2$. Since $\sum_t f_t(x_t)$ is α-strongly convex and smooth, we only need to study $\frac{\beta}{2}\|x_t - x_{t-1}\|^2$. The Hessian of switching cost is

$$A = \beta \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

and $0 \leq A \leq 4\beta I$. So $L = l + 4\beta$, and $C_1^T(x)$ is α-strongly convex. □

B. Proof of Lemma 1

Before the formal proof, we introduce a supporting lemma, which upper bounds the switching cost of OGD outputs.

Lemma 2. *Given $f_t \in \mathcal{F}_X(\alpha, l, G)$ for $t = 1, \ldots, T$, and stepsize $\frac{1}{l}$, the outputs of OGD $\{x_t\}_{t=1}^T$ satisfy*

$$\sum_{t=1}^{T} \|x_t - x_{t-1}\|^2 \leq \frac{2G}{l(1-\kappa)} \sum_{t=1}^{T} \|\theta_t - \theta_{t-1}\|$$

where x_1 is chosen to be $x_1 = x_0$, $\kappa = \sqrt{(1 - \frac{\alpha}{l})}$, $\theta_t := \arg\min_{x_t \in X} f_t(x_t)$ for $1 \leq t \leq T$ and $\theta_0 = x_0$.

Proof. Firstly, given $x_1 = x_0$, we have

$$\sum_{t=1}^{T} \|x_t - x_{t-1}\|^2 = \sum_{t=2}^{T} \|x_{t+1} - x_t\|^2$$

According to Corollary 2.3.2 in [33]

$$\sum_{t=1}^{T-1} \|x_{t+1} - x_t\|^2 \leq \frac{2}{l} \sum_{t=1}^{T-1} f_t(x_t) - f_t(\theta_t) \tag{15}$$

Since $f_t \in \mathcal{F}_X(\alpha, l, G)$, and $\|x_T - \theta_T\| \geq 0$,

$$\sum_{t=1}^{T-1} f_t(x_t) - f_t(\theta_t) \leq G \sum_{t=1}^{T-1} \|x_t - \theta_t\| \leq G \sum_{t=1}^{T} \|x_t - \theta_t\| \tag{16}$$

The remainder of the proof is to bound $\sum_{t=1}^{T}\|x_t-\theta_t\|$. By triangle inequality of Euclidean norm,

$$\sum_{t=1}^{T}\|x_t-\theta_t\|\le\sum_{t=1}^{T}(\|x_t-\theta_{t-1}\|+\|\theta_t-\theta_{t-1}\|) \tag{17}$$

From Theorem 2.3.4 in [33], we have

$$\sum_{t=2}^{T}\|x_t-\theta_{t-1}\|\le\kappa\sum_{t=1}^{T-1}\|x_t-\theta_t\|$$

where $\kappa=\sqrt{(1-\frac{\alpha}{\gamma})}$. Plug this in (17), we have

$$\begin{aligned}\sum_{t=1}^{T}\|x_t-\theta_t\|&\le\kappa\sum_{t=1}^{T-1}\|x_t-\theta_t\|+\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\|\\&\le\kappa\sum_{t=1}^{T}\|x_t-\theta_t\|+\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\|\end{aligned} \tag{18}$$

where the first inequality is due to $x_1=y_0^*=x_0$, and the second one is due to $\|x_T-\theta_T\|\ge 0$.

Regrouping the terms in (18) gives us:

$$\sum_{t=1}^{T}\|x_t-\theta_t\|\le\frac{1}{1-\kappa}\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\| \tag{19}$$

This inequality together with (15) and (16) proves the bound in the lemma. □

Now we are ready to prove Lemma 1.

