2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Efficient Online Learning with Memory via Frank-Wolfe Optimization: Algorithms with Bounded Dynamic Regret and Applications to Control

Hongyu Zhou, Zirui Xu, Vasileios Tzoumas

***Abstract*— Projection operations are a typical computation bottleneck in online learning. In this paper, we enable projection-free online learning within the framework of *Online Convex Optimization with Memory* (OCO-M) —OCO-M captures how the history of decisions affects the current outcome by allowing the online learning loss functions to depend on both current and past decisions. Particularly, we introduce a projection-free meta-base learning algorithm with memory that minimizes dynamic regret, *i.e.*, that minimizes the suboptimality against *any* sequence of time-varying decisions. We are motivated by applications where autonomous agents need to adapt to time-varying environments in real-time, accounting for how past decisions affect the present. Examples of such applications are: online control of dynamical systems; statistical arbitrage; and time series prediction. The algorithm builds on the Online Frank-Wolfe (OFW) and Hedge algorithms. We demonstrate how our algorithm can be applied to the online control of linear time-varying systems in the presence of unpredictable process noise. To this end, we develop a projection-free OCO-M controller with bounded dynamic regret against any optimal time-varying linear feedback control policy. We validate our algorithm in simulated scenarios of online control of linear time-invariant systems.**

I. Introduction

Online Convex Optimization (OCO) [1], [2] has found widespread application in statistics, information theory, and operation research [3]. OCO can be interpreted as a sequential game between an optimizer and an adversary over T time steps: at each time step $t = 1, \dots, T$, first the optimizer chooses a decision $\mathbf{x}_t$ from a convex set $\mathcal{X}$; then, the adversary reveals a convex loss function f_t and the optimizer suffers the loss $f_t(\mathbf{x}_t)$. The optimizer aims to minimize cumulative loss, despite knowing f_t only after $\mathbf{x}_t$ has been decided.

Static regret is the standard approach to measure the suboptimality of the optimizer's decisions $\mathbf{x}_1, \dots, \mathbf{x}_T$. Particularly, given a decision $\mathbf{x} \in \mathcal{X}$ to compare $\mathbf{x}_1, \dots, \mathbf{x}_T$ with, the static regret of $\mathbf{x}_1, \dots, \mathbf{x}_T$ with respect to $\mathbf{v}$ is defined as follows [2]:

$$\text{S-Reg}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{v}). \tag{1}$$

That is, when $\mathbf{v}$ minimizes $\sum_{t=1}^{T} f_t(\mathbf{v})$, then S-Reg_T captures the suboptimality of $\mathbf{x}_1, \dots, \mathbf{x}_T$ against the optimal *static* decision that would have been made in hindsight.

Algorithms that guarantee *static no-regret* have been widely adopted in applications pertained to recommendation systems and communication-channel allocation [3].[1]

But the application of such algorithms to complex artificial intelligence tasks such as *online control under unpredictable disturbances* [4] and *collaborative multi-robot motion planning* [5] is hindered by three main technological challenges:

- **Challenge I: Dynamic Environments.** Complex tasks such as the above require decisions that adapt to changing environments. For example, *target tracking with multiple robots* requires the robots to continuously change their position to track moving targets [5]. Therefore, measuring performance against a static (optimal) decision per eq. (1) is insufficient. Instead, we need to measure performance against *time-varying* (optimal) decisions.
- **Challenge II: Past Decisions Affect the Present.** In complex tasks, past decisions often affect the present outcome. Therefore, the OCO framework we discussed above, where each loss function f_t depends on the most recent decision $\mathbf{x}_t$ only, fails to capture the effect of earlier decisions to the present. Instead, we need an OCO framework with memory, where each loss function f_t depends on $\mathbf{x}_t$ as well as on the past $\mathbf{x}_{t-m}, \dots, \mathbf{x}_{t-1}$, for some $m \geq 0$.
- **Challenge III: Fast Decision-Making.** Complex control tasks often require decisions to be made fast. For example, such is the case for the effective *online control of quadrotors against wind disturbances* [6]. But the current OCO algorithms typically rely on projection operations which can be computationally expensive since they require solving quadratic programs [7]. Instead, we need fast OCO algorithms that are inevitably projection-free.

The above challenges give rise to the need for online learning algorithms for OCO with Memory (OCO-M) that are projection-free and near-optimal in dynamic environments. The decisions' near-optimality may be captured by bounding their suboptimality with respect to optimal decisions that adapt to the changing environment knowing its future evolution, *i.e.*, by bounding *dynamic regret*.

Dynamic regret for the classical OCO without memory is defined as follows [8]: given a time-varying comparator sequence $\mathbf{v}_1, \dots, \mathbf{v}_T$, then[2]

$$\text{D-Reg}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{v}_t). \tag{2}$$

Dynamic regret contrasts static regret: static regret compares $(\mathbf{x}_1, \dots, \mathbf{x}_T)$ against a merely static $\mathbf{v}$. Thus, when $\mathbf{v}_1, \dots, \mathbf{v}_T$ minimize $\sum_{t=1}^{T} f_t(\mathbf{v}_t)$, then D-Reg_T captures the suboptimality of $\mathbf{x}_1, \dots, \mathbf{x}_T$ against the optimal *time-varying* decisions that would have been made in hindsight. Hence, dynamic regret bounds are typically larger than static regret bounds, depending on terms that capture the change of the environment. Such terms are *loss variation* V_T, *gradient*

Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109 USA; {zhouhy, ziruixu, vtzoumas}@umich.edu

[1] An algorithm has *static no-regret* when $\text{S-Reg}_T/T$ tends to 0 when T tends to $+\infty$, implying $f_t(\mathbf{x}_t)$ tends to $f_t(\mathbf{v})$ for t large.

