Bregman-style Online Convex Optimization with Energy Harvesting Constraints

KAMIAR ASGARI* and MICHAEL J. NEELY*, University of Southern California

This paper considers online convex optimization (OCO) problems where decisions are constrained by available energy resources. A key scenario is optimal power control for an energy harvesting device with a finite capacity battery. The goal is to minimize a time-average loss function while keeping the used energy less than what is available. In this setup, the distribution of the randomly arriving harvestable energy (which is assumed to be i.i.d.) is unknown, the current loss function is unknown, and the controller is only informed by the history of past observations. A prior algorithm is known to achieve $O(\sqrt{T})$ regret by using a battery with an $O(\sqrt{T})$ capacity. This paper develops a new algorithm that maintains this asymptotic trade-off with the number of time steps T while improving dependency on the dimension of the decision vector from $O(\sqrt{n})$ to $O(\sqrt{\log(n)})$. The proposed algorithm introduces a separation of the decision vector into amplitude and direction components. It uses two distinct types of Bregman divergence, together with energy queue information, to make decisions for each component.

CCS Concepts: • **Networks** → *Network resources allocation*; • **Theory of computation** → **Online learning algorithms**; *Convergence and learning in games.*

Additional Key Words and Phrases: Online learning; mirror descent; wireless networks; scheduling

ACM Reference Format:
Kamiar Asgari and Michael J. Neely. 2020. Bregman-style Online Convex Optimization with Energy Harvesting Constraints. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 3, Article 52 (December 2020), 25 pages. https://doi.org/10.1145/3428337

1 INTRODUCTION

Consider a system that draws energy from a battery and allocates it over time to n different subsystems. The system operates in slotted time over a fixed time horizon $t \in \{1, 2, \ldots, T\}$, where T is a given positive integer. Let $X_t = [X_t(1), \ldots, X_t(n)]$ denote the decision vector on time slot t, where $X_t(i)$ is the amount of energy allocated to subsystem $i \in \{1, \ldots, n\}$. The decision vector X_t incurs a loss $L_t(X_t)$ for slot t, where $L_t(\cdot)$ is a convex but unknown function that shall be called a *loss function*. The loss function models the penalty and/or utility associated with the decision X_t (utility can be defined as -1 times the loss). There are three challenges in choosing the decision vector X_t:

- The convex loss function $L_t(\cdot)$ is unknown at the start of slot t. The X_t decision is made without knowledge of this function. The corresponding loss $L_t(X_t)$ is only revealed after the

*Both authors contributed equally to this research.

Authors' address: Kamiar Asgari, Kamiaras@usc.edu; Michael J. Neely, Mjneely@usc.edu, University of Southern California, Los Angeles, California, 90007.

2476-1249/2020/12-ART52 $15.00
https://doi.org/10.1145/3428337

X_t decision is made. Further, $L_t(\cdot)$ can vary arbitrarily over the time horizon $t \in \{1, \ldots, T\}$ (with no associated probability model).
- The decision vector X_t is constrained by

$$\sum_{i=1}^{n} X_t(i) \leq B_{t-1} + E_t \tag{1}$$

where B_{t-1} is the amount of energy currently available in the battery and E_t is the random *energy arrival* that can either be used or harvested on slot t.
- The battery energy evolves according to the following queue update equation:

$$B_t = \min\left\{B_{t-1} - \sum_{i=1}^{n} X_t(i) + E_t, B_{max}\right\} \quad \forall t \in \{1, 2, \ldots T\} \tag{2}$$

where B_{max} is the battery storage capacity and B_0 is the initial battery energy.

The first bullet point aligns with a class of problems called Online Convex Optimization (OCO) problems that have been well studied, see for example [5, 8, 15, 16, 30, 44]. The second two bullet points introduce nontrivial constraints on resource allocation. These constraints are mathematically challenging to combine with OCO because the energy queue maintains a memory of past decisions. If energy is allocated too aggressively then the battery B_t can fall to zero. If this happens, then energy cannot be further allocated until new energy arrivals are harvested. These energy constraints are crucial for the operation of practical energy-limited systems. It is important to develop a mathematical technique to incorporate them into the OCO paradigm.

This problem of combining OCO with energy harvesting was first studied in [41], which is also the motivation of the current paper. There, a drift-plus-penalty technique was combined with OCO to ensure that, for any $\epsilon > 0$, the time-average loss is at most ϵ and the battery size needs to be at most $O(1/\epsilon)$. When formulated over a fixed time horizon T, these results translate into $O(\sqrt{T})$ regret with battery size $O(\sqrt{T})$. However, for the algorithm in [41], the coefficient that multiplies the regret expression has a linear dependence on n.

This paper seeks to develop a tighter result that reduces the dependence on n from $O(n)$ to $O(\sqrt{\log(n)})$. Such reductions are known to be possible for the simpler class of OCO problems without energy constraints, and in the special case when the decision vector is constrained to a probability simplex, via the use of *Bregman divergence* [1, 17, 31] (see also related techniques for the class of multi-armed bandit problems in [6]). The success of Bregman divergence in that context offers some hope that such an improvement may be possible for the energy harvesting problem. This question is important because a reduction to $\sqrt{\log(n)}$ is a significant improvement when n is large. However, the application of Bregman divergence for the energy harvesting problem is not trivial. First note that the structure of the constraint (1) is only consistent with a probability simplex if the time-varying $B_{t-1} + E_t$ on the right-hand-side is replaced by 1. Second, it is not obvious how to incorporate Bregman divergence into the OCO analysis when there is an energy queue that maintains a memory of past decisions. We are not aware of prior work that uses Bregman divergence in this context.

This paper presents a new approach that separates the decision vector $X_t = [X_t(1), \ldots, X_t(n)]$ into amplitude and direction components, and then makes decisions for each component that are informed by the prior loss functions and by the currently available energy B_t. Specifically, we write

$$X_t = A_t \cdot [P_t(1), P_t(2), \ldots, P_t(n)]$$

where A_t is the nonnegative amplitude and $[P_t(1), \ldots P_t(n)]$ is the direction vector. The direction vector is constrained to the probability simplex and the proposed algorithm chooses this vector by minimizing an expression that involves a Kullback-Leibler (KL) divergence term. On the other hand, the amplitude A_t is chosen separately by minimizing an expression that involves the battery B_t and a quadratic divergence term. The KL divergence and the quadratic divergence terms are two distinct forms of Bregman divergence. Both forms are needed for the algorithm. The algorithm also incorporates a Lyapunov function that biases the battery state B_t away from zero.

1.1 Wireless transmission example

Consider rateless coding over a multi-channel wireless transmitter. Each channel $i \in \{1, \dots, n\}$ offers a bit rate on slot t according to a curve that is similar to the Shannon-Hartley capacity formula:

$$C_t(i) = B(i) \log_2 \left(1 + \tfrac{X_t(i) S_t(i)}{N_t(i)}\right)$$

where $C_t(i)$ is the bit rate for channel i on slot t (which is a concave function so we define the loss function as the multiplication of this with -1); $X_t(i)$ is the power allocated to channel i on slot t; $S_t(i)/N_t(i)$ is the attenuation-to-noise coefficient for channel i on slot t; $B(i)$ is a positive constant that depends on the available bandwidth and the efficiency of the rateless coding scheme. For simplicity it shall be assumed that $B(i)$ is the same for all i. The $S_t(i)/N_t(i)$ values are time-varying, are possibly different for each channel i, and can have arbitrary time and space dependencies. The values $S_t(i)/N_t(i)$ can (possibly) be chosen by an adversary. These values are unknown until *after* the $X_i(t)$ decision is made.

The transmitter harvests energy from an inconsistent source such as a solar panel. The new energy E_t that arrives for each time slot t is an i.i.d sample of a random and unknown distribution. The transmitter's goal is to manage the received energy, meaning that it decides how much to store for the future in its finite capacity battery, and how much to allocate to each channel, in order to maximize the time average of $\sum_{i=1}^{n} C_t(i)$ over $t \in \{1, \dots, T\}$. The convex loss function is thus

$$L_t(X_t) = -\sum_{i=1}^{n} B(i) \log_2 \left(1 + \frac{X_t(i) S_t(i)}{N_t(i)}\right)$$

To consider an adversarial situation, imagine an adversary as a second "virtual" transmitter that makes decisions on a virtual system with the same $S_t(i)/N_t(i)$ sample path. The adversary has an unlimited battery capacity, knows the expectation $\mathbb{E}[E_t]$, and can choose the $S_t(i)/N_t(i)$ values however it likes over time. However, it is constrained to choosing a fixed allocation vector $[X^*(1), \dots, X^*(n)]$ that is the same on each slot $t \in \{1, \dots, T\}$, where $[X^*(1), \dots, X^*(n)]$ is a vector with nonnegative components that sum to $\mathbb{E}[E_t]$. The goal of the adversary is to choose a constant vector $[X^*(1), \dots, X^*(n)]$ and a sample path for $S_t(i)/N_t(i)$ for i and t so that the difference between its average bit rate and the original transmitter's average bit rate is maximized.

1.2 Why using Bregman divergence is important: An example

As another example, consider a situation where every slot t the controller chooses a decision A_t from a finite set $\mathcal{A} = \{a(0), a(1), \dots, a(n)\}$. A nonnegative reward of $f_t(A_t)$ is incurred, where $f : \mathcal{A} \to \mathbb{R}$ is an arbitrary (nonconvex and nonconcave) function that varies arbitrarily with time and that is unknown at the start of each slot t when the decision $A_t \in \mathcal{A}$ is made. Assume that one unit of energy is expended whenever $A_t \in \{a(1), \dots, a(n)\}$; zero units are used when $A_t = a(0)$; zero reward is earned if $A_t = a(0)$. Thus, rewards can only be earned if there is enough energy to choose $A_t \in \{a(1), \dots, a(n)\}$, else, the reward on slot t is zero. This can be transformed to the online *convex* framework of this paper by defining $\mathcal{P}_n = \{(x_0, x_1, \dots x_n) : x_i \geq 0 \quad \forall i \in \{0, \dots, n\}, \sum_{i=0}^{n} x_i = 1\}$ and defining the (linear) loss function $L_t : \mathcal{P}_n \to \mathbb{R}$ by

$$L_t(x(0), x(1), \dots, x(n)) = -\textstyle\sum_{i=1}^{n} x(i) f_t(a(i))$$

Then $X_t = [X_t(0), X_t(1), \ldots, X_t(n)]$ is a decision vector that represents the probability of choosing the particular elements $\{a(0), a(1), \ldots, a(n)\}$ on slot t and $L_t(X_t)$ is the corresponding (expected) loss. When n is large, say $n = 10^{10}$, algorithms with regret that depends linearly on n cannot perform well. The reduction to $O(\sqrt{\log(n)})$ achieved in this paper enables reasonable regret bounds (and battery capacities) even for very large values of n. For example, $\sqrt{\log(10^{10})} \approx 4.798$. Of course, even though the regret bound is small, the per-slot implementation can be high when n is very large because our algorithm chooses X_t according to a formula that is computed by summing over n terms.