Proof of Lemma 1: For any $\{f_t\}_{t=1}^{T}\in\mathcal{L}_T(L_T)$, we have

$$\begin{aligned}&\sum_{t=1}^{T}\big(f_t(x_t)-f_t(x_t^*)+\beta/2\|x_t-x_{t-1}\|^2\big)\\&\le\sum_{t=1}^{T}\big(f_t(x_t)-f_t(\theta_t)+\beta/2\|x_t-x_{t-1}\|^2\big)\\&\le G\sum_{t=1}^{T}\|x_t-\theta_t\|+\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\|\frac{G\beta}{l(1-\kappa)}\\&\le\frac{G}{(1-\kappa)}\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\|+\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\|\frac{G\beta}{l(1-\kappa)}\\&=(\beta/l+1)\frac{G}{(1-\kappa)}\sum_{t=1}^{T}\|\theta_t-\theta_{t-1}\|\\&\le(\beta/l+1)\frac{G}{(1-\kappa)}L_T\end{aligned}$$

The first inequality is by throwing away negative term $-\beta/2\|x_t^*-x_{t-1}^*\|^2$. The second one is because θ_t minimizes $f_t(x_t)$. The third one is from bounded gradient and Lemma 2. The last one is by (19) and combining like terms. □

C. Proof of Theorem 6

The proof of Theorem 6 relies on the following Lemma. It shows that, in the optimal offline solution, the dependence of the action x_t on the future y_{t+W}^* decays at most exponentially fast in W.

Lemma 3. *Given $y_t^*\in[-1,1], t\in[T]$, the optimal offline solution can be written as $\mathbf{x}^*=A\mathbf{y}^*$ where $\mathbf{y}^*=[y_1^*,y_2^*\ldots,y_T^*]'$, and A is a T-by-T symmetric matrix. We write $A=[\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_T]'$, and $\mathbf{a}_t'=[a_{t,1},a_{t,2},\ldots,a_{t,T}]$. Then, for $\tau\ge 0$, $a_{t,t+\tau}$ satisfies*

$$a_{t,t+\tau}\ge\frac{1}{\sqrt{4\frac{\beta}{\alpha}+1}}\rho^{\tau}(1-\rho^2-2\rho^{2T}). \tag{20}$$

Proof. Recall the offline optimization problem is given by,

$$\min_{x\in\mathbb{R}^T}\sum_{t=1}^{T}\left[\frac{\alpha}{2}(x_t-v_t)^2+\frac{\beta}{2}(x_t-x_{t-1})^2\right] \tag{21}$$

where $x_1,\ldots,x_T$ are the decision variables and $x_0=0$. Taking derivative of the cost function, we have (defining $\xi=\frac{\beta}{\alpha}$),

$$\begin{cases}x_t-v_t+\theta(x_t-x_{t-1})+\theta(x_t-x_{t+1})=0, & t\in[T-1]\\ x_T-v_T+\theta(x_T-x_{T-1})=0\end{cases} \tag{22}$$

Define

$$H=\begin{bmatrix}1+2\xi&-\xi&0&0&\cdots&0\\-\xi&1+2\xi&-\xi&0&\cdots&0\\0&-\xi&1+2\xi&-\xi&\cdots&0\\\vdots&\ddots&\ddots&\ddots&\ddots&\vdots\\0&\cdots&0&-\xi&1+2\xi&-\xi\\0&\cdots&0&0&-\xi&1+\xi\end{bmatrix}. \tag{23}$$

Then, (22) can be rewritten as $H\mathbf{x}=\mathbf{y}^*$. Since H is diagonally dominant, it is invertible. Also since, H's diagonal entries are positive, while the off-diagonal entries are non-positive, we have H^{-1} is a nonnegative matrix. Let $A=H^{-1}$. Clearly A is symmetric since H is symmetric. Then we have,

$$\mathbf{x}^*=A\mathbf{y}^*. \tag{24}$$

It remains to prove the lower bound on the entries of A. Define $\tilde{A}=[\tilde{a}_{i,j}]$ to be a T-by-T matrix, where $\tilde{a}_{t,t+\tau}=c\rho^{|\tau|}$ (for $t=1,\ldots,T$, $\tau=1-t,\ldots,T-t$). In what follows, we calculate $H\tilde{A}$. Firstly, for the second row till the second to last row, i.e. for $2\le t\le T-1$, $1-t\le\tau\le T-t$