[2] A related measure to dynamic regret is *adaptive regret* [9]. Adaptive regret captures the worst-case static regret on any contiguous time interval. [10] studies the relation of dynamic regret to adaptive regret.

variation D_T, and *path length* C_T:[3]

$$V_T \triangleq \sum_{t=1}^{T} \sup_{\mathbf{x}\in\mathcal{X}} \left|f_t(\mathbf{x}) - f_{t-1}(\mathbf{x})\right|, \tag{3}$$

$$D_T \triangleq \sum_{t=1}^{T} \left\|\nabla f_t\left(\mathbf{x}_t\right) - \nabla f_{t-1}\left(\mathbf{x}_{t-1}\right)\right\|_2^2, \tag{4}$$

$$C_T \triangleq \sum_{t=1}^{T} \left\|\mathbf{v}_t - \mathbf{v}_{t-1}\right\|_2. \tag{5}$$

Dynamic regret for OCO-M with memory m, where the loss function at each time step t takes the form $f_t(\mathbf{x}_{t-m},\ldots,\mathbf{x}_t) : \mathcal{X}^{m+1} \mapsto \mathbb{R}$, is defined as follows:

$$\text{Regret}_T^D = \sum_{t=1}^{T} f_t(\mathbf{x}_{t-m},\ldots,\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{v}_{t-m},\ldots,\mathbf{v}_t), \tag{6}$$

where it is assumed that $\mathbf{x}_{t-m} = \mathbf{0}$ for $t \leq m$.

Contributions. We introduce a projection-free algorithm for OCO-M with bounded dynamic regret (Sections IV and V) —the regret bound is presented in Table I. The algorithm builds on the projection-free algorithms **Hedge** [17] and **OFW** [7], [18]. We apply our algorithm to the online control of linear time-varying systems in the presence of unpredictable noise (Section VI). We thus introduce a projection-free OCO-M controller with bounded dynamic regret against any optimal time-varying linear feedback control gains. Particularly, our comparator class of optimal time-varying linear feedback control gains does not require the a priori knowledge of stabilizing control gains. Instead, the state-of-the-art OCO-M controller by [16] requires a comparator class of optimal time-varying policies with an a priori knowledge of stabilizing control gains.

To enable the aforementioned algorithmic and regret-bound contributions: (i) We analyze dynamic regret of the **OFW** algorithm (Section IV). The analysis enables the state-of-the-art bound in [7, Theorem 1] to hold true for any convex loss functions (see Table I). Instead, [7, Theorem 1] holds true for smooth convex functions only. (ii) We prove that the *Disturbance-Action Control* (DAC) policy [19] —widely used in online non-stochastic control to reduce the online control problem to OCO-M [19]–[21]— is able to approximate time-varying linear feedback controllers (Proposition 2 in [22, Appendix D.5]). Previous results have established that a DAC policy can approximate time-invariant linear feedback controllers only [19], instead of a time-varying controllers.

Numerical Evaluations. We validate our algorithm in simulated scenarios of online control of linear time-invariant systems (Section VI-D with additional experiments available in [22, Appendix E]). We compare our algorithm with **OGD** [8], **Ader** [15], and **Scream** [16] algorithms. Our algorithm is observed 3 times faster than the state-of-the-art OCO-M algorithm **Scream** [16] as system dimension increases, and achieves comparable or better loss performance over all compared algorithms [22, Appendix E].

[3]Obtaining a no-regret algorithm hence requires the growth of the metrics in eqs. (3) to (5) to be sublinear [7], [11], [12]. V_T and D_T are small when the loss function and decisions change slowly.

Organization. Section II reviews the related work. Section III introduces the problem of OCO-M. Section IV presents a projection-free meta-base algorithm (*i.e.*, **Meta-OFW** algorithm) to the problem of OCO-M. Section V develops performance guarantee for **Meta-OFW** algorithm. Section VI presents applications of **Meta-OFW** algorithm with numerical experiments to the online non-stochastic control problem. All proofs are in the Appendix of [22].

II. Related Work

We review the literature by first reviewing *OCO without Memory* and *OCO with Memory*; then, we review *Online Learning for Control via OCO with Memory*.

OCO without Memory. The *OCO without Memory* literature is vast [2]. We here focus on algorithms that guarantee bounded dynamic regret; a representative subset is in Table I.

[15] prove that the optimal dynamic regret for OCO without Memory is $\Omega\left(\sqrt{T(1+C_T)}\right)$, and provide an algorithm matching this bound. The algorithm is based on *Online Gradient Descent* (**OGD**), which is a projection-based algorithm: at each time step t, **OGD** chooses a decision $\mathbf{x}_t$ by first computing an intermediate decision $\mathbf{x}'_t = \mathbf{x}_{t-1} - \eta\nabla f_{t-1}(\mathbf{x}_{t-1})$ —given the previous decision $\mathbf{x}_{t-1}$, the gradient of the previously revealed loss $f_{t-1}(\mathbf{x}_{t-1})$, and a step size $\eta > 0$— and then projects $\mathbf{x}'_t$ back to the feasible convex set $\mathcal{X}$ to output the final decision $\mathbf{x}_t$. This projection operation is often computationally expensive since it requires solving a quadratic program [23]. When the projection operation is indeed computationally expensive, the *Online Frank-Wolfe* (**OFW**) algorithm is employed as a projection-free alternative [18], [24]: **OFW** seeks a feasible descent direction by solving $\mathbf{x}'_{t-1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \langle \nabla f_{t-1}(\mathbf{x}_{t-1}), \mathbf{x}\rangle$ and then updating $\mathbf{x}_t = (1-\eta)\mathbf{x}_{t-1} + \eta\mathbf{x}'_{t-1}$. [7] generalize the **OFW** method to OCO without Memory to achieve a bounded dynamic regret and **OFW** has been observed 20 times faster than **OGD** [7].[4]