1.3 Related work

The OCO problem of minimizing $L_t(X_t)$ for a sequence of convex loss functions $L_t(\cdot)$ was introduced in [44], where a subgradient-based algorithm was shown to achieve a *regret* of $O(\sqrt{T})$, where regret is measured with respect to the best fixed allocation decision X^* that could be chosen in hindsight. Specifically,

$$Regret(T) = \textstyle\sum_{t=1}^{T} L_t(X_t) - \inf_{X \in \mathcal{X}} \sum_{t=1}^{T} L_t(X)$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ is the convex domain of the $L_t(\cdot)$ functions. The asymptotic regret of $O(\sqrt{T})$ is known to be optimal over the class of general problems, but can be improved to $O(\log(T))$ regret in the special case when the loss functions are strongly convex with a common strong convexity parameter [16]. Bregman divergence has been used in [1, 17, 31] for online learning problems, including bandit problems in [6].

It is impossible to achieve similar regret guarantees for OCO problems with general time-varying constraints. This is shown in [23] for an example with just one constraint: Any algorithm that makes efficient decisions that satisfy the constraints over time $\{1, \ldots, T/2\}$ necessarily makes decisions that are either inefficient or violate the constraints when viewed over time $\{1, \ldots, T\}$. Therefore, constrained OCO problems require more structured assumptions for the constraints. OCO problems with non-time-varying constraints are studied in [19, 22, 33, 42, 43], and OCO with time-varying constraints that are i.i.d. over time are studied in [12, 20, 40]. A recent work in [36] introduces Bregman divergence into the study of constrained OCO, although the formulation does not have a memory-based energy queue and the context and algorithm developed there are different from the current paper.

The prior work [41], described in the previous section, treats OCO with energy harvesting but has $O(\sqrt{n})$ dependence on the system dimension. Energy harvesting has also been studied without the OCO structure, see, for example, [2, 4, 14, 18, 24, 32, 35, 37, 38].

Also, there have been papers focused on handling "inventory constraints" such as [10, 21, 26, 39] where, for example, [39] provides an algorithm with a theoretical guarantee to solve a problem where we have a limited capacity inventory. However, the objective function is linear and known on each slot t whereas the objective function in this paper can be nonlinear and unknown at the time of making each decision. In addition, there are related works focused on the one-way trading problem and the online knapsack problem which uses techniques that can be applied generally to OCO with constraints such as [7, 13].

1.4 Our contributions

This paper shows how to use Bregman divergence for OCO energy harvesting. We develop an algorithm that achieves $O(\sqrt{T})$ regret while reducing the regret coefficient from $O(n)$ to $O(\sqrt{\log(n)})$. This is a significant improvement when n is large. It should be noted that the $\sqrt{T}$ asymptotic is optimal and cannot be improved even for simpler OCO problems without energy harvesting constraints. To avoid the fundamental impossibility result of constrained OCO with arbitrary time-varying loss and constraint functions of [23], we assume the energy arrival process $\{E_t\}_{t=1}^T$ is independent and identically distributed (i.i.d.) over slots with an unknown distribution. The loss functions $\{L_t\}_{t=1}^T$ are arbitrary and are not required to be i.i.d. over slots. Our algorithm uses an energy queue, a Lyapunov function that biases the battery B_t away from zero, and a technique that separates the decision vector into amplitude and direction components.

2 PROBLEM FORMULATION

The system operates over slotted time $t \in \{1, 2, \dots, T\}$, where T is a positive integer. Fix n as a positive integer and define

$$\mathcal{X} = \left\{x \in \mathbb{R}^n : A_{min} \leq \textstyle\sum_{i=1}^n x_i \leq A_{max},\ x_i \geq 0,\ \forall i \in \{1, 2, \dots, n\}\right\} \tag{3}$$

where A_{min} and A_{max} are given real numbers that represent the minimum and maximum amount of energy that can be allocated per slot, where $0 \leq A_{min} \leq A_{max}$. (Typically $A_{min} = 0$.) For each $t \in \{1, \dots, T\}$ define

- E_t : The random amount of energy arrivals that can be harvested on slot t. Assume $\{E_t\}_{t=1}^T$ is i.i.d. with an unknown distribution. However, it is assumed that the random variables E_t are bounded so that $E_{min} \leq E_t \leq E_{max}$ for all t, where E_{min} and E_{max} are constants that satisfy $0 \leq E_{min} \leq E_{max}$. (Typically $E_{min} = 0$.)
- B_{t-1}: The available energy in the battery at the start of slot t.
- $X_t = [X_t(1), \dots, X_t(n)]$: The energy decision vector on slot t.
- $L_t : \mathcal{X} \to \mathbb{R}$: A continuous and convex function that shall be called a *loss function.*

The sequence of functions $\{L_t\}_{t=1}^T$ arises according to an arbitrary probability law and is not necessarily i.i.d. over slots. It is assumed that for each $t \in \{1, \dots, T\}$, the random energy arrival E_t is independent of the realization of function L_t. The function L_t is unknown to the system controller at the start of slot t and is only revealed at the end of slot t (after the X_t decision is made). For example, $\{L_t\}_{t=1}^T$ might be a deterministic sequence of functions that is fixed on slot 0 but only revealed to the controller gradually over time. Alternatively, $\{L_t\}_{t=1}^T$ can arise according to some random process that depends on the $(B_\tau, E_\tau, X_\tau, L_\tau)$ history over all slots $\tau < t$ (which still maintains independence between $L_t(\cdot)$ and E_t). This includes the possibility that L_t is chosen *adversarially* by an enemy that chooses loss functions in an effort to disrupt the system.

Fix $B_0 = 0$ as the initial battery energy. Every slot $t \in \{1, \dots, T\}$ the controller observes B_{t-1} and E_t and chooses a decision vector X_t that satisfies

$$\textstyle\sum_{i=1}^n X_t(i) \leq B_{t-1} + E_t \tag{4}$$

$$X_t \in \mathcal{X} \tag{5}$$

where (4) implies the total energy used on slot t does not exceed the available energy on that slot; (5) ensures the allocated energy is nonnegative and does not violate the maximum or minimum levels specified by constants A_{min} and A_{max}. The entire loss function L_t is revealed at the end of slot t, the corresponding loss $L_t(X_t)$ is incurred, and the battery energy is updated via (2).

2.1 Assumptions

Assume that each function $L_t : \mathcal{X} \to \mathbb{R}$ is continuous, convex, and has subgradients at each point $x \in \mathcal{X}$. Let $\nabla L_t(x)$ denote a subgradient vector for L_t at the point $x \in \mathcal{X}$ (note that $\nabla L_t(x)$ can be a gradient if L_t is differentiable). Let $\frac{\partial}{\partial x(i)} L(x), \forall i \in \{1, 2, \dots, n\}$ denote each component of vector $\nabla L(x)$. Suppose there is a positive constant G such that for all $i \in \{1, 2, \dots, n\}$, all $t \in \{1, 2, \dots, T\}$, and all $x \in \mathcal{X}$:

$$\left|\frac{\partial}{\partial x(i)} L_t(x)\right| \leq G \tag{6}$$

This implies that $||\nabla L_t(x)||_\infty \leq G$. Assume the $A_{min}, A_{max}, E_{min}, E_{max}$ constants satisfy

$$0 \leq E_{min} \leq E_{max} \tag{7}$$

$$0 \leq A_{min} \leq E_{min} < A_{max} \tag{8}$$

This holds whenever $A_{min} = E_{min} = 0$ and $A_{max} > 0$. The constraint $A_{min} \leq E_{min}$ ensures the problem is *feasible*, so the amount of new energy that arrives every slot is at least the minimum value needed for allocation. The constraint $E_{min} < A_{max}$ makes the problem *nontrivial*: If this inequality does not hold then there is no need for a battery because the new energy arrival on each slot would be at least as large as the amount allowed for allocation.

3 ALGORITHM

3.1 Amplitude and direction components

It is convenient to decompose the decision vector $X_t = [X_t(1), \ldots, X_t(n)]$ into amplitude and direction components, so that

$$X_t = A_t P_t$$

where

$$A_t = \sum_{i=1}^n X_t(i)$$

and

$$P_t = [P_t(1), \ldots, P_t(n)] = \begin{cases} \frac{X_t}{A_t} & \text{if } A_t > 0 \\ [\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}] & \text{else} \end{cases}$$

3.2 Algorithm specification

The following algorithm chooses A_t and P_t every slot t. It uses nonnegative system constants $A_{max}, A_{min}, E_{max}, E_{min}, B_{max}$ defined above. It also uses positive parameters η, λ, θ that shall be carefully selected later. There is no information available on slot $t = 1$ and so the algorithm chooses $X_1 = [\frac{A_{min}}{n}, \ldots, \frac{A_{min}}{n}]$. The final step in the algorithm chooses A_{t+1} as a projection of a real number z onto the interval $[A_{min}, \min\{A_{max}, B_t + E_{t+1}\}]$, denoted $[z]_{A_{min}}^{\min\{A_{max}, B_t+E_{t+1}\}}$. Note that $A_{min} \leq \min\{A_{max}, B_t + E_{t+1}\}$ because $E_{t+1} \geq E_{min} \geq A_{min}$.

Algorithm 1: General Amplitude-Direction algorithm

Fix constants A_{max}, A_{min}, E_{max}, E_{min}, and B_{max} $(0 \leq A_{min} \leq E_{min})$;
Fix parameters $\eta > 0$, $\lambda > 0$, and $\theta > 0$;
Fix $B_0 = 0$, $P_1 = [\frac{1}{n}, \cdots, \frac{1}{n}]$, and $A_1 = A_{min}$;
for $t \leftarrow 1$ **to** T **do**
  Define $X_t = A_t P_t$;
  Get $L_t(X_t)$ and E_t;
  Put $B_t = \min\{B_{t-1} - A_t + E_t, B_{max}\}$;
  Put $P_{t+1}(i) = P_t(i) \frac{exp(-\lambda[\nabla L_t(X_t)](i))}{\sum_{i=1}^n P_t(i) exp(-\lambda[\nabla L_t(X_t)](i))}, \forall i \in \{1, 2, \cdots, n\}$;
  Put $A_{t+1} = [A_t + \theta(B_t - B_{max}) - \eta \nabla L_t(X_t)^\top P_t]_{A_{min}}^{\min\{A_{max}, B_t+E_{t+1}\}}$;
end

3.3 Bregman divergence

This section describes key properties of *Bregman divergence*, a concept that is useful for development and analysis of the algorithm. Fix d as a positive integer and let $G \subseteq \mathbb{R}^d$ be a convex set with nonempty interior. Let $\Phi : G \to \mathbb{R}$ be a (possibly nonconvex) function that is continuously differentiable in the interior of G. Let $C \subseteq G$ be a convex subset that intersects interior(G) and define $C^o = C \cap \text{interior}(G)$. Note that $C^0 = C$ in the special case when G is an open set, such as when $G = \mathbb{R}^d$. The *Bregman divergence* $D : C \times C^o \to \mathbb{R}$ generated from $\Phi(\cdot)$ is

$$D(x, y) = \Phi(x) - \Phi(y) - \nabla\Phi(y)^\top \cdot (x - y)$$

If Φ is a convex function then the basic subgradient inequality for convex functions ensures that $D(x, y) \geq 0$ for all $x \in C, y \in C^o$. The following result can be found in various forms in [11, 27, 34], the particular form stated below is proven in [36].[1]