$$\begin{aligned}&[H\tilde{A}]_{t,t+\tau}=-\xi\tilde{a}_{t-1,t+\tau}+(1+2\xi)\tilde{a}_{t,t+\tau}-\xi\tilde{a}_{t+1,t+\tau}\\&=c[(1+2\xi)\rho^{|\tau|}-\xi\rho^{|\tau+1|}-\xi\rho^{|\tau-1|}]\\&=\begin{cases}c\rho^{\tau}[(1+2\xi)-\xi\rho-\xi\frac{1}{\rho}]=0 & \text{when } \tau\ge 1\\ c\rho^{-\tau}[(1+2\xi)-\xi\rho-\xi\frac{1}{\rho}]=0 & \text{when } \tau\le -1\\ c[(1+2\xi)-2\xi\rho]=1 & \text{when } \tau=0\end{cases}\end{aligned}$$

Then, we calculate the first row of $H\tilde{A}$. When $t=1$, $0\le\tau\le T-1$,

$$\begin{aligned}&[H\tilde{A}]_{1,1+\tau}=(1+2\xi)\tilde{a}_{1,1+\tau}-\xi\tilde{a}_{2,1+\tau}\\&=(1+2\xi)c\rho^{|\tau|}-\xi c\rho^{|\tau-1|}\\&=\begin{cases}c\rho^{\tau}(1+2\xi-\xi\frac{1}{\rho}) & \text{when } \tau\ge 1\\ c(1+2\xi-\xi\rho) & \text{when } \tau=0\end{cases}\\&=\begin{cases}\rho^{\tau}[\frac{1}{2}(\frac{1}{\sqrt{4\xi+1}}-1)+\frac{\xi}{\sqrt{4\xi+1}}]=\rho^{\tau}\rho\xi c & \text{when } \tau\ge 1\\ \frac{1}{2}(\frac{1}{\sqrt{4\xi+1}}+1)+\frac{\xi}{\sqrt{4\xi+1}}=1+\rho\xi c & \text{when } \tau=0\end{cases}\end{aligned}$$

For the last row, we have when $t=T$, $1-T\le\tau\le 0$, we have

$$\begin{aligned}
&[H\tilde{A}]_{T,T+\tau} = (1+\xi)\tilde{a}_{T,T+\tau} - \xi\tilde{a}_{T-1,T+\tau}\\
&= (1+\xi)c\rho^{|\tau|} - \xi c\rho^{|\tau+1|}\\
&= \begin{cases} c\rho^{-\tau}(1+\xi-\xi\frac{1}{\rho}) & \text{when } \tau\le -1\\ c(1+\xi-\xi\rho) & \text{when } \tau = 0\end{cases}\\
&= \begin{cases} \rho^{-\tau}\frac{1}{2}(\frac{1}{\sqrt{4\xi+1}}-1) = \rho^{-\tau}\frac{1}{2}(c-1) & \text{when } \tau\le -1\\ \frac{1}{2}(\frac{1}{\sqrt{4\xi+1}}+1) = \frac{1}{2}(c+1) & \text{when } \tau = 0\end{cases}
\end{aligned}$$

Therefore, we have

$$I - H\tilde{A} = \begin{bmatrix} -\xi c\rho & -\xi c\rho^2 & \cdots & -\xi c\rho^T\\ 0 & 0 & 0 & \cdot\\ \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdot & 0\\ \frac{1}{2}(1-c)\rho^{T-1} & \frac{1}{2}(1-c)\rho^{T-2} & \cdots & \frac{1}{2}(1-c)\end{bmatrix}$$

Recall the t'th row of A is $\mathbf{a}_t'$. We also let the t'th row of $\tilde{A}$ be $\tilde{\mathbf{a}}_t'$. Let $\mathbf{e}_t$ be the T-dimensional vector with the t'th entry being 1 and other entries being 0. Then, $\mathbf{a}_t = A\mathbf{e}_t$ and $\tilde{\mathbf{a}}_t = \tilde{A}\mathbf{e}_t$. Then, we have,

$$\begin{aligned}
\mathbf{a}_t - \tilde{\mathbf{a}}_t &= A\mathbf{e}_t - \tilde{A}\mathbf{e}_t = A(I-H\tilde{A})\mathbf{e}_t\\
&= A[-\xi c\rho^t, 0, \cdots, 0, \frac{1}{2}(1-c)\rho^{T-t}]'\\
&= -\xi c\rho^t\mathbf{a}_1 + \frac{1}{2}(1-c)\rho^{T-t}\mathbf{a}_T \qquad (25)
\end{aligned}$$