OCO with Memory. [16] prove that the optimal dynamic regret for OCO-M is $\Omega(\sqrt{T\,(1+C_T)})$, and provide an algorithm matches this bound based on **OGD**. Earlier works have provided static regret bounds for OCO-M, such as the bound $\mathcal{O}(T^{2/3})$ by [32], and the bound $\mathcal{O}(\sqrt{T})$ by [33]. We provide the first projection-free algorithm for OCO-M that also guarantees bounded dynamic regret.

Online Learning for Control via OCO-M. OCO-M has been recently applied to the control of linear dynamical systems in the presence of adversarial (non-stochastic) noise [1], [19], [34]. The noise is adversarial in the sense that it may adapt to the system's evolution. Generally, the noise can evolve arbitrarily, subject to a given upper bound on its magnitude —the upper bound ensures problem feasibility. Thus, no stochastic model is assumed regarding the noise's evolution, in contrast to classical control that typically assumes Gaussian noise [35].

The current OCO-M algorithms for control prescribe control policies by optimizing linear feedback control gains. The algorithms rely on projection-based methods such as

[4]Additional examples of works utilizing **OFW** for OCO without Memory are: [7], [18], [25]–[30]. Examples of works utilizing **OGD** for OCO without Memory are: [8], [12]–[15], [31].

TABLE I: **Comparison of related work and our work on contributed algorithms with bounded dynamic regret bounds for Online Convex Optimization**. GO denotes the number of gradient oracle calls per iteration of the respective algorithm.

Reference	Loss function	Projection-free	Memory	GO	Regret Rate
[8]	Convex	No	No	$\mathcal{O}(1)$	$\mathcal{O}(\sqrt{T(1+C_T)})$
[13]	Convex smooth	No	No	$\mathcal{O}(1)$	$\mathcal{O}\left(\sqrt{(1+D_T)}+\min\left\{\sqrt{(1+D_T)C_T},(1+D_T)^{\frac{1}{3}}T^{\frac{1}{3}}V_T^{\frac{1}{3}}\right\}\right)$
[12]	Strongly convex	No	No	$\mathcal{O}(1)$	$\mathcal{O}(1+C_T)$
[14]	Convex smooth	No	No	$\mathcal{O}(1)$	$\mathcal{O}(C_T)$
[15]	Convex	No	No	$\mathcal{O}(1)$	$\mathcal{O}(\sqrt{T(1+C_T)})$
[7]	Convex smooth	**Yes**	No	$\mathcal{O}(1)$	$\mathcal{O}\left(\sqrt{T}(1+V_T+\sqrt{D_T}\right)$
Ours (Theorem 1)	Convex	**Yes**	No	$\mathcal{O}(1)$	$\mathcal{O}\left(\sqrt{T}(1+V_T+\sqrt{D_T}\right), \mathcal{O}\left(\sqrt{T\left(V_T+D_T\right)}\right)$
[16]	Convex	No	**Yes**	$\mathcal{O}(1)$	$\mathcal{O}(\sqrt{T(1+C_T)})$
Ours (Theorem 2)	Convex	**Yes**	**Yes**	$\mathcal{O}(1)$	$\mathcal{O}(\sqrt{T(1+V_T+D_T+C_T)})$

OGD, and guarantee bounded static regret [19], [21], [34], [36], adaptive regret [20], [37], or dynamic regret [16]. Specifically, the said OCO-M regret bounds are against optimal static feedback control gains with the exception of the bound by [16] which is against a class of optimal time-varying policies; however, the definition of this class requires an a priori knowledge of linear feedback control gains that ensure stability. We provide a projection-free controller with memory and bounded dynamic regret against any optimal time-varying linear feedback control policy, where the optimal time-varying linear feedback control policy does not depend on pre-specified stabilizing feedback control gains, in contrast to [16].

III. Problem Formulation

We define the problem of *Online Convex Optimization with Memory* (OCO-M), along with standard assumptions.

Problem 1 (Online Convex Optimization with Memory (OCO-M) [32])**.** *There exist 2 players, an online optimizer and an adversary, who choose decisions sequentially over a time horizon T. At each time step $t=1,\dots,T$, the online optimizer chooses a decision $\mathbf{x}_t$ from a convex set $\mathcal{X}$; then, the adversary chooses a loss $f_t:\mathcal{X}^{m+1}\mapsto\mathbb{R}$ to penalize the optimizer's most recent $m+1$ decisions. Particularly, the adversary reveals f_t to the optimizer and the optimizer computes its loss $f_t(\mathbf{x}_{t-m},\dots,\mathbf{x}_t)$, where $\mathbf{x}_{t-m}$ is $\mathbf{0}$ for $t\leq m$. The optimizer aims to minimize $\sum_{t=1}^{T}f_t(\mathbf{x}_{t-m},\dots,\mathbf{x}_t)$.*

The challenge in solving OCO-M optimally, *i.e.*, in minimizing $\sum_{t=1}^{T}f_t(\mathbf{x}_{t-m},\dots,\mathbf{x}_t)$, is that the optimizer gets to know f_t only after $\mathbf{x}_t$ has been chosen, instead of before.