Lemma 3.1. *(Pushback) Let $f : G \to \mathbb{R}$ be a convex function. Fix $\alpha > 0$, $y \in C^o$. Suppose*

$$\hat{x} \in \arg\min_{x \in C} \{f(x) + \alpha D(x, y)\} \tag{9}$$

and also suppose $\hat{x} \in C^o$. Then

$$f(\hat{x}) + \alpha D(\hat{x}, y) \leq f(z) + \alpha D(z, y) - \alpha D(z, \hat{x}) \quad \forall z \in C \tag{10}$$

For intuition about the above lemma, note that inequality (10) would follow immediately by definition of $\hat{x}$ as a minimizer if the final term $-D(z, \hat{x})$ on the right-hand-side were removed. The structure of the minimization problem (9) ensures that the inequality can be strengthened to include the "pushback" term $-D(z, \hat{x})$. Two types of Bregman divergence functions shall be used:

- Euclidean distance: Let $G = C = C^o = \mathbb{R}^d$. Let $\Phi : \mathbb{R}^d \to \mathbb{R}$ be $\Phi(x) = \frac{1}{2}||x||_2^2$. Define $D : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ by

  $$D(x, y) = \tfrac{1}{2}||x - y||_2^2$$

  With this divergence function, it can be shown that the minimization in (9) has a unique minimizer $\hat{x} \in \mathbb{R}^d$. We use this type of divergence for the amplitude decisions A_t and define $D_A : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ by

  $$D_A(x, y) = \tfrac{1}{2}(x - y)^2 \tag{11}$$

- Generalized Kullback-Leibler divergence: Fix n as a positive integer. Let $C = G = [0, \infty)^n$ and let $C^o = (0, \infty)^n$. Let $\Phi : [0, \infty)^n \to \mathbb{R}$ be $\Phi(x) = \sum_{i=1}^n x(i) \log x(i)$, where $x \log(x)$ is defined to be 0 if $x = 0$. Then $D : [0, \infty)^n \times (0, \infty)^n \to \mathbb{R}$ is defined

  $$D(x, y) = \textstyle\sum_{i=1}^n x(i) \log \frac{x(i)}{y(i)} - \sum_{i=1}^n x(i) + \sum_{i=1}^n y(i)$$

  With this divergence function, it can be shown that if f is a linear function then the minimization in (9) has a unique minimizer $\hat{x}$, and that $\hat{x} \in (0, \infty)^n$. We use this type of divergence for the direction decisions $P_t = [P_t(1), \ldots, P_t(n)]$ and define $D_P : [0, \infty)^n \times (0, \infty)^n \to \mathbb{R}$ by

  $$D_P(x, y) = \textstyle\sum_{i=1}^n x(i) \log \frac{x(i)}{y(i)} - x(i) + y(i) \tag{12}$$

The technique of optimizing a function using the sum of a sub-gradient term and a Bregman divergence term is called *mirror descent* for offline problems [3, 28] and *online mirror descent* for online problems [9, 17]. This paper shall use a hybrid version of online mirror descent.

[1]The statement in [36] adds an unnecessary condition that C is compact and contains the origin, although this condition is not used in the proof given in [36].

3.4 Algorithm intuition

Recall that the decision vector is $X_t = A_t P_t$. Define $H_t(A_t, P_t)$ as the loss function $L_t(X_t)$ written explicitly in terms of the components $A_t \in \mathbb{R}$ and $P_t \in \mathbb{R}^n$:

$$H_t(A_t, P_t) = L_t(A_t P_t)$$

For intuition, temporarily assume the function $L_t(X_t)$ is defined for all $X_t \in \mathbb{R}^n$, and the function H_t is defined over all $(A_t, P_t) \in \mathbb{R} \times \mathbb{R}^n$ (this assumption is only used in this subsection to motivate the algorithm, and is not used in the mathematical analysis of the algorithm). Under this temporary assumption we formally have by the chain rule of differentiation:

$$\frac{\partial H(A_t,P_t)}{\partial a} = \nabla L_t(A_t P_t)^\top P_t \tag{13}$$

$$\frac{\partial H(A_t,P_t)}{\partial p(i)} = A_t \nabla L_t(A_t P_t)^\top u(i) \quad \forall i \in \{1, \dots, n\} \tag{14}$$

where (13) takes a partial derivative with respect to the first component A_t; (14) takes a partial derivative with respect to the ith component of the P_t vector; and $u(i)$ denotes a unit vector in $\mathbb{R}^n$ with all zeros except for a 1 in entry i.

To maintain a battery B_t far from 0, define

$$V(t) = \frac{1}{2}(B_t - B_{max})^2$$

The function $V(t)$ shall be called a *Lyapunov function.* Define $\Delta(t) = V(t+1) - V(t)$ as the change in the Lyapunov function over one slot. Recall that the decision vector is $X_t = A_t P_t$. The idea is to make separate decisions for A_{t+1} and P_{t+1} on each slot $t+1$ that minimize a bound on:

$$\underbrace{\theta\Delta(t)}_{1} + \underbrace{\eta \nabla L_t(A_t P_t)^\top P_t \cdot A_{t+1} + D_A(A_{t+1}, A_t)}_{2} + \underbrace{\lambda L_t(A_t P_t)^\top \cdot P_{t+1} + D_P(P_{t+1}, P_t)}_{3} \tag{15}$$

where θ, η, λ are positive parameters that shall be chosen later. Term 1 in expression (15) is the change in the Lyapunov function, often called the *drift term* in queue optimization [25]. Minimizing this alone would intuitively maintain a battery level close to B_{max}. Term 2 in expression (15) relates to the partial derivative of the loss function with respect to the amplitude component A_t (compare to (13)). Minimizing this alone could be viewed as a "partial" online mirror descent method that uses a divergence $D_A(\cdot,\cdot)$ and seeks to minimize the loss function by only using the amplitude component A_t (see [28] for use of Bregman divergence terms for subgradient-based optimization, often called *mirror descent*).

Term 3 in expression (15) relates to the partial derivative of the loss function with respect to the direction component P_t (compare to (14)). Minimizing this alone could *almost* be viewed as a partial online mirror descent that uses a divergence $D_P(\cdot,\cdot)$ and seeks to minimize the loss function by only using the direction component P_t. However, a careful comparison of Term 3 with (14) shows that the scalar value A_t is missing! It is not obvious why this scalar value should be missing. It means that the expression (15) is *not* treating the partial derivative terms with respect to the components A_t and P_t equally. Rather, it separates out these components, removes a time-varying A_t term, and weights them by different (constant) parameters η and λ. This was done to make the regret analysis of the overall system possible. As shown in the analysis in the next two sections, there is a careful selection of parameters λ, η, and θ that align all pieces of the problem.

At the start of slot $t+1$, the $\Delta(t)$ term in (15) is not a known function of the decisions A_{t+1} and P_{t+1}. It turns out that it suffices to use a bound on $\Delta(t)$:

$$\begin{aligned}
\Delta(t) &\overset{(a)}{=} \frac{1}{2}(B_{t+1}-B_{max})^2 - \frac{1}{2}(B_t - B_{max})^2 \\
&\overset{(b)}{=} \frac{1}{2}(\min\{B_t - A_{t+1} + E_{t+1}, B_{max}\} - B_{max})^2 - \frac{1}{2}(B_t - B_{max})^2 \\
&= \frac{1}{2}\min\{B_t - B_{max} - A_{t+1} + E_{t+1}, 0\}^2 - \frac{1}{2}(B_t - B_{max})^2 \\
&\overset{(c)}{\leq} \underbrace{(B_t - B_{max})(-A_{t+1}+E_{t+1})}_{\text{include this part}} + \frac{1}{2}(-A_{t+1}+E_t)^2
\end{aligned} \tag{16}$$

where (a) holds by definition of $\Delta(t)$; (b) holds by the battery update equation (2); (c) holds by the inequality $\min\{x,0\}^2 \leq x^2$. Replacing $\Delta(t)$ in (15) with the above upper bound (and including only the term marked by an underbrace in (16)) means that our algorithm shall seek to minimize:[2]

$$\begin{aligned}
&\underbrace{\theta(B_t - B_{max})(-A_{t+1}+E_{t+1})}_{1'} + \underbrace{\eta\nabla L_t(A_tP_t)^\top P_t \cdot A_{t+1} + D_A(A_{t+1},A_t)}_{2} \\
&\quad + \underbrace{\lambda L_t(A_tP_t)^\top \cdot P_{t+1} + D_P(P_{t+1},P_t)}_{3}
\end{aligned} \tag{17}$$

3.5 Algorithm development

On slot $t = 1$ there is no information available and our algorithm chooses $A_1 = A_{min}$ and $P_1 = [\frac{1}{n}, \ldots, \frac{1}{n}]$. On each slot $t+1$ we choose A_{t+1} and P_{t+1} to directly minimize their corresponding terms in (17), which amounts to:

- Selecting P_{t+1}: Choose $P_{t+1} \in \mathbb{R}^n$ as the solution to:

$$\begin{aligned}
\text{Minimize:} \quad & \lambda\nabla L_t(A_tP_t)^\top \cdot P_{t+1} + D_P(P_{t+1},P_t) \\
\text{Such that:} \quad & 0 \leq P_{t+1}(i), \quad \forall i \in \{1,\cdots,n\} \\
& \textstyle\sum_{i=1}^n P_{t+1}(i) = 1
\end{aligned} \tag{18}$$

- Selecting A_{t+1}: Choose $A_{t+1} \in \mathbb{R}$ as the solution to:

$$\begin{aligned}
\text{Minimize:} \quad & \eta\nabla L_t(A_tP_t)^\top P_t \cdot A_{t+1} + D_A(A_{t+1},A_t) \\
& \quad + \theta(B_{max} - B_t)A_{t+1} \\
\text{Such that:} \quad & A_{min} \leq A_{t+1} \leq \min\{B_t + E_{t+1}, A_{max}\}
\end{aligned} \tag{19}$$

Then define $X_{t+1} = A_{t+1}P_{t+1}$ and update the battery to obtain B_{t+1} via (2). Recall that $A_{min} \leq E_{min} \leq E_{t+1}$ and so the interval constraint on A_{t+1} in (19) is feasible and ensures $X_{t+1} \in \mathcal{X}$ and the sum of components is no more than the available energy $B_t + E_{t+1}$.