Now we set $t=1$ and $t=T$ in (25), and get

$$\mathbf{a}_1 - \tilde{\mathbf{a}}_1 = -\xi c\rho\mathbf{a}_1 + \frac{1}{2}(1-c)\rho^{T-1}\mathbf{a}_T \qquad (26)$$

$$\mathbf{a}_T - \tilde{\mathbf{a}}_T = -\xi c\rho^T\mathbf{a}_1 + \frac{1}{2}(1-c)\mathbf{a}_T \qquad (27)$$

Manipulating (27), and we can get

$$\mathbf{a}_T = \frac{2}{1+c}\tilde{\mathbf{a}}_T - \rho^T\frac{2\xi c}{1+c}\mathbf{a}_1 \le \frac{2}{1+c}\tilde{\mathbf{a}}_T \qquad (28)$$

where we have used the fact that all the entries of $\mathbf{a}_1$ are nonnegative. Manipulating (26), using $1+\rho\xi c = \frac{\xi c}{\rho}$ and plugging (28) into (26), we have,

$$\begin{aligned}
\mathbf{a}_1 &= \frac{1}{1+c\xi\rho}\tilde{\mathbf{a}}_1 + \frac{1-c}{2(1+c\xi\rho)}\rho^{T-1}\mathbf{a}_T\\
&= \frac{\rho}{\xi c}\tilde{\mathbf{a}}_1 + (1-c)\frac{1}{2\xi c}\rho^T\mathbf{a}_T\\
&\le \frac{\rho}{\xi c}\tilde{\mathbf{a}}_1 + \frac{1-c}{1+c}\frac{1}{\xi c}\rho^T\tilde{\mathbf{a}}_T \qquad (29)
\end{aligned}$$

Plugging the above into (28),

$$\begin{aligned}
\mathbf{a}_T &= \frac{2}{1+c}\tilde{\mathbf{a}}_T - \rho^T\frac{2\xi c}{1+c}\mathbf{a}_1\\
&\ge \frac{2}{1+c}\tilde{\mathbf{a}}_T - \rho^T\frac{2\xi c}{1+c}(\frac{\rho}{\xi c}\tilde{\mathbf{a}}_1 + \frac{1-c}{1+c}\frac{1}{\xi c}\rho^T\tilde{\mathbf{a}}_T)\\
&= \frac{2}{1+c}\tilde{\mathbf{a}}_T - \frac{2}{1+c}\rho^{T+1}\tilde{\mathbf{a}}_1 - \frac{1-c}{1+c}\frac{2}{1+c}\rho^{2T}\tilde{\mathbf{a}}_T\\
&\ge \frac{2}{1+c}\tilde{\mathbf{a}}_T - \frac{2}{1+c}\rho^{T+1}\tilde{\mathbf{a}}_1 - \frac{2}{1+c}\rho^{2T}\tilde{\mathbf{a}}_T \qquad (30)
\end{aligned}$$

Using the upper bound on $\mathbf{a}_1$ (29) and the lower bound on $\mathbf{a}_T$ (30) and plugging them into (25), we have