Despite the above challenge, our objective is to develop an efficient (projection-free) online algorithm for OCO-M that despite its efficiency still enjoys sublinear dynamic regret.

To achieve our objective, we adopt standard assumptions in online convex optimization [2], [15], [16], [19], [20], [33]:

Assumption 1 (Convex and Compact Bounded Domain, Containing the Origin)**.** *The domain set $\mathcal{X}$ is convex and compact, contains the zero point, and has diameter D, where D is a given non-negative number;* i.e., *$\mathbf{0}\in\mathcal{X}$, and $\|\mathbf{x}-\mathbf{y}\|_2\leq D$ for all $\mathbf{x},\mathbf{y}\in\mathcal{X}$.*

Definition 1 (Unary Loss Function)**.** *Given $f_t:\mathcal{X}^{m+1}\mapsto\mathbb{R}$, the* unary loss function *is the $\widetilde{f}_t(\mathbf{x})\triangleq f_t(\mathbf{x},\dots,\mathbf{x})$.*

Assumption 2 (Convex Loss)**.** *The loss function $f_t:\mathcal{X}^{m+1}\mapsto\mathbb{R}$ is convex,* i.e., *the unary loss function $\widetilde{f}_t(\mathbf{x})$ is convex in $\mathbf{x}$, where m is the memory length, and $\mathbf{x}\in\mathcal{X}$.*

Assumption 3 (Bounded Loss)**.** *The loss function f_t takes values in $[a,a+c]$, where a and c are known non-negative numbers;* i.e., *$0\leq a\leq f_t(\mathbf{x}_0,\dots,\mathbf{x}_m)\leq a+c$, for all $(\mathbf{x}_0,\dots,\mathbf{x}_m)\in\mathcal{X}^{m+1}$ and $t\in\{1,\dots,T\}$.*

Assumption 4 (Coordinate-Wise Lipschitz)**.** *The loss function f_t is coordinate-wise L-Lipschitz, where L is a given non-negative number;* i.e., *$|f_t(\mathbf{x}_0,\dots,\mathbf{x}_m)-f_t(\mathbf{y}_0,\dots,\mathbf{y}_m)|\leq L\sum_{i=0}^{m}\|\mathbf{x}_i-\mathbf{y}_i\|_2$, for all $(\mathbf{x}_0,\dots,\mathbf{x}_m)\in\mathcal{X}^{m+1}$, and $(\mathbf{y}_0,\dots,\mathbf{y}_m)\in\mathcal{X}^{m+1}$, and for all $t\in\{1,\dots,T\}$.*

Assumption 5 (Bounded Gradient)**.** *The gradient norm of $\widetilde{f}_t$ is at most G, where G is a given non-negative number;* i.e., *$\left\|\nabla\widetilde{f}_t(\mathbf{x})\right\|_2\leq G$ for all $\mathbf{x}\in\mathcal{X}$ and $t\in\{1,\dots,T\}$.*

Remark 1 (Handling Unknown a and c)**.** *We show how to derive an upper bound for D in Lemma 8 in [22, Appendix D.7]. Then, we are able to also derive bounds for a and c in Assumption 3.*

IV. Meta-OFW Algorithm for OCO-M

We present **Meta-OFW**, a projection-free algorithm with bounded dynamic regret for OCO-M. **Meta-OFW** leverages as subroutine the Online Frank-Wolfe (**OFW**) algorithm, introduced by [7] for the OCO problem without memory. We next first present the **OFW** algorithm (Section IV-A), and then present the **Meta-OFW** algorithm (Section IV-B).

A. Online Frank-Wolfe Algorithm for OCO without Memory

We present the **OFW** algorithm (Algorithm 1) along with its dynamic regret analysis (Theorem 1). Particularly, our analysis results in the same regret bound as **OFW**'s state of the art bound in [7, Theorem 1] but under Assumption 1 and Assumption 2 only. Instead, **OFW**'s bound in [7] holds true under the additional assumption of smooth loss functions.

Theorem 1 (Dynamic Regret Bound of **OFW** for OCO with no memory)**.** *Consider the OCO problem with no memory,* i.e., *Problem 1 with $m=0$. Under Assumption 1 and Assumption 2, **OFW** achieves against any sequence of comparators $(\mathbf{v}_1,\dots,\mathbf{v}_T)\in\mathcal{X}^T$ the dynamic regret*

$$\text{Regret}_T^D\leq\mathcal{O}\left(\frac{1+V_T}{\eta}+\sqrt{TD_T}\right). \tag{7}$$

Particularly, when η is chosen such that $\eta=\mathcal{O}\left(1/\sqrt{T}\right)$, then $\text{Regret}_T^D\leq\mathcal{O}\left(\sqrt{T}\left(1+V_T+\sqrt{D_T}\right)\right)$. Further, if we

Algorithm 1: Online Frank-Wolfe Algorithm (**OFW**) [7].

Input: Time horizon T; step size η.
Output: Decision $\mathbf{x}_t$ at each time step $t = 1, \ldots, T$.