The two problems (18) and (19) have a structure similar to a simple online mirror descent update (where (19) includes an additional term $\theta(B_{max} - B_t)A_{t+1}$ from the Lyapunov drift). By a standard Lagrange multiplier argument, it is not difficult to show that the solution to (18) is

$$P_{t+1}(i) = P_t(i)\frac{exp(-\lambda[\nabla L_t(A_tP_t)](i))}{\sum_{i=1}^n P_t(i)exp(-\lambda[\nabla L_t(A_tP_t)](i))} \quad \forall i \in \{1,\ldots,n\} \tag{20}$$

[2]The final term $\frac{1}{2}(-A_{t+1}+E_t)^2$ in (16) shall be bounded by a constant later and does not affect algorithm decisions.

Further, the solution to (19) is

$$A_{t+1} = \left[A_t + \theta(B_t - B_{max}) - \eta\nabla L_t(A_t P_t)^\top P_t\right]_{A_{min}}^{\min\{A_{max}, B_t + E_{t+1}\}} \tag{21}$$

The resulting algorithm is specified in Section 3.2. Observe from (20) that the P_t vector has strictly positive components for all $t \in \{1, \ldots, T\}$.

4 RELAXATION THEOREM

This section considers a sample path implementation of Algorithm 1 with parameters $\eta > 0$, $\theta > 0$, $\lambda > 0$, and constants E_{min}, E_{max}, A_{min}, A_{max}, G that satisfy the assumptions (6)-(8). It is proven that if the battery size B_{max} is is chosen wisely, then a special property holds: For each $t \geq 1$ the decision A_{t+1} produced by the algorithm, which is the solution to (19), is also a solution to the *relaxed problem* of choosing $A_{t+1} \in \mathbb{R}$ to solve

$$\begin{aligned} \text{Minimize:} \quad & \eta\nabla L_t(A_t P_t)^\top P_t \cdot A_{t+1} + D_A(A_{t+1}, A_t) \\ & \quad + \theta(B_{max} - B_t)A_{t+1} \\ \text{Such that:} \quad & A_{min} \leq A_{t+1} \leq A_{max} \end{aligned} \tag{22}$$

The difference between (19) and (22) is that the constraint has been relaxed to $A_{min} \leq A_{t+1} \leq A_{max}$, so that this constraint does not depend on the time-varying $B_t + E_{t+1}$ value. This paves the way to the regret analysis of the next section. The solution to (22) is (compare with (21)):

$$A_{t+1} = \left[A_t + \theta(B_t - B_{max}) - \eta\nabla L_t(A_t P_t)^\top P_t\right]_{A_{min}}^{A_{max}} \tag{23}$$

4.1 Relaxed version

Define the *relaxed version* of Algorithm 1 as the same algorithm but with the A_{t+1} decision rule (21) replaced by (23). Since this decision rule no longer explicitly guarantees $A_{t+1} \leq B_t + E_{t+1}$, the relaxed version may not be implementable because it may cause the battery energy B_t to go negative. The decision rules (21) and (23) are one and the same, so that the relaxed algorithm is exactly the same as the original, if and only if B_t never goes negative.

Fix $a \in (0, A_{max} - E_{min}]$ and define

$$B_{max} = \frac{a + \eta G}{\theta} - E_{min} + A_{min} + \frac{A_{max} - E_{min}}{a}(A_{max} - A_{min}) \tag{24}$$

It is not difficult to show that this choice of B_{max} is strictly positive (since $A_{max} > E_{min}$ by (8)).

Lemma 4.1. *Fix $a \in (0, A_{max} - E_{min}]$ and assume B_{max} satisfies (24). Fix $t \geq 1$. Under the decisions of the relaxed algorithm, if $(B_t - B_{max}) \leq \frac{-1}{\theta}(a + \eta G)$ then $A_{t+1} \leq \max\{A_t - a, A_{min}\}$.*

Proof. From (6) and the fact that components of P_t are non-negative and sum to 1, we have:

$$|\nabla L_t(A_t P_t)^\top P_t| \leq G$$

So

$$\begin{aligned} A_t + \theta(B_t - B_{max}) - \eta\nabla L_t(A_t P_t)^\top P_t &\leq A_t + \theta(B_t - B_{max}) + \eta G \\ &\overset{(a)}{\leq} A_t - (a + \eta G) + \eta G \\ &= A_t - a \end{aligned}$$

where (a) holds by the assumption of the lemma. Then from the relaxed update rule (23):

$$A_{t+1} \leq [A_t - a]_{A_{min}}^{A_{max}} \leq \max\{A_t - a, A_{min}\}$$

□

For the next theorem, we recall that $E_{min} < A_{max}$ (assumption (8)).

THEOREM 4.2. *Fix $a \in (0, A_{max} - E_{min}]$ and assume B_{max} satisfies (24). Under the decisions of the relaxed algorithm we have $B_t \geq 0$ for all $t \in \{1, \dots, T\}$ and so the relaxed algorithm and the original algorithm are identical.*

PROOF. We shall use induction to show that for all slots $t \geq 2$ we have:

$$(B_{t-1} - B_{max}) \geq \frac{-1}{\theta}(a + \eta G) + E_{min} - A_{t-1} + \frac{A_{max} - E_{min}}{a}(A_{t-1} - A_{max}) \tag{25}$$

To see that this inequality is sufficient to ensure $B_t \geq 0$ for all $t \in \{1, \dots, T\}$, we can substitute the definition of B_{max} from (24) into (25) to find

$$\begin{aligned} B_{t-1} &\geq A_{min} - A_{t-1} + \frac{A_{max} - E_{min}}{a}(A_{t-1} - A_{min}) \\ &\overset{(a)}{\geq} A_{min} - A_{t-1} + (A_{t-1} - A_{min}) = 0 \end{aligned}$$

where (a) holds because $A_{max} - E_{min} > 0$ and $a \in (0, A_{max} - E_{min}]$ and so

$$\frac{A_{max} - E_{min}}{a} \geq 1 \tag{26}$$

We first show (25) holds for the base case $t = 2$. From the queue update equation (2) we have

$$\begin{aligned} B_1 &= \min\{B_0 - A_1 + E_1, B_{max}\} \\ &\overset{(a)}{=} \min\{-A_{min} + E_1, B_{max}\} \\ &\overset{(b)}{\geq} \min\{0, B_{max}\} \geq 0 \end{aligned} \tag{27}$$

where (a) holds by our initializations $B_0 = 0$, $A_1 = A_{min}$; (b) holds because $A_{min} \leq E_{min} \leq E_1$. Thus, $B_1 \geq 0$. From (24) we have

$$\begin{aligned} B_1 - B_{max} &= B_1 - \frac{a + \eta G}{\theta} + E_{min} - A_{min} - \frac{A_{max} - E_{min}}{a}(A_{max} - A_{min}) \\ &\overset{(a)}{\geq} -\frac{a + \eta G}{\theta} + E_{min} - A_{min} - \frac{A_{max} - E_{min}}{a}(A_{max} - A_{min}) \\ &\overset{(b)}{=} -\frac{a + \eta G}{\theta} + E_{min} - A_1 - \frac{A_{max} - E_{min}}{a}(A_{max} - A_1) \end{aligned}$$

where (a) holds because $B_1 \geq 0$; (b) holds because $A_1 = A_{min}$.

Now, we show the same inequality holds for slot t. There are three cases.

Case 1: If $B_{t-1} + E_t - A_t \geq B_{max}$ then from B_t update rule (2) we have $B_t = B_{max}$. Then

$$\begin{aligned} &\frac{-1}{\theta}(a + \eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max}) \\ &\overset{(a)}{\leq} E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max}) \\ &\overset{(b)}{\leq} E_{min} - A_{max} \\ &\overset{(c)}{<} 0 \\ &\overset{(d)}{=} B_t - B_{max} \end{aligned}$$

where (a) holds because $\frac{-1}{\theta}(a+\eta G) \leq 0$; (b) holds because the assumption in the statement of the theorem ensures $A_{max} \geq E_{min} + a$; (c) holds by assumption (8); (d) holds because $B_t = B_{max}$. So the claim holds in Case 1.

In the remaining two cases we assume

$$B_{t-1} + E_t - A_t < B_{max} \tag{28}$$

which by (2) implies

$$B_t = B_{t-1} + E_t - A_t \tag{29}$$

Case 2: Suppose (28) and $(B_{t-1} - B_{max}) \geq \frac{-1}{\theta}(a+\eta G)$ hold. By (29) we have

$$\begin{aligned}
B_t - B_{max} &= B_{t-1} - B_{max} + E_t - A_t \\
&\overset{(a)}{\geq} \frac{-1}{\theta}(a+\eta G) + E_t - A_t \\
&\overset{(b)}{\geq} \frac{-1}{\theta}(a+\eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max})
\end{aligned}$$

where (a) holds by the assumption of this Case 2; (b) holds because $E_t \geq E_{min}$ and $\frac{A_{max}-E_{min}}{a}(A_t - A_{max}) \leq 0$. So the claim holds for Case 2.

Case 3: Suppose (28) and $(B_{t-1} - B_{max}) < \frac{-1}{\theta}(a+\eta G)$ hold. Then from Lemma 4.1 we have

$$A_t \leq \max\{A_{t-1} - a, A_{min}\} \tag{30}$$

We separate Case 3 into two subcases.

Case 3a: Suppose $A_{t-1} - a \geq A_{min}$. Then (30) implies

$$A_t \leq A_{t-1} - a \tag{31}$$

and from (29)

$$\begin{aligned}
B_t - B_{max} &= B_{t-1} - B_{max} + E_t - A_t \\
&\overset{(a)}{\geq} \frac{-1}{\theta}(a+\eta G) + E_{min} - A_{t-1} + \frac{A_{max} - E_{min}}{a}(A_{t-1} - A_{max}) + E_t - A_t \\
&= \frac{-1}{\theta}(a+\eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_{t-1} - a + a - A_{max}) + E_t - A_{t-1} \\
&\overset{(b)}{\geq} \frac{-1}{\theta}(a+\eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t + a - A_{max}) + E_t - A_{t-1} \\
&\overset{(c)}{\geq} \frac{-1}{\theta}(a+\eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max})
\end{aligned}$$

where (a) holds by (25); (b) holds by (31) and the fact $A_{max} - E_{min} \geq 0$; (c) holds because $A_{max} - A_{t-1} \geq 0$ and $E_t - E_{min} \geq 0$. So the claim holds for Case 3a.