$$\begin{aligned}
&\mathbf{a}_t - \tilde{\mathbf{a}}_t = -\xi c\rho^t\mathbf{a}_1 + \frac{1}{2}(1-c)\rho^{T-t}\mathbf{a}_T\\
&\ge -\xi c\rho^t[\frac{\rho}{\xi c}\tilde{\mathbf{a}}_1 + \frac{1-c}{1+c}\frac{1}{\xi c}\rho^T\tilde{\mathbf{a}}_T]\\
&+ \frac{1}{2}(1-c)\rho^{T-t}[\frac{2}{1+c}\tilde{\mathbf{a}}_T - \frac{2}{1+c}\rho^{T+1}\tilde{\mathbf{a}}_1 - \frac{2}{1+c}\rho^{2T}\tilde{\mathbf{a}}_T]\\
&= -\rho^{t+1}\tilde{\mathbf{a}}_1 - \frac{1-c}{1+c}\rho^{t+T}\tilde{\mathbf{a}}_T\\
&+ \frac{1-c}{1+c}\rho^{T-t}\tilde{\mathbf{a}}_T - \frac{1-c}{1+c}\rho^{2T-t+1}\tilde{\mathbf{a}}_1 - \frac{1-c}{1+c}\rho^{3T-t}\tilde{\mathbf{a}}_T\\
&\ge -\rho^{t+1}\tilde{\mathbf{a}}_1 - \rho^{2T-t+1}\tilde{\mathbf{a}}_1 - \rho^{3T-t}\tilde{\mathbf{a}}_T
\end{aligned}$$

Therefore, for $1\le t\le T$ and $0\le\tau\le T-t$,

$$\begin{aligned}
&a_{t,t+\tau}\\
&\ge c\rho^\tau - \rho^{t+1}c\rho^{t+\tau-1} - \rho^{2T-t+1}c\rho^{t+\tau-1} - \rho^{3T-t}c\rho^{T-t-\tau}\\
&= c\rho^\tau - c\rho^{2t+\tau} - c\rho^{2T+\tau} - c\rho^{4T-2t-\tau}\\
&= c\rho^\tau(1-\rho^{2t}-\rho^{2T}-\rho^{4T-2t-2\tau})\\
&\ge c\rho^\tau(1-\rho^2-2\rho^{2T})
\end{aligned}$$

where in the last inequality we have used $\rho^{2t}\le\rho^2$, and by $t+\tau\le T$, $\rho^{4T-2t-2\tau}\le\rho^{2T}$. □

We are now ready to prove Theorem 6.

Proof of Theorem 6:

Proof. All we need to do is to construct a cost function list in $\mathcal{L}^T$, such for any $x_t^{\mathcal{A}} = \mathcal{A}_t(I_t)$,

$$C_1^T(x^{\mathcal{A}}) - C_1^T(x^*) \ge \frac{\alpha^2 D(1-\rho^2)^2}{128L}(\frac{\sqrt{Q}_f-1}{\sqrt{Q}_f+1})^{2W}L_T$$

WLOG, we consider $X=[-1/2,1/2]$, $D=1$ and $n=1$, $x_0=0$. Also, we assume L_T is integer. (If it is not, slightly adjust the range of θ_t from $[-1/2,1/2]$).

Define $\Delta = \lceil T/L_T\rceil$. Divide T into L_T parts:

$$1,\ldots,\Delta,\Delta+1,\ldots,2\Delta,\ldots,(L_T-1)\Delta+1,\ldots,T$$

each part has Δ stages, except that the last stage may have less stages.

Consider cost function in this form: $f_t(x_t) = \frac{\alpha}{2}(x_t-\theta_t)^2$. It is easy to see $f_t(x_t)\in\mathcal{F}(\alpha,\alpha)$. Generate θ_t in this way: for any $t=k\Delta_T+1$ for $k=0,\ldots,L_T-1$, let θ_t i.i.d. from Bernoulli distribution with $\Pr(\theta_t=1/2)=\Pr(\theta_t=-1/2)=1/2$. Then for all the following θ_t in the same subsequence, let them be the same with the first one in this subsequence, i.e.

$$\theta_t = \theta_{k\Delta_T+1},\ \text{for } k\Delta_T+1\le t\le(k+1)\Delta_T, k=0,\ldots,L_T-1.$$

Now, path length satisfies:

$$\sum_{t=1}^T\|\theta_t-\theta_{t-1}\| = \sum_{k=0}^{L_T-1}\|\theta_{k\Delta_T+1}-\theta_{(k-1)\Delta_T+1}\le L_T$$

where $x_{1-\Delta}=x_0=0$. So $\{f_t(\cdot)\}_{t=1}^T\in\mathcal{L}^T(L_T)$.