1: Initialize $\mathbf{x}_1 \in \mathcal{X}$;
2: **for** each time step $t = 1, \ldots, T$ **do**
3: Suffer a loss $f_t(\mathbf{x}_t)$;
4: Obtain gradient $\nabla f_t(\mathbf{x}_t)$;
5: Compute $\mathbf{x}'_t = \arg\min_{\mathbf{x} \in \mathcal{X}} \langle \nabla f_t(\mathbf{x}_t), \mathbf{x} \rangle$;
6: Update $\mathbf{x}_{t+1} = (1-\eta)\mathbf{x}_t + \eta \mathbf{x}'_t$;
7: **end for**

select $\eta = \sqrt{\frac{c}{b}}$, where b is a tuning parameter satisfying $c < b$, then $\text{Regret}_T^D \leq \mathcal{O}\left(\sqrt{T\left(V_T + D_T\right)}\right)$.

The **OFW** algorithm achieves Theorem 1 by executing the following projection-free steps (Algorithm 1): **OFW** first takes as input the time horizon T and a constant step size η. Then, at each iteration $t = 1, \ldots, T$, **OFW** chooses an $\mathbf{x}_t$, after which the learner suffers a loss $f_t(\mathbf{x}_t)$ and evaluates the gradient $\nabla f_t(\mathbf{x}_t)$ (lines 3-4). Afterwards, **OFW** seeks a direction $\mathbf{x}'_t$ that is parallel to the gradient within the feasible set $\mathcal{X}$ by solving $\arg\min_{\mathbf{x} \in \mathcal{X}} \langle \nabla f_t(\mathbf{x}_t), \mathbf{x} \rangle$ only once per iteration (line 5). Finally, the decision for next iteration is then updated by $\mathbf{x}_{t+1} = (1-\eta)\mathbf{x}_t + \eta \mathbf{x}'_t$ (line 6).

Remark 2 (Efficiency due to only Projection-Free Operations)**.** **OFW** *in Algorithm 1 is projection-free: it finds a descent direction within the feasible set via solving a convex optimization problem with linear objective function* once *per iteration (line 5). Instead,* e.g., **OGD** *requires solving a convex optimization problem with quadratic objective function [8]. Thus,* **OFW** *is more efficient when projections are costly. For example, [7] demonstrates that* **OFW** *is 20 times faster than* **OGD** *in matrix completion scenarios. In the numerical evaluations with applications to online control ([22, Appendix E]), over online non-stochastic control scenarios, we observe that the proposed* **OFW***-based algorithm is about 3 times faster than the* **OGD***-based algorithm (achieving comparable or superior loss performance).*

B. **Meta-OFW** *Algorithm for OCO-M*

We present **Meta-OFW** (Algorithm 2). To this end, we start with the intuition on how Algorithm 2's steps achieve a bounded dynamic regret (the rigorous dynamic regret analysis of **Meta-OFW** is given in Section V).

Algorithm 2 utilizes multiple copies of the **OFW** algorithm as base-learners —each one with a different step size η— and the **Hedge** algorithm [17] as a meta-learner. The multiple copies of **OFW** aim to cope with the a priori unknown loss variation V_T via a trick reminiscent of the "doubling trick" [1], *i.e.*, via covering the spectrum of step sizes such that there exist a step size that approximately minimizes eq. (7) as if V_T was known; and **Hedge** fuses the decisions provided by the base-learners to output a final decision $\mathbf{x}_t$.

We discuss in more detail the role of the base- and meta-learners in Remark 3 and Remark 4 below, respectively. To this end, we use the following notation and definitions:

- $\lambda \triangleq m^2 L$ is a regularizing constant;
- N is the total number of the base-learners;
- $\mathcal{B}_i$ is the i-th base-learner running **OFW** with step size η_i and output $\mathbf{x}_{t,i}$ at each iteration t, where $i \in \{1, \ldots, N\}$;

Algorithm 2: Meta OFW Algorithm (**Meta-OFW**).

Input: Time horizon T; number of base-learners N per eq. (10); step-size pool $\mathcal{H}$ per eq. (11); initial weight of base-learners $\boldsymbol{p}_1$ per eq. (12); learning rate ϵ for meta-algorithm per eq. (13).
Output: Decision $\mathbf{x}_t$ at each time step $t = 1, \ldots, T$.

1: Set $\mathbf{x}_\tau = \mathbf{0}$, $\forall \tau \leq 0$;
2: Initialize $\mathbf{x}_{1,i} \in \mathcal{X}$, $\forall i \in \{1, \ldots, N\}$;
3: **for** each time step $t = 1, \ldots, T$ **do**
4: Receive $\mathbf{x}_{t,i}$ from base-learner $\mathcal{B}_i$ for all i;
5: Output the Decision $\mathbf{x}_t = \sum_{i=1}^N p_{t,i} \mathbf{x}_{t,i}$;
6: Suffer loss $f_t(\mathbf{x}_{t-m}, \ldots, \mathbf{x}_t)$;
7: Observe the loss function $f_t : \mathcal{X}^{m+1} \mapsto \mathbb{R}$;
8: Construct linearized loss
$$g_t(\mathbf{x}) = \left\langle \nabla \widetilde{f}_t\left(\mathbf{x}_t\right), \mathbf{x} \right\rangle;$$
9: Construct the switching-cost-regularized surrogate loss $\boldsymbol{\ell}_t \in \mathbb{R}^N$ with
$$\ell_{t,i} = g_t(\mathbf{x}_{t,i}) + \lambda \left\| \mathbf{x}_{t,i} - \mathbf{x}_{t-1,i} \right\|_2;$$
10: Update the weight of base-learners $\boldsymbol{p}_{t+1} \in \Delta_N$ by
$$p_{t+1,i} = \frac{p_{t,i} e^{-\epsilon \ell_{t,i}}}{\sum_{j=1}^N p_{t,j} e^{-\epsilon \ell_{t,j}}};$$
11: Base-learner $\mathcal{B}_i$ updates $\mathbf{x}_{t+1,i}$ with step size η_i for all i and gradient $\nabla \widetilde{f}_t\left(\mathbf{x}_t\right)$, per Algorithm 1;
12: **end for**