Case 3b: Suppose $A_{t-1} - a < A_{min}$. By (30) we know $A_t \leq A_{min}$ and so $A_t = A_{min}$ (since A_t cannot be less than A_{min}). Thus

$$A_{t-1} - A_t \geq 0 \tag{32}$$

By (29) we have

$$
\begin{aligned}
B_t - B_{max} &= B_{t-1} - B_{max} + E_t - A_t \\
&\overset{(a)}{\geq} \frac{-1}{\theta}(a + \eta G) + E_{min} - A_{t-1} + \frac{A_{max} - E_{min}}{a}(A_{t-1} - A_{max}) + E_t - A_t \\
&= \frac{-1}{\theta}(a + \eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max}) \\
&\quad + E_t - A_{t-1} + \frac{A_{max} - E_{min}}{a}(A_{t-1} - A_t) \\
&\overset{(b)}{\geq} \frac{-1}{\theta}(a + \eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max}) \\
&\quad + E_t - A_{t-1} + \frac{A_{max} - E_{min}}{A_{max} - E_{min}}(A_{t-1} - A_t) \\
&\overset{(c)}{\geq} \frac{-1}{\theta}(a + \eta G) + E_{min} - A_t + \frac{A_{max} - E_{min}}{a}(A_t - A_{max})
\end{aligned}
$$

where (a) holds by (25); (b) holds because $a \in (0, A_{max} - E_{min}]$ and $(A_{max} - E_{min})(A_{t-1} - A_t) \geq 0$ from (32); (c) holds because $E_t - A_t = E_t - A_{min} \geq 0$. And so the theorem is proved. □

4.2 Optimizing B_{max}

As $a \in (0, A_{max} - E_{min}]$ was a parameter of choice in (24), we can choose $a \in (0, A_{max} - E_{min}]$ to minimize the required B_{max} value for our battery capacity.

$$
a^* = \left[\sqrt{\theta(A_{max} - E_{min})(A_{max} - A_{min})}\right]_0^{A_{max}-E_{min}}
$$

and so, assuming θ is small enough to ensure a^* is interior to the interval $(0, A_{max} - E_{min}]$, the battery capacity for this specific choice is

$$
B_{max} = \frac{\eta}{\theta}G + \frac{\sqrt{(A_{max} - E_{min})(A_{max} - A_{min})}}{\sqrt{\theta}} - E_{min} + A_{min} \tag{33}
$$

5 REGRET ANALYSIS OF THE RELAXED ALGORITHM

This section compares performance of the proposed algorithm to a virtual system that uses a fixed decision $X^* = [X^*(1), \ldots, X^*(n)]$. The battery available for the virtual system has infinite capacity, meaning that at each round it can use as much energy as it wants. However, the fixed vector $X^* = [X^*(1), \ldots, X^*(n)]$ is required to satisfy

$$
A^* = \textstyle\sum_{i=1}^n X^*(i) \leq \overline{E} = \mathbb{E}\left[E_t\right] \tag{34}
$$

$$
X^* \in \mathcal{X} \tag{35}
$$

where $\mathcal{X}$ is defined in (3), so $A^* \leq \min\{A_{max}, \overline{E}\}$. This means that the fixed decision of the virtual algorithm does not use more energy than the average available input energy of our algorithm. The regret is defined:

$$
Regret(T) = \textstyle\sum_{t=1}^T \mathbb{E}\left[L_t(X_t)\right] - \mathbb{E}\left[\inf_{X^* \in \mathcal{A}} \sum_{t=1}^T L_t(X^*)\right]
$$

where $\mathcal{A}$ is the set of all X^* that satisfy (34)-(35). Note that X^* can be chosen in the set $\mathcal{A}$ based on full knowledge of the L_t functions.

It is assumed throughout this section that the B_{max} parameter given in (24) is used, so that Algorithm 1 and its relaxed version that uses (23) are identical. Define

$$
A^*_{max} = \min\{A_{max}, \overline{E}\} \tag{36}
$$

5.1 Part 1

Fix $t \geq 1$. Since P_{t+1} is the solution to (18), we have by the pushback lemma (Lemma 3.1) that for any P^* in the n-dimensional simplex:

$$\lambda \nabla L_t(A_t P_t)^\top P_{t+1} + D_P(P_{t+1}, P_t) \leq \lambda \nabla L_t(A_t P_t)^\top P^* + D_P(P^*, P_t) - D_P(P^*, P_{t+1})$$

Adding $\lambda \nabla L_t(A_t P_t)^\top P_t$ to both sides and rearranging the terms,

$$\begin{aligned} &\lambda \nabla L_t(A_t P_t)^\top (P_t - P^*) \leq \\ &\left(\lambda \nabla L_t(A_t P_t)^\top (P_t - P_{t+1}) - D_P(P_{t+1}, P_t)\right) + \left(D_P(P^*, P_t) - D_P(P^*, P_{t+1})\right) \end{aligned} \tag{37}$$

The first part of the right-hand-side of (37) gives

$$\begin{aligned} &\lambda \nabla L_t(A_t P_t)^\top (P_{t+1} - P_t) + D_P(P_{t+1}, P_t) \\ &\stackrel{(a)}{=} \lambda \nabla L_t(A_t P_t)^\top (P_{t+1} - P_t) + \sum_{i=1}^{n} \left[P_{t+1}(i) \log \frac{P_{t+1}(i)}{P_t(i)} - P_{t+1}(i) + P_t(i)\right] \\ &\stackrel{(b)}{=} \lambda \nabla L_t(A_t P_t)^\top (P_{t+1} - P_t) + \left(\sum_{i=1}^{n} P_{t+1}(i) \log \frac{P_{t+1}(i)}{P_t(i)}\right) \\ &\stackrel{(c)}{\geq} \lambda \nabla L_t(A_t P_t)^\top (P_{t+1} - P_t) + \frac{1}{2}||P_{t+1} - P_t||_1^2 \\ &\stackrel{(d)}{\geq} -\lambda \nabla ||L_t(A_t P_t)||_\infty ||P_{t+1} - P_t||_1 + \frac{1}{2}||P_{t+1} - P_t||_1^2 \\ &\stackrel{(e)}{\geq} -\frac{\lambda^2}{2}||\nabla L_t(A_t P_t)||_\infty^2 \\ &\stackrel{(f)}{\geq} -\frac{\lambda^2}{2}G^2 \end{aligned}$$

where (a) uses the definition (12); (b) uses the fact that the P_t and P_{t+1} are in the n-dimensional simplex; (c) uses the Pinsker inequality; (d) is achieved by the Cauchy-Schwarz inequality; (e) holds by completing the square; (f) is from the finite gradient assumption (6). Replacing this result in (37) gives

$$\lambda \nabla L_t(A_t P_t)^\top (P_t - P^*) \leq \frac{\lambda^2}{2}G^2 + \left(D_P(P^*, P_t) - D_P(P^*, P_{t+1})\right) \tag{38}$$

5.2 Part 2

Similar to Part 1, since A_{t+1} is the solution to the optimization (22), we have by the pushback lemma (Lemma 3.1) that for any A^* that satisfies $A_{min} \leq A^* \leq A_{max}$:

$$\begin{aligned} &\eta \nabla L_t(A_t P_t)^\top P_t A_{t+1} + \theta(B_{max} - B_t)A_{t+1} + D_A(A_{t+1}, A_t) \\ &\leq \eta \nabla L_t(A_t P_t)^\top P_t A^* + \theta(B_{max} - B_t)A^* + D_A(A^*, A_t) - D_A(A^*, A_{t+1}) \end{aligned}$$

Adding $(\eta \nabla L_t(A_t P_t)^\top P_t)\, A_t$ to both sides and rearranging equations,

$$\begin{aligned} &\left(\eta \nabla L_t(A_t P_t)^\top P_t\right)(A^* - A_t) + \theta(B_t - B_{max})(A_{t+1} - A^*) \\ &\geq \left(\eta \nabla L_t(A_t P_t)^\top P_t\right)(A_{t+1} - A_t) + D_A(A_{t+1}, A_t) + D_A(A^*, A_{t+1}) - D_A(A^*, A_t) \end{aligned} \tag{39}$$

The first part of RHS gives

$$
\begin{aligned}
&\left(\eta \nabla L_t(A_t P_t)^\top P_t\right)(A_{t+1} - A_t) + D_A(A_{t+1}, A_t) \\
&\qquad \overset{(a)}{=} \left(\eta \nabla L_t(A_t P_t)^\top P_t\right)(A_{t+1} - A_t) + \frac{1}{2}(A_{t+1} - A_t)^2 \\
&\qquad \overset{(b)}{\geq} -\frac{\eta^2}{2}|\nabla L_t(A_t P_t)^\top P_t|^2 \\
&\qquad \overset{(c)}{\geq} -\frac{\eta^2}{2}G^2
\end{aligned}
$$

where (a) uses the definition (11); (b) is just the simple inequality $\frac{1}{2}a^2 + ab \geq -\frac{1}{2}b^2$, (c) uses the assumption (6). Substituting this result in (39) gives

$$
\begin{aligned}
&\left(\eta \nabla L_t(A_t P_t)^\top P_t\right)(A^* - A_t) + \theta(B_t - B_{max})(A_{t+1} - A^*) \\
&\geq -\frac{\eta^2}{2}G^2 + D_A(A^*, A_{t+1}) - D_A(A^*, A_t)
\end{aligned}
\tag{40}
$$

5.3 Summing Part1 & Part2

Multiply equation (38) by $\frac{A^*}{\lambda}$ and equation (40) by $\frac{-1}{\eta}$ and sum these two resulting inequalities to obtain

$$
\begin{aligned}
\nabla L_t(A_t P_t)^\top (A_t P_t - A^* P^*) + \frac{\theta}{\eta}(B_{max} - B_t)(A_{t+1} - A^*) \leq \\
(\eta + \lambda A^*)\frac{G^2}{2} \\
-\frac{1}{\eta}\left(D_A(A^*, A_{t+1}) - D_A(A^*, A_t)\right) \\
-\frac{A^*}{\lambda}\left(D_P(P^*, P_{t+1}) - D_P(P^*, P_t)\right)
\end{aligned}
$$

The above inequality holds for all $t \geq 1$. Choose $P_1 = [\frac{1}{n}, \cdots, \frac{1}{n}]$, $A_1 = A_{min}$, and take summation from time 1 to T

$$
\begin{aligned}
&\sum_{t=1}^{T}\left(\nabla L_t(A_t P_t)^\top (A_t P_t - A^* P^*) + \frac{\theta}{\eta}(B_{max} - B_t)(A_{t+1} - A^*)\right) \\
&\overset{(a)}{\leq} (\eta + \lambda A^*)\frac{G^2}{2}T \\
&\quad -\frac{1}{\eta}\left(D_A(A^*, A_{T+1}) - D_A(A^*, A_1)\right) \\
&\quad -\frac{A^*}{\lambda}\left(D_P(P^*, P_{T+1}) - D_P(P^*, P_1)\right) \\
&\overset{(b)}{\leq} (\eta + \lambda A^*)\frac{G^2}{2}T \\
&\quad +\frac{1}{\eta}D_A(A^*, A_1) + \frac{A^*}{\lambda}D_P(P^*, P_1) \\
&\overset{(c)}{\leq} (\eta + \lambda A^*_{max})\frac{G^2}{2}T \\
&\quad +\frac{1}{\eta}(A^*_{max} - A_{min})^2/2 + \frac{A^*_{max}}{\lambda}\log(n)
\end{aligned}
$$

where (a) uses the telescopic sum technique ($\sum_{t=1}^{T}(a_{t+1}-a_t)=a_{T+1}-a_1$); (b) uses the fact that $D_A(.,.)\geq 0$ and $D_P(.,.)\geq 0$; for (c) we used (36) along with these two upper bounds $D_A(A^*,A_1)\leq (A^*_{max}-A_{min})^2/2$ and $D_P(P^*,P_1)\leq \log(n)$. These upper bounds are proven as follows. For first one notice that $A_1=A_{min}$ and consider:

$$\begin{aligned}\text{Maximize:}\quad & D_A(A^*,A_{min})\\ \text{Such that:}\quad & A_{min}\leq A^*\leq \min\{A_{max},\overline{E}\}\end{aligned}$$

where the answer is $(\min\{A_{max},\overline{E}\}-A_{min})^2/2$. For the second bound consider:

$$\begin{aligned}\text{Maximize:}\quad & D_P(P^*,P_1)\\ \text{Such that:}\quad & 0\leq P_1(i),\quad \forall i\in\{1,\cdots,n\}\\ & \textstyle\sum_{i=1}^{n}P_1(i)=1\end{aligned}$$

where the answer is $\log n$.