For $t\le T-W$, define $\mathbf{b}_t=[a_{t,1},\ldots,a_{t,t+W-1},0,\ldots]'$; for $t\ge T-W+1$, define $\mathbf{b}_t=\mathbf{a}_t$. Define $\mathbf{c}_t=\mathbf{a}_t-\mathbf{b}_t$ ($\mathbf{c}_t'$

is the t'th row of A with the first $t+W-1$ entries being set to 0), and $C = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_T]'$.

Denote $\mathcal{T} = \{1 \le t \le T-W | t \equiv 1 \pmod{\Delta}\}$. We then have for $t \in \mathcal{T}$,

$$\begin{aligned}(x_t - x_t^*)^2 &= (x_t - \mathbf{b}_t'\theta - \mathbf{c}_t'\theta)^2 \\ &= (x_t - \mathbf{b}_t'\theta)^2 + (\mathbf{c}_t'\theta)^2 - 2(\mathbf{c}_t'\theta)(x_t - \mathbf{b}_t'\theta) \\ &\ge (\mathbf{c}_t'\theta)^2 + 2(\mathbf{c}_t'\theta)(\mathbf{b}_t'\theta - x_t)\end{aligned}$$

We take expectation conditioned on I_t for $t \in \mathcal{T}$,

$$\begin{aligned}&\mathbb{E}[(x_t - x_t^*)^2 | I_t] \\ &\overset{(a)}{\ge} \mathbb{E}[(\mathbf{c}_t'\theta)^2 | I_t] \\ &\quad + 2\mathbb{E}[(\mathbf{c}_t'\theta)|I_t](\mathbf{b}_t'\theta - x_t) \\ &\overset{(b)}{\ge} \mathbb{E}[(\mathbf{c}_t'\theta)^2 | I_t] + 2\mathbb{E}[(\mathbf{c}_t'\theta)](\mathbf{b}_t'\theta - x_t) \\ &\overset{(c)}{=} \mathbb{E}[(\mathbf{c}_t'\theta)^2 | I_t]\end{aligned}$$

where (a) is because $\mathbf{b}_t'\theta$ and x_t are constant given I_t; (b) is because $\mathbf{c}_t'\theta$ only depends on $\theta_{t+W}^*, \ldots, \theta_T^*$, which are independent from I_t when $t \equiv 1 \pmod{\Delta}$,; (c) is because θ_τ^* has expectation 0 for any τ. Now we take expectation again over I_t, and get $\mathbb{E}[(x_t - x_t^*)^2] \ge \mathbb{E}[(\mathbf{c}_t'\theta)^2]$.

Summing over t,

$$\begin{aligned}\mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 &\ge \sum_{t\in\mathcal{T}} \|x_t - x_t^*\|^2 \\ &\ge \sum_{t\in\mathcal{T}} \mathbb{E}(\mathbf{c}_t'\theta)^2 = \mathbb{E}\,\theta' R \theta\end{aligned}$$

where $R = (\sum_{t\in\mathcal{T}} \mathbf{c}_t\mathbf{c}_t')$. Now since R is a T-by-T symmetric positive semi-definite matrix, we diagonalize it to get $R = \sum_{t=1}^T \lambda_t \boldsymbol{\zeta}_t \boldsymbol{\zeta}_t'$, where λ_t is a nonegative eigenvalue of R associated with eigenvector $\boldsymbol{\zeta}_t$, and $\boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_T$ form a orthonormal basis of $\mathbb{R}^T$. Thus,

$$\mathbb{E}\,\theta' R \theta = \sum_{t=1}^T \mathbb{E}(\lambda_t \theta' \boldsymbol{\zeta}_t \boldsymbol{\zeta}_t' \theta) = \sum_{t=1}^T \lambda_t \mathbb{E}(\boldsymbol{\zeta}_t'\theta)^2$$