- $g_t(\mathbf{x}) \triangleq \left\langle \nabla \widetilde{f}_t\left(\mathbf{x}_t\right), \mathbf{x} \right\rangle$ is the linearized loss of $\widetilde{f}_t(\mathbf{x}_t)$ over which each base-learner optimizes via the **OFW**;
- $\ell_{t,i} \triangleq g_t(\mathbf{x}_{t,i}) + \lambda \left\| \mathbf{x}_{t,i} - \mathbf{x}_{t-1,i} \right\|_2$ is a surrogate loss associated with the i-th base-learner $\mathcal{B}_i$ —the meta-learner collects $\ell_{t,i}$ for all base-learners, *i.e.*, for all $i \in \{1, \ldots, N\}$, and optimizes $\mathbf{x}_t$ via **Hedge**;
- $p_{t,i}$ is the assigned weight to the i-th base-learner $\mathcal{B}_i$ by **Hedge** —each $p_{t,i}$, $i \in \{1, \ldots, N\}$, is used to output **Meta-OFW**'s final decision $\mathbf{x}_t$ as the weighted sum of base-learners' decisions $\mathbf{x}_{t,i}$; *i.e.*, $\mathbf{x}_t = \sum_{i=1}^N p_{t,i} \mathbf{x}_{t,i}$;
- $\boldsymbol{\ell}_t \in \mathbb{R}^N$ is the vector whose i-th entry is $\ell_{t,i}$;
- $\boldsymbol{p}_t$ is the vector with i-th entry as $p_{t,i}$;
- $\alpha \triangleq 2(a + c)$ is a constant introduced for notational simplicity (a and c are per Assumption 3).

Remark 3 (Unknown Loss Variation V_T Requires Multiple **OFW** Base-Learners)**.** *The multiple* **OFW** *base-learners aim to overcome the challenge of the a priori unknown loss variation V_T. To illustrate this, we first consider that V_T is known a priori, and show that a single* **OFW** *suffices to achieve bounded dynamic regret for OCO-M. Then, we consider that V_T is unknown a priori, and show how multiple base-learners with appropriate step sizes η can approximate the case where V_T is known a priori. To these ends, we leverage the following dynamic regret bound for OCO-M [33, Proof of Theorem 3.1]:*

$$\begin{aligned} \text{Regret}_T^D \leq & \underbrace{\sum_{t=1}^T \widetilde{f}_t\left(\mathbf{x}_t\right) - \sum_{t=1}^T \widetilde{f}_t\left(\mathbf{v}_t\right)}_{\textit{unary cost}} \\ & + \lambda \underbrace{\sum_{t=2}^T \left\| \mathbf{x}_t - \mathbf{x}_{t-1} \right\|_2}_{\textit{switching cost}} + \lambda \underbrace{\sum_{t=2}^T \left\| \mathbf{v}_t - \mathbf{v}_{t-1} \right\|_2}_{\textit{path length}}, \end{aligned} \tag{8}$$

which we can simplify to

$$\text{Regret}_T^D \leq \mathcal{O}\left(\sqrt{T\left(1 + V_T + D_T + C_T\right)}\right), \tag{9}$$

when V_T is known a priori. Assume that $\mathbf{x}_t$ is updated by an **OFW** *algorithm applied to $\widetilde{f}_1, \ldots, \widetilde{f}_T$ with the V_T-dependent step size $\eta_* = \mathcal{O}\left(\sqrt{(1+V_T)/T}\right)$. Then, eq. (9) results from eq. (8) since the three terms in eq. (8) can be bounded as follows: (i) the unary cost can be bounded by eq. (7) where $\eta = \eta_*$; (ii) the switching cost can be bounded by $\eta_* TD$ due to* **OFW***'s line 6 and due to Assumption 1; and (iii) the path length is by definition equal to C_T. Then, an application of the Cauchy-Schwarz inequality completes the proof of eq. (9). All in all, when V_T is known a priori, a single* **OFW** *suffices to achieve bounded dynamic regret for OCO-M.*

But V_T is unknown a priori since it depends on the loss functions, which are unknown a priori. Instead, an upper bound to V_T is known, i.e.*, it holds true that $V_T \leq Tc$ under Assumption 3. Leveraging this, we can approximate the case where V_T is known a priori by employing an appropriate number of* **OFW** *base-learners, each with a different step size, per eq. (10) and eq. (11) below. Intuitively, we can guarantee that way that there exists a base-learner i with step size η_i close to the unknown step size η_* (the full justification of eq. (10) and eq. (11) is given in Theorem 2's proof in [22, Appendix C.2]. The challenge now is to fuse the decisions of the multiple* **OFW** *to a final decision $\mathbf{x}_t$.*