Using the definitions $X_t=A_tP_t$ and $X^*=A^*P^*$ this can be written as

$$\sum_{t=1}^{T}\left(\nabla L_t(X_t)^\top(X_t-X^*)+\frac{\theta}{\eta}(B_{max}-B_t)(A_{t+1}-A^*)\right)\\ \leq(\eta+\lambda A^*_{max})\frac{G^2}{2}T+\frac{1}{\eta}(A^*_{max}-A_{min})^2/2+\frac{A^*_{max}}{\lambda}\log(n)$$

Using the loss function convexity and taking expectation

$$\sum_{t=1}^{T}\mathbb{E}\left[L_t(X_t)-L_t(X^*)+\frac{\theta}{\eta}(B_{max}-B_t)(A_{t+1}-A^*)\right]\\ \leq(\eta+\lambda A^*_{max})\frac{G^2}{2}T+\frac{1}{\eta}(A^*_{max}-A_{min})^2/2+\frac{A^*_{max}}{\lambda}\log(n) \tag{41}$$

which can be written as

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}\left[L_t(X_t)-L_t(X^*)\right]\\ &\quad\leq(\eta+\lambda A^*_{max})\frac{G^2}{2}T+\frac{1}{\eta}(A^*_{max}-A_{min})^2/2+\frac{A^*_{max}}{\lambda}\log(n)\\ &\qquad+\frac{\theta}{\eta}\sum_{t=1}^{T}\mathbb{E}\left[(B_t-B_{max})(A_{t+1}-E_{t+1}+E_{t+1}-A^*)\right]\end{aligned}\tag{42}$$

Now consider the following two lemmas.

Lemma 5.1.

$$\textstyle\sum_{t=1}^{T}\mathbb{E}\left[(B_t-B_{max})(E_{t+1}-A^*)\right]\leq 0 \tag{43}$$

Proof. Since E_{t+1} is independent of B_t we have

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[(B_t-B_{max})(E_{t+1}-A^*)\right]&=\sum_{t=1}^{T}\mathbb{E}\left[B_t-B_{max}\right]\mathbb{E}\left[E_{t+1}-A^*\right]\\ &\leq\sum_{t=1}^{T}\mathbb{E}\left[B_t-B_{max}\right]\left(\mathbb{E}\left[E_{t+1}\right]-\mathbb{E}\left[E_{t+1}\right]\right)\\ &=0\end{aligned}$$

where the inequality uses $A^* = \sum_{i=1}^{n} X^*(i) \leq \overline{E} = \mathbb{E}[E_t]$ (by (34)) and $B_t \leq B_{max}$ (as is clear by the update equation (2)). □

For convenience, define a constant C by

$$C = \max((E_{max} - A_{min})^2, (A_{max} - E_{min})^2) \tag{44}$$

Lemma 5.2. *We have*

$$\sum_{t=1}^{T} (B_t - B_{max})(A_{t+1} - E_{t+1}) \leq \tfrac{T}{2} C + {B_{max}}^2 \tag{45}$$

Proof. From equation (2) we have

$$B_{t+1} - B_{max} = \min\{B_t - B_{max} + E_{t+1} - A_{t+1}, 0\}$$

Since $\min\{x, 0\}^2 \leq x^2$ for all $x \in \mathbb{R}$ we have

$$\begin{aligned}(B_{t+1} - B_{max})^2 &\leq (B_t - B_{max})^2 + (E_{t+1} - A_{t+1})^2 + 2(B_t - B_{max})(E_{t+1} - A_{t+1}) \\ &\leq (B_t - B_{max})^2 + C + 2(B_t - B_{max})(E_{t+1} - A_{t+1})\end{aligned}$$

By summing over $t \in \{1, \ldots, T\}$ we obtain

$$(B_{T+1} - B_{max})^2 \leq (B_1 - B_{max})^2 + TC + 2\sum_{t=1}^{T} (B_t - B_{max})(E_{t+1} - A_{t+1})$$

$$0 \leq (B_1 - B_{max})^2 + TC + 2\sum_{t=1}^{T} (B_t - B_{max})(E_{t+1} - A_{t+1})$$

$$\sum_{t=1}^{T} (B_t - B_{max})(A_{t+1} - E_{t+1}) \leq \frac{T}{2} C + {B_{max}}^2$$

The two last inequalities use the fact that $x^2 \geq 0$ and $B_1 = 0$. □

Substituting inequalities (43) and (45) in the main equation (42) gives

$$\begin{aligned}\sum_{t=1}^{T} \mathbb{E}[L_t(X_t) - L_t(X^*)] \leq (\eta + \lambda A^*_{max}) \frac{G^2}{2} T + \frac{1}{\eta} (A^*_{max} - A_{min})^2 / 2 \\ + \frac{A^*_{max}}{\lambda} \log(n) + \frac{\theta}{\eta} (\frac{T}{2} C + {B_{max}}^2)\end{aligned} \tag{46}$$

where B_{max} also depends on θ and η and is given from (33):

$$B_{max} = \tfrac{\eta}{\theta} G + \frac{\sqrt{(A_{max} - E_{min})(A_{max} - A_{min})}}{\sqrt{\theta}} - E_{min} + A_{min}$$

So we have the main theorem:

Theorem 5.3. *If the Algorithm 1 run with parameters θ, η and λ, and the battery capacity be the optimal value given by Eq. 33, then the regret will be:*

$$\begin{aligned}Regret(T) \leq (\eta + \lambda A^*_{max}) \frac{G^2}{2} T + \frac{1}{\eta} (A^*_{max} - A_{min})^2 / 2 \\ + \frac{A^*_{max}}{\lambda} \log(n) + \frac{\theta}{\eta} (\frac{T}{2} C + {B_{max}}^2)\end{aligned} \tag{47}$$

5.4 Choosing the parameters

The regret bound in the right-hand-side of (47) can be minimized over all parameter choices $\eta > 0$, $\theta > 0$, and $\lambda > 0$. The optimized λ is

$$\lambda^* = \sqrt{\frac{2\log(n)}{G^2 T}} \tag{48}$$

However, optimizing η and θ is not as clean and is most easily done by a numerical search to minimize the right-hand-side of (47). To illustrate asymptotic tradeoffs of $O(\sqrt{T})$ and $O(\sqrt{\log(n)})$ we provide the following example sizings of η and θ that arise from optimizing only the first two terms in the right-hand-side of (47):

$$\eta^* = \frac{(A^*_{max} - A_{min})}{G\sqrt{T}}$$

and

$$\theta^* = G\eta^* \sqrt{\frac{2}{TC}} = \sqrt{\frac{2}{C}} \frac{(A^*_{max} - A_{min})}{T}$$

where C is defined in (44). So the regret is $O(\sqrt{T \ln(n)})$ and the required battery capacity B_{max} is $O(\sqrt{T})$ (with no dependence on n).

5.5 Implementing with large batteries and beyond slot T

If the battery storage device has physical capacity larger than the B_{max} value specified in (33), we can still set the algorithm parameter to this B_{max} value. Then, we partition the battery storage to a unit of size B_{max} that is used only for this algorithm and a remaining unit that can store energy for other purposes.

The algorithm in this paper was described over a finite time horizon $t \in \{1, \ldots, T\}$ to clearly illustrate the regret properties and battery requirements. Of course, the algorithm can run forever (beyond slot T). In that case we use T only to size the parameters of Algorithm 1, but we can run the algorithm for a time arbitrarily larger than T. Using similar analysis, it can be shown that similar $O(\sqrt{nT})$ regret properties hold over any consecutive T slots of the sample path. The analysis of this fact is similar and is omitted for brevity (the only significant difference is that the initial condition at the start of the T-slot path is no longer zero).

5.6 Discussion on change of variable

Our algorithm takes the convex objective $L_t(X)$ and turns it to a function $g_t(A, P) = L_t(A.P)$ that involves a non-convex multiplication of variables. However, our algorithm chooses each variable separately by solving a separate optimization problem and the function $g_t(A, P)$ is convex with respect to A or P (while it is not jointly convex). This approach still works because we exploit convexity in each separate variable A, P, and also convexity of the original function L_t. The separation is needed to overcome key challenges.

6 SIMULATION

In this section we simulate the example explained in Section 1.1 for a case with $n = 100$ channels. We generate the noise levels in this specific simulation from a Markovian random walk. Specifically, define $Z_t(i) = S_t(i)/N_t(i)$. For each $i \in \{1, \ldots, 100\}$ we generate $\{Z_t(i)\}_{t=1}^T$ as an independent random walk inside the interval $[0, 1/N_{min}]$ where the borders are reflective and the steps are i.i.d samples of a mean zero Gaussian distribution with variance $\frac{1}{10000 N_{min}}$. The initial condition is $Z_0(i) = \frac{1}{2N_{min}}$ for all $i \in \{1, \cdots, n\}$.

The following two simulations use the same objective function described above. To summarize we have:

- The steps of each random walk $Z_t(i)$ are i.i.d samples of a mean zero Gaussian distribution with variance $\frac{1}{10000 N_{min}}$. The constant 10000 is chosen in a way to make the random walk be more "continuous" while it is still able to reflect off the boundary several times during the simulation.
- $Z_t(i) = I\left(Z_0(i) + \sum_{\tau=1}^{t} R_\tau(i)\right)$ (for all $t \in \{1, \cdots, T\}$ and $i \in \{1, \cdots, n\}$) where function $I(\cdot)$ is defined:
$$I(x) = \frac{2}{N_{min}} \left| \frac{x N_{min}}{2} - \left\lfloor \frac{x N_{min}}{2} + \frac{1}{2} \right\rfloor \right|$$
- $L_t(x) = -\sum_{i=1}^{n} \log(1 + Z_t(i) x(i))$ for all $x \in \mathcal{X}$.
- The gradient upper bound is
$$\left| \frac{\partial}{\partial x(i)} L_t(x) \right| = \left| \frac{Z_t(i)}{1 + Z_t(i) x(i)} \right| \leq \frac{1}{N_{min} + A_{min}} = G$$
- $A_{min} = 0$, $A_{max} = 2$, $N_{min} = 1$, $T = 10000$, $n = 100$.
- The parameter λ is chosen by (48). The parameters $\eta > 0, \theta > 0$ are chosen by numerically minimizing the regret bound (47).