Let $\boldsymbol{\zeta}_t = [\zeta_{t,1}, \ldots, \zeta_{t,T}]'$, then $\mathbb{E}(\boldsymbol{\zeta}_t'\theta)^2 = \mathrm{Var}(\boldsymbol{\zeta}_t'\theta) = \sum_{\tau=1}^t \zeta_{t,\tau}^2 \mathrm{Var}(\theta_t) = 1/4$. Therefore, $\mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 \ge \sum_{t=1}^T \lambda_t/4 = \mathrm{Trace}(R)/4 = \sum_{t\in\mathcal{T}} \|\mathbf{c}_t\|^2/4$. By the definition of $\mathbf{c}_t$, and (20), for $t \le T-W$,

$$\|\mathbf{c}_t\|^2 \ge a_{t,t+W}^2 \ge \frac{\alpha(1-\rho^2)^2}{4(4\beta+\alpha)}\rho^{2W}$$

where we have used by the lower bound on T, $2\rho^{2T} \le \frac{1-\rho^2}{2}$. As a result,

$$\mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 \ge |\mathcal{T}| \frac{\alpha(1-\rho^2)^2}{16(4\beta+\alpha)}\rho^{2W}.$$

Let's compute $|\mathcal{T}|$.

$$\begin{aligned}|\mathcal{T}| &= \lceil \frac{T-W}{\lceil T/L_T \rceil} \rceil \ge \lceil \frac{T-W}{T/L_T + 1} \rceil \\ &\ge L_T \frac{T-W}{T+L_T} \ge L_T \frac{T-W}{2T}\end{aligned}$$

The last inequality is because $L_T \le T$ when $D = [-1/2, 1/2]$.

Therefore, there must exist a realization of θ, s.t.

$$\begin{aligned}\|\mathbf{x} - \mathbf{x}^*\|^2 &\ge (T-W)\frac{\alpha(1-\rho^2)^2}{4(4\beta+\alpha)}\rho^{2W} \\ &\ge \frac{T-W}{T}\frac{\alpha(1-\rho^2)^2}{32(4\beta+\alpha)}\rho^{2W} L_T\end{aligned}$$

The results of the theorem follows directly from the α-strong convexity of the off-line cost function and $T \ge 2W$. □

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] E. Hazan, *Introduction to Online Convex Optimization*, ser. Foundations and Trends(r) in Optimization Series. Now Publishers, 2016. [Online]. Available: https://books.google.com/books?id=IFxLvgAACAAJ

[2] A. Mokhtari, S. Shahrampour, A. Jadbabaie, and A. Ribeiro, "Online optimization in dynamic environments: Improved regret rates for strongly convex problems," in *Decision and Control (CDC), 2016 IEEE 55th Conference on*. IEEE, 2016, pp. 7195–7201.

[3] L. Andrew, S. Barman, K. Ligett, M. Lin, A. Meyerson, A. Roytman, and A. Wierman, "A tale of two metrics: Simultaneous bounds on competitiveness and regret," in *Conference on Learning Theory*, 2013, pp. 741–763.

[4] E. C. Hall and R. M. Willett, "Online convex optimization in dynamic environments," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 647–662, 2015.

[5] B. Narayanaswamy, V. K. Garg, and T. Jayram, "Online optimization for the smart (micro) grid," in *Proceedings of the 3rd international conference on future energy systems: where energy, computing and communication meet*. ACM, 2012, p. 19.

[6] M. Moeini-Aghtaie, P. Dehghanian, M. Fotuhi-Firuzabad, and A. Abbaspour, "Multiagent genetic algorithm: an online probabilistic view on economic dispatch of energy hubs constrained by wind availability," *IEEE Transactions on Sustainable Energy*, vol. 5, no. 2, pp. 699–708, 2014.

[7] T. Chen, Q. Ling, and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE Transactions on Signal Processing*, 2017.

[8] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *Green Computing Conference (IGCC), 2012 International*. IEEE, 2012, pp. 1–10.

[9] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 5, pp. 1378–1391, 2013.

[10] L. Gan, A. Wierman, U. Topcu, N. Chen, and S. H. Low, "Real-time deferrable load control: handling the uncertainties of renewable generation," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 3, pp. 77–79, 2014.