Remark 4 (The Multiple **OFW** Require a **Hedge** Meta-Learner)**.** *The* **Hedge** *meta-learner in* **Meta-OFW** *aims to fuse the decisions of the multiple* **OFW** *base-learners to a final decision $\mathbf{x}_t$. Specifically, the* **OFW** *base-learners provide multiple decisions at each iteration, the $\mathbf{x}_{t,i}$, $i \in \{1, \ldots, N\}$ (line 4 in Algorithm 2). Then,* **Meta-OFW** *utilizes the* **Hedge** *steps in lines 5, 9, and 10 to fuse those decisions to a single decision, aiming to "track" the best base-learner.*

We next formally describe **Meta-OFW**. First, the algorithm specifies the number of base learners, their corresponding step sizes, and their initial weights as follows, respectively:

$$N = \left\lceil \frac{1}{2}\log_2(1 + \frac{Tc}{\alpha}) \right\rceil + 1 = \mathcal{O}(\log T), \tag{10}$$

$$\mathcal{H} = \left\{ \eta_i \mid \eta_i = 2^{i-1}\sqrt{\frac{\alpha}{\lambda TD}} \leq 1, i \in \{1, \ldots, N\} \right\}, \tag{11}$$

$$p_{1,i} = \frac{1}{i(i+1)} \cdot \frac{N+1}{N}, \text{ for any } i \in \{1, \ldots, N\}. \tag{12}$$

Also, **Meta-OFW** sets the meta-learner's learning rate as

$$\epsilon = \sqrt{2/((2\lambda + G)(\lambda + G)D^2 T)}. \tag{13}$$

The dependence on T can be removed by a "doubling trick" [38], similarly to how **Meta-OFW** copes with unknown V_T.

At each iteration $t = 1, \ldots, T$, **Meta-OFW** receives the intermediate decisions $\mathbf{x}_{t,i}$ from all the base-learners $\mathcal{B}_i$, $i \in \{1, \ldots, N\}$ (line 4) to fuse them into a final decision $\mathbf{x}_t = \sum_{i=1}^{N} p_{t,i}\mathbf{x}_{t,i}$ (line 5). Then, **Meta-OFW** suffers a loss of $f_t(\mathbf{x}_{t-m}, \ldots, \mathbf{x}_t)$ (lines 6-7). Afterwards, **Meta-OFW** constructs the linearized loss $g_t(\mathbf{x})$ and switching-cost-regularized loss ℓ_t (lines 8-9). To this end, **Meta-OFW** needs to evaluate only once the gradient $\nabla \widetilde{f}_t(\mathbf{x}_t)$. Finally, the meta-learner and base-learners update the weights $\boldsymbol{p}_{t+1}$ and $\mathbf{x}_{t+1,i}$ for the next iteration (lines 10-11).

V. Dynamic Regret Guarantees of **Meta-OFW**

To present **Meta-OFW**'s dynamic regret bound, we define:

- $D_{T,i} \triangleq \sum_{t=1}^{T} \|\nabla f_t(\mathbf{x}_{t,i}) - \nabla f_{t-1}(\mathbf{x}_{t-1,i})\|_2^2$ is the gradient variation associated with the base-learner i;
- $\bar{D}_T \triangleq \max_{i \in \{1,\ldots,N\}} D_{T,i}$ is the upper bound for $D_{T,i}$.

We present **Meta-OFW**'s dynamic regret bound against any comparator sequence (Theorem 2). Particularly, the bound below holds true, even if the loss variation V_T, gradient variation $\bar{D}_T$, and path length C_T are unknown to **Meta-OFW**.

Theorem 2 (Dynamic Regret Bound of **Meta-OFW**)**.** *For any comparator sequence $(\mathbf{v}_1, \ldots, \mathbf{v}_T) \in \mathcal{X}^T$,* **Meta-OFW** *achieves a dynamic regret Regret_T^D that enjoys the bound:*

$$\mathrm{Regret}_T^D \leq \mathcal{O}\left(\sqrt{T\left(1 + V_T + \bar{D}_T + C_T\right)}\right). \tag{14}$$

The dependency on V_T and $\bar{D}_T$ results from **OFW** being a base-learner in **Meta-OFW**; similar dependencies, due to projection-free subroutines in online algorithms, have been observed in the literature: see, *e.g.*, [7] and the references in Table I. The dependency on $\bar{D}_T$, instead of D_T in Theorem 1, is to upper bound the gradient variation $D_{T,i}$ such that the base-learner i with step size close to the unknown step size $\eta^\star$ (Remark 3) satisfies $D_{T,i} \leq \bar{D}_T$.

The dependency on C_T is due to the time-varying sequence of comparators. [15] proved that any optimal dynamic regret bound for OCO is $\Omega\left(\sqrt{T(1+C_T)}\right)$, and thus the bound necessarily depends on C_T in the worst case.

Remark 5 (Trade-Off of Projection-Free Efficiency with Regret Optimality)**.** *[16] prove that the optimal dynamic regret for OCO-M is $\Omega(\sqrt{T(1+C_T)})$, and provide a projection-based algorithm using* **OGD** *that matches this bound. In contrast,* **Meta-OFW***'s regret bound in Theorem 2 cannot match the bound $\Omega(\sqrt{T(1+C_T)})$ due to the presence of V_T and D_T in eq. (14). But* **Meta-OFW** *is projection-free and thus is more efficient than the* **OGD***-based algorithm in [16] [18]. All in all, the dependence of eq. (14) on V_T and $\bar{D}_T$ is the regret suboptimality cost we pay in this paper to solve OCO-M efficiently via the projection-free* **OFW***.*

VI. Application to Non-Stochastic Control

We apply **Meta-OFW** to the online non-stochastic control problem [19], and present a projection-free controller with memory (Algorithm 3), and with bounded dynamic regret against any linear time-varying feedback control policy (Theorem 3). The results of the numerical evaluations are present in Section VI-D and [22, Appendix E].