6.1 Algorithm with optimal battery vs. Algorithm with lowered battery

We compare two versions of Algorithm 1, one that uses the proposed B_{max} given by (33) (for which analytical guarantees are proven) and the other using a heuristically chosen value $B_{max}/2$. This is to test if the proposed sizing of B_{max}, which was based on a worst-case analysis that ensured the battery avoids the zero state, was overly conservative. Intuitively, if the lowered-battery heuristic does not hit zero very often, then, since its decisions are similar to that of the proposed algorithm, we expect it to have similar regret but with a reduced battery capacity requirement.

In this simulation the input energy (E_t) are i.i.d samples of a uniform distribution over interval $[0, 1]$ so $E_{min} = 0$, $E_{max} = 1$, and $\overline{E} = \frac{1}{2}$

There are four figures. Fig. 1 shows the sample path random walk of the "inverse noise" $Z_t(i)$ for the first three channels $i \in \{1, 2, 3\}$ from the 100 channels. Figs. 2, 3, and 4 plot results for the two different algorithms.[3] Both algorithms show regrets that converge to values smaller than zero, and so both are significantly better than the best fixed-decision policy (see Fig. 2). The vector $X^* = [X^*(1), \ldots, X^*(n)]$ for the best fixed-decision policy (used in Fig. 2) was computed offline with full knowledge of the $L_t(\cdot)$ functions by minimizing
$$\sum_{t=1}^{T} L_t(X^*)$$
over all X^* that satisfy (34)-(35). Thus, Fig. 2 plots the sample-path regret:
$$Regret(t) = \frac{1}{t} \sum_{\tau=1}^{t} L_\tau(X_\tau) - \frac{1}{t} \sum_{\tau=1}^{t} L_\tau(X^*)$$
Remarkably, from Fig. 2 it can be seen that the heuristic "lowered-battery" algorithm gets (slightly) better regret. This is likely because the proposed algorithm is making more conservative decisions in order to increase the battery level (which starts at $B_0 = 0$) to values closer to B_{max} (rather than $B_{max}/2$). Of course, only the proposed algorithm comes with the analytical performance guarantees established in previous sections. In Fig. 3 the B_t is pictured over time for both algorithms. It can be seen that the proposed algorithm never meets $B_t = 0$ while the lowered-battery algorithm goes to zero multiple times. The amplitude of output energy is shown in Fig. 4. It is clear from Fig. 4 that both algorithms have an average output power equal to the expected energy arrival per slot ($\overline{E} = 1/2$). The proposed algorithm is far more stable on the output level while the lowered battery algorithm shows significantly larger time variation.

[3]Fig. 2 has been updated due to an error in this figure in the original published paper. The new figure is different but does not qualitatively change the results.

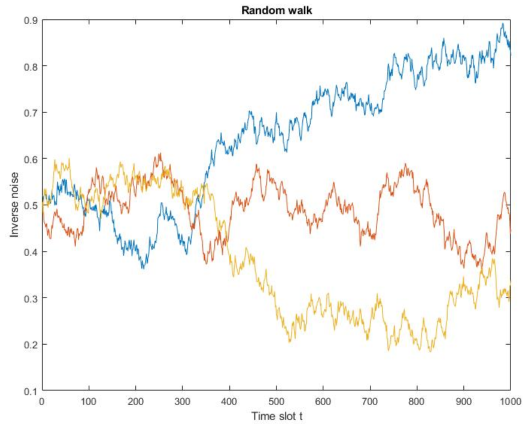

Fig. 1. The inverse noise level for channels 1,2, and 3 from all 100 channels

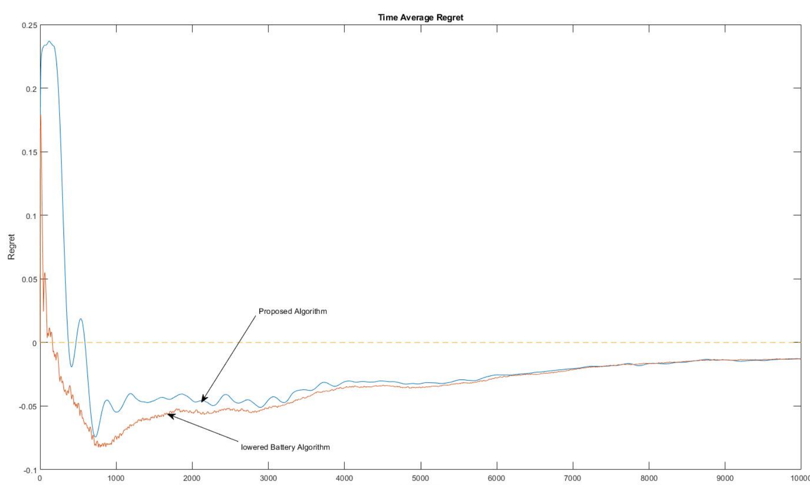

Fig. 2. Regret versus time for proposed algorithm and lowered battery algorithm

6.2 Real life non-i.i.d energy input

In this simulation, the proposed algorithm with a optimal battery receives the real life energy output of a solar cell. We used the data provided by [29] [4] . The Fig. 5 shows the energy delivered by the solar cell. The Figs. 6,7, 8 show the results of the simulation. As the input energy is pretty much periodic, the battery level and the total output energy are also semi periodic. While the input energy is completely non-i.i.d, still the algorithm managed to keep the battery non-zero all the time and at the same time providing a satisfying regret.

[4]The first $T = 10000$ steps of the file "Actual_32.95_-115.15_2006_UPV_100MW_5_Min" has been used. We also normalised the data by dividing it by two times its average so energy input has the same average as the first simulation.

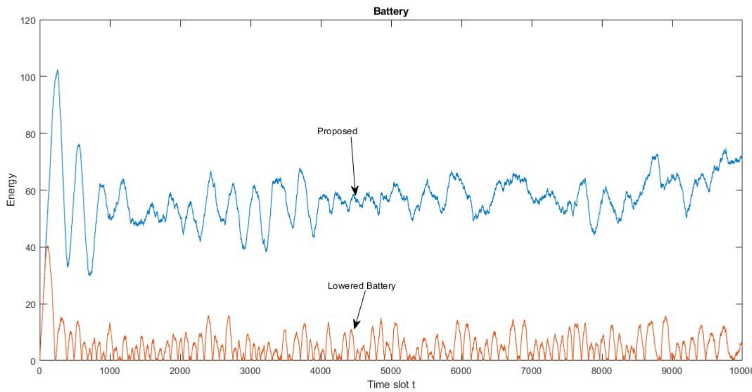

Fig. 3. Battery level versus time for proposed algorithm and lowered battery algorithm

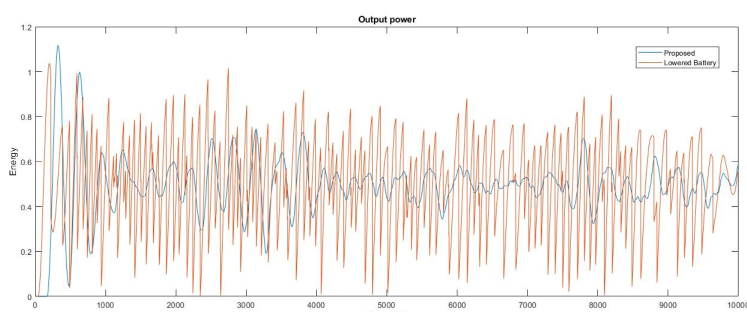

Fig. 4. The output power amplitude versus time for proposed algorithm and lowered battery algorithm

7 CONCLUSION

This paper develops an efficient method for online convex optimization (OCO) with energy harvesting constraints. This is a generalization of OCO problems where resource allocations are restricted by the amount of energy currently stored in a battery, which depends on the amount of energy used in the past. Our paper focuses on applications to energy-constrained wireless transmission problems. An algorithm was developed that achieves regret that grows like $O(\sqrt{T})$, which is known to be optimal (the square root law cannot be improved even for simpler unconstrained OCO problems). Further, our algorithm improves state-of-the-art from $O(n)$ dependence on the dimension (number of wireless channels) to $O(\sqrt{\log(n)})$ dependence. This achievement is significant and nontrivial. To accomplish this, we used a separation of decisions into an amplitude component and a direction component, a Lyapunov drift term, and two distinct Bregman divergence functions. These techniques can likely be used to design efficient scheduling policies in other OCO contexts.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation grant NSF CCF-1718477.

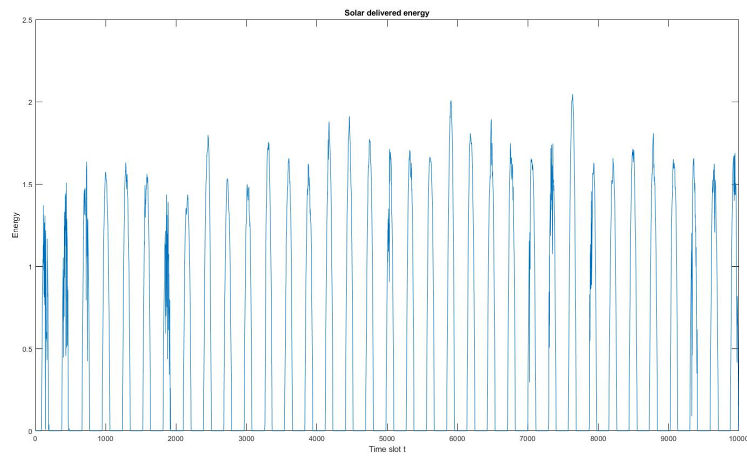

Fig. 5. The energy provided by the solar cell for the transmitter versus time, which is completely non-i.i.d

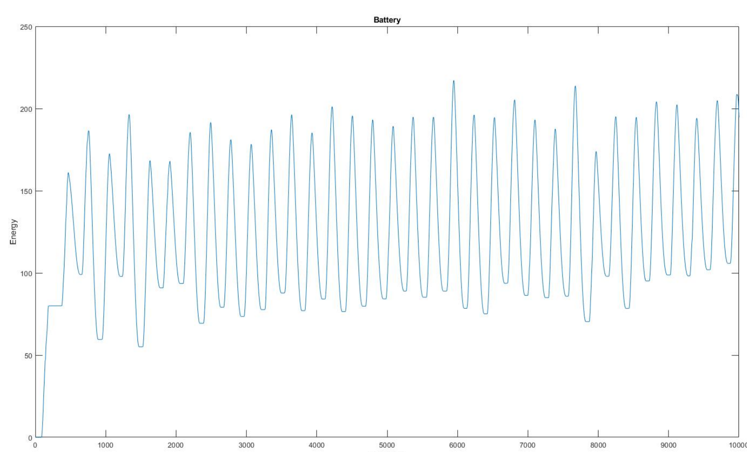

Fig. 6. Battery level versus time for proposed algorithm with non-i.i.d real life energy input

REFERENCES

[1] Jacob D Abernethy, Elad Hazan, and Alexander Rakhlin. 2009. Competing in the dark: An efficient algorithm for bandit linear optimization. (2009).