[11] S.-J. Kim and G. B. Giannakis, "Real-time electricity pricing for demand response using online convex optimization," in *Innovative Smart Grid Technologies Conference (ISGT), 2014 IEEE PES*. IEEE, 2014, pp. 1–5.

[12] V. Joseph and G. de Veciana, "Jointly optimizing multi-user rate adaptation for video transport over wireless systems: Mean-fairness-variability tradeoffs," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 567–575.

[13] F. Zanini, D. Atienza, G. De Micheli, and S. P. Boyd, "Online convex optimization-based algorithm for thermal management of mpsocs," in *Proceedings of the 20th symposium on Great lakes symposium on VLSI*. ACM, 2010, pp. 203–208.

[14] M. Tanaka, "Real-time pricing with ramping costs: A new approach to managing a steep change in electricity demand," *Energy Policy*, vol. 34, no. 18, pp. 3634–3643, 2006.

[15] R. Mookherjee, B. F. Hobbs, T. L. Friesz, and M. A. Rigdon, "Dynamic oligopolistic competition on an electric power network with ramping costs and joint sales constraints," *J. Ind. Manag. Optim*, vol. 4, no. 3, pp. 425–452, 2008.

[16] https://www.misoenergy.org/MarketsOperations/RealTimeMarketData\\/Pages/DayAheadWindForecast.aspx.

[17] http://www.pjm.com/planning/resource-adequacy-planning/load-forecast-dev-process.aspx.

[18] A. Rakhlin and K. Sridharan, "Online learning with predictable sequences," in *Conference on Learning Theory*, 2013, pp. 993–1019.

[19] A. Jadbabaie, A. Rakhlin, S. Shahrampour, and K. Sridharan, "Online optimization: Competing with dynamic comparators," in *Artificial Intelligence and Statistics*, 2015, pp. 398–406.

[20] D. Angeli, R. Amrit, and J. B. Rawlings, "On average performance and stability of economic model predictive control," *IEEE transactions on automatic control*, vol. 57, no. 7, pp. 1615–1626, 2012.

[21] L. Grüne and M. Stieler, "Asymptotic stability and transient optimality of economic mpc without terminal conditions," *Journal of Process Control*, vol. 24, no. 8, pp. 1187–1196, 2014.

[22] A. Ferramosca, D. Limon, and E. F. Camacho, "Economic mpc for a changing economic criterion for linear systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 10, pp. 2657–2667, 2014.

[23] M. Kögel and R. Findeisen, "Stabilization of inexact mpc schemes," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, 2014, pp. 5922–5928.

[24] K. Graichen and A. Kugi, "Stability and incremental improvement of suboptimal mpc without terminal constraints," *IEEE Transactions on Automatic Control*, vol. 55, no. 11, pp. 2576–2580, 2010.

[25] D. A. Allan, C. N. Bates, M. J. Risbeck, and J. B. Rawlings, "On the inherent robustness of optimal and suboptimal nonlinear mpc," *Systems & Control Letters*, vol. 106, pp. 68–78, 2017.

[26] J. Rawlings and D. Mayne, "Postface to model predictive control: Theory and design," *Nob Hill Pub*, pp. 155–158, 2012.

[27] M. Badiei, N. Li, and A. Wierman, "Online convex optimization with ramp constraints," in *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*. IEEE, 2015, pp. 6730–6736.

[28] N. Chen, J. Comden, Z. Liu, A. Gandhi, and A. Wierman, "Using predictions in online optimization: Looking forward with an eye on the past," in *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*. ACM, 2016, pp. 193–206.

[29] R. Rosales and S. Sclaroff, "Improved tracking of multiple humans with trajectory prediction and occlusion modeling," Boston University Computer Science Department, Tech. Rep., 1998.

[30] O. Besbes, Y. Gur, and A. Zeevi, "Non-stationary stochastic optimization," *Operations research*, vol. 63, no. 5, pp. 1227–1244, 2015.

[31] https://www.iso-ne.com/isoexpress/web/reports/operations/\\/tree/daily-gen-fuel-type.

[32] https://www.iso-ne.com/isoexpress/web/reports/load-\\and-demand/-/tree/zone-info.

[33] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.