[2] Ahmed Arafa, Abdulrahman Baknina, and Sennur Ulukus. 2017. Energy harvesting networks with general utility functions: Near optimal online policies. In *2017 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 809–813.

[3] Amir Beck and Marc Teboulle. 2003. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters* 31, 3 (2003), 167–175.

[4] Pol Blasco, Deniz Gunduz, and Mischa Dohler. 2013. A learning theoretic approach to energy harvesting communication system optimization. *IEEE Transactions on Wireless Communications* 12, 4 (2013), 1872–1882.

[5] Sébastien Bubeck. 2011. Introduction to online optimization. *Lecture Notes* 2 (2011).

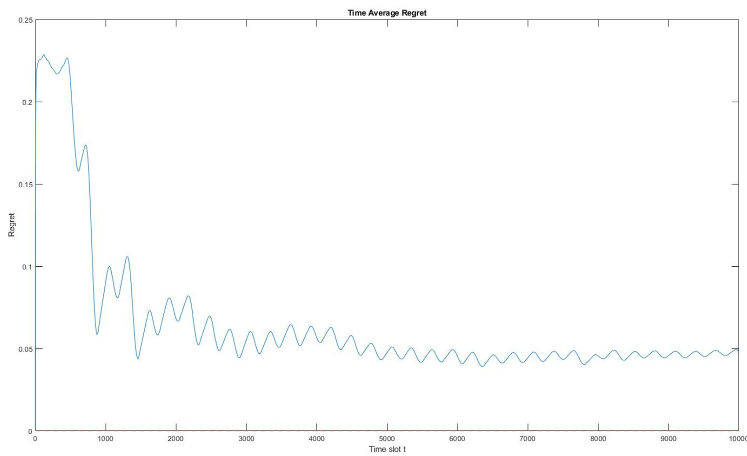

Fig. 7. Regret versus time for proposed algorithm with non-i.i.d real life energy input

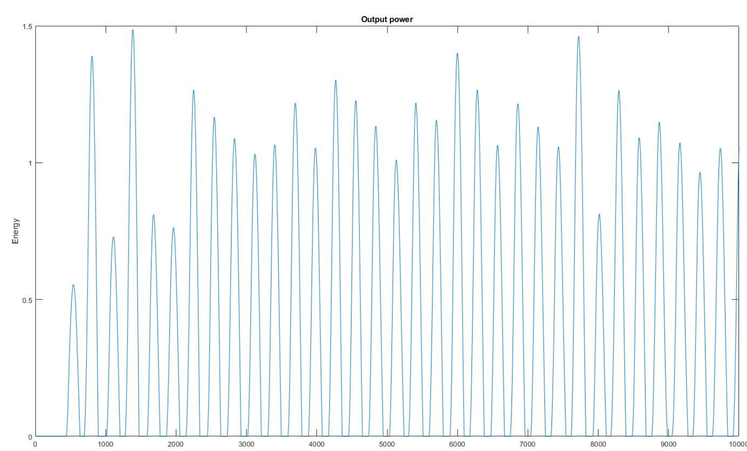

Fig. 8. The output power amplitude versus time for proposed algorithm with non-i.i.d real life energy input

[6] Sébastien Bubeck and Nicolo Cesa-Bianchi. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721* (2012).

[7] Ying Cao, Bo Sun, and Danny HK Tsang. 2020. Optimal Online Algorithms for One-Way Trading and Online Knapsack Problems: A Unified Competitive Analysis. *arXiv preprint arXiv:2004.10358* (2020).

[8] Nicolo Cesa-Bianchi, Philip M Long, and Manfred K Warmuth. 1996. Worst-case quadratic loss bounds for prediction using linear functions and gradient descent. *IEEE Transactions on Neural Networks* 7, 3 (1996), 604–619.

[9] Nicolo Cesa-Bianchi and Gábor Lugosi. 2006. *Prediction, learning, and games.* Cambridge university press.

[10] Chi-Kin Chau, Guanglin Zhang, and Minghua Chen. 2016. Cost minimizing online algorithms for energy storage management with worst-case guarantee. *IEEE Transactions on Smart Grid* 7, 6 (2016), 2691–2702.

[11] G. Chen and M. Teboulle. 1993. Convergence Analysis of a Proximal-Like Minimization Algorithm Using Bregman Functions. *SIAM Journal on Optimization* 3, 3 (1993), 538–543.

[12] Tianyi Chen and Georgios B Giannakis. 2018. Bandit convex optimization for scalable and dynamic IoT management. *IEEE Internet of Things Journal* 6, 1 (2018), 1276–1286.

[13] Ran El-Yaniv, Amos Fiat, Richard M Karp, and Gordon Turpin. 2001. Optimal search and one-way trading online algorithms. *Algorithmica* 30, 1 (2001), 101–139.

[14] M. Gatzianas, L. Georgiadis, and L. Tassiulas. Feb. 2010. Control of Wireless Networks with Rechargeable Batteries. *IEEE Transactions on Wireless Communications* vol. 9, no. 2, pp. 581-593 (Feb. 2010).

[15] Elad Hazan. 2019. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207* (2019).

[16] Elad Hazan, Amit Agarwal, and Satyen Kale. 2007. Logarithmic regret algorithms for online convex optimization. *Machine Learning* 69, 2-3 (2007), 169–192.

[17] Elad Hazan and Satyen Kale. 2010. Extracting certainty from uncertainty: Regret bounded by variation in costs. *Machine learning* 80, 2-3 (2010), 165–188.

[18] L. Huang and M. J. Neely. Aug. 2013. Utility Optimal Scheduling in Energy Harvesting Networks. *IEEE/ACM Transactions on Networking* vol. 21, no. 4, pp. 1117-1130 (Aug. 2013).

[19] Rodolphe Jenatton, Jim Huang, and Cédric Archambeau. 2016. Adaptive algorithms for online convex optimization with long-term constraints. In *International Conference on Machine Learning*. 402–411.

[20] Nikolaos Liakopoulos, Apostolos Destounis, Georgios Paschos, Thrasyvoulos Spyropoulos, and Panayotis Mertikopoulos. 2019. Cautious regret minimization: Online optimization with long-term budget constraints. In *International Conference on Machine Learning*. 3944–3952.

[21] Qiulin Lin, Hanling Yi, John Pang, Minghua Chen, Adam Wierman, Michael Honig, and Yuanzhang Xiao. 2019. Competitive online optimization under inventory constraints. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 3, 1 (2019), 1–28.

[22] Mehrdad Mahdavi, Rong Jin, and Tianbao Yang. 2012. Trading regret for efficiency: online convex optimization with long term constraints. *The Journal of Machine Learning Research* 13, 1 (2012), 2503–2528.

[23] Shie Mannor, John N Tsitsiklis, and Jia Yuan Yu. 2009. Online Learning with Sample Path Constraints. *Journal of Machine Learning Research* 10, 3 (2009).

[24] Nicolo Michelusi, Kostas Stamatiou, and Michele Zorzi. 2013. Transmission policies for energy harvesting sensors with time-correlated energy supply. *IEEE Transactions on Communications* 61, 7 (2013), 2988–3001.

[25] M. J. Neely. 2010. *Stochastic Network Optimization with Application to Communication and Queueing Systems.* Morgan & Claypool.

[26] M. J. Neely and L. Huang. Dec. 2010. Dynamic Product Assembly and Inventory Control for Maximum Profit. *Proc. IEEE Conf. on Decision and Control (CDC)* Atlanta, GA (Dec. 2010).

[27] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. 2009. Robust Stochastic Approximation Approach to Stochastic Programming. *SIAM Journal on Optimization* 19, 4 (2009), 1574–1609.

[28] Arkadiĭ Nemirovsky. [n.d.]. Problem complexity and method efficiency in optimization. ([n. d.]).

[29] Michael Rossol, Bri-Mathias Hodge, Caroline Draxl, Andrew Clifton, Jim McCaa, Tarek Elgindy, Manajit Sengupta, Yu Xie, Anthony Lopez, and Aron Habte. [n.d.]. NREL Renewable Energy Resource Data. ([n. d.]). https://doi.org/10.17041/drp/1473618

[30] Shai Shalev-Shwartz et al. 2011. Online learning and online convex optimization. *Foundations and trends in Machine Learning* 4, 2 (2011), 107–194.

[31] Shai Shalev-Shwartz and Yoram Singer. 2007. A primal-dual perspective of online learning algorithms. *Machine Learning* 69, 2-3 (2007), 115–142.

[32] Dor Shaviv and Ayfer Özgür. 2016. Universally near optimal online power control for energy harvesting nodes. *IEEE Journal on Selected Areas in Communications* 34, 12 (2016), 3620–3631.

[33] Alexander A Titov, Fedor S Stonyakin, Alexander V Gasnikov, and Mohammad S Alkousa. 2018. Mirror descent and constrained online optimization problems. In *International Conference on Optimization and Applications.* Springer, 64–78.

[34] Paul Tseng. 2008. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization* 2, 3 (2008).

[35] Kaya Tutuncuoglu and Aylin Yener. 2012. Optimum transmission policies for battery limited energy harvesting nodes. *IEEE Transactions on Wireless Communications* 11, 3 (2012), 1180–1189.

[36] Xiaohan Wei, Hao Yu, and Michael J Neely. 2020. Online primal-dual mirror descent under stochastic constraints. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems.* 3–4.

[37] Weiwei Wu, Jianping Wang, Xiumin Wang, Feng Shan, and Junzhou Luo. 2016. Online throughput maximization for energy harvesting communication systems with battery overflow. *IEEE Transactions on Mobile Computing* 16, 1 (2016), 185–197.

[38] Jing Yang and Sennur Ulukus. 2011. Optimal packet scheduling in an energy harvesting communication system. *IEEE Transactions on Communications* 60, 1 (2011), 220–230.

[39] Lin Yang, Mohammad H Hajiesmaili, Ramesh Sitaraman, Adam Wierman, Enrique Mallada, and Wing S Wong. 2020. Online Linear Optimization with Inventory Management Constraints. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 4, 1 (2020), 1–29.

[40] Hao Yu, Michael Neely, and Xiaohan Wei. 2017. Online convex optimization with stochastic constraints. In *Advances in Neural Information Processing Systems*. 1428–1438.

[41] Hao Yu and Michael J Neely. 2019. Learning-Aided Optimization for Energy-Harvesting Devices With Outdated State Information. *IEEE/ACM Transactions on Networking* 27, 4 (2019), 1501–1514.

[42] Hao Yu and Michael J. Neely. 2020. A Low Complexity Algorithm with $O(\sqrt{T})$ Regret and $O(1)$ Constraint Violations for Online Convex Optimization with Long Term Constraints. *Journal of Machine Learning Research* 21, 1 (2020), 1–24. http://jmlr.org/papers/v21/16-494.html

[43] Jianjun Yuan and Andrew Lamperski. 2018. Online convex optimization for cumulative constraints. In *Advances in Neural Information Processing Systems*. 6137–6146.

[44] Martin Zinkevich. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*. 928–936.

Received August 2020; revised September 2020; accepted October 2020