A. The Non-Stochastic Control Problem

We consider Linear Time-Varying systems of the form

$$x_{t+1} = A_t x_t + B_t u_t + w_t, \quad t = 0, \ldots, T, \tag{15}$$

where $x_t \in \mathbb{R}^{d_x}$ is the state of the system, $u_t \in \mathbb{R}^{d_u}$ is the control input, and $w_t \in \mathbb{R}^{d_x}$ is the process noise. The system and input matrices, A_t and B_t, respectively, are known.

At each time step t, the controller chooses a control action u_t and then suffers a loss $c_t(x_t, u_t)$. The loss function c_t is

revealed to the controller only after the controller has chosen the control action u_t, similarly to the OCO setting.

Assumption 6 (Convex and Bounded Loss Function with Bounded Gradient)**.** *The cost function $c_t(x_t, u_t) : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \mapsto \mathbb{R}$ is convex in x_t and u_t. Further, when $\|x\|_2 \leq D$, $\|u\|_2 \leq D$ for some $D > 0$, then $|c_t(x, u)| \leq \beta D^2$ and $\|\nabla_x c_t(x, u)\|_2 \leq G_c D, \|\nabla_u c_t(x, u)\|_2 \leq G_c D$, for given positive numbers β and G_c.*

Assumption 7 (Bounded System Matrices and Noise)**.** *The system matrices and noise are bounded,* i.e.*, $\|A_t\|_{\text{op}} \leq \kappa_A$, $\|B_t\|_{\text{op}} \leq \kappa_B$, and $\|w_t\|_2 \leq W$ for given positive numbers κ_A, κ_B, and W, where $\|\cdot\|_{\text{op}}$ is the operator norm.*

Per Assumption 7, we assume no stochastic model for the process noise w_t: the noise may even be adversarial, subject to the bounds prescribed by W.

Problem 2 (Non-Stochastic Control (NSC) Problem)**.** *At each time step $t = 0, \ldots, T$, first a control action u_t is chosen; then, a loss function $c_t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \mapsto \mathbb{R}$ is revealed and the system suffers a loss $c_t(x_t, u_t)$. The goal is to minimize the dynamic policy regret defined below.*

Definition 2 (Dynamic Policy Regret)**.** *We define the* dynamic policy regret *as*

$$\text{Regret-NSC}_T^D = \sum_{t=0}^{T} c_t\left(x_t, u_t\right) - \sum_{t=0}^{T} c_t\left(x_t^*, u_t^*\right), \quad (16)$$

where (i) both sums in eq. (16) are evaluated with the same noise $\{w_0, \ldots, w_T\}$, which is the noise experienced by the system during its evolution per the control input $\{u_0, \ldots, u_T\}$, (ii) $u_t^* = -K_t^* x_t^*$ is the optimal linear feedback control input in hindsight, i.e.*, the optimal input given a priori knowledge of c_t and of the realized w_t, and (iii) x_t^* is the state reached by applying the optimal control inputs $\{u_0^*, \ldots, u_{t-1}^*\}$.*

Reduction to OCO-M. We present the reduction of the non-stochastic control problem to OCO-M, following [19].

Per eq. (15), x_t depends on the control actions chosen in the past, *i.e.*, $\{u_0, \ldots, u_{t-1}\}$, and similarly, the control action u_t depends on x_{t-1}, *i.e.*, $\{u_0, \ldots, u_{t-2}\}$. To reduce the non-stochastic control problem to OCO-M, there are thus 2 challenges: (i) we need a control parameterization such that the cost function $c_t(x_t, u_t)$ is convex in the parameters of the control actions$\{u_0, \ldots, u_{t-1}\}$, since $c_t(x_t, u_t)$ is implicitly a function of $\{u_0, \ldots, u_{t-1}\}$ via u_t; and, similarly, (ii) we need the memory length of $c_t(x_t, u_t)$, *i.e.*, its implicit dependence on the past control inputs $\{u_0, \ldots, u_{t-1}\}$, to stop growing as t increases; that is, we need $c_t(x_t, u_t)$ to instead depend on the most recent control inputs only, in particular, on $\{u_{t-m}, \ldots, u_t\}$ for memory length m. To address these challenges, [19] propose the *Disturbance-Action Control* policy and the notion of *truncated loss*.

Definition 3 (Disturbance-Action Control Policy)**.** *A* Disturbance-Action Control (DAC) *policy $\pi_t(K_t, M_t)$ chooses the control action u_t at state x_t as $u_t = -K_t x_t + \sum_{i=1}^{H} M_t^{[i-1]} w_{t-i}$,*[5] *where we use the notation: H is the chosen memory length of the DAC policy ($H \geq 1$); K_t is sequentially stabilizing [20]; $M_t = (M_t^{[0]}, \ldots, M_t^{[H-1]})$, with $M_t^{[i]} \in \mathbb{R}^{d_u \times d_x}$ being a control gain such that $\left\|M_t^{[i]}\right\|_{\text{op}} \leq \kappa_B \kappa^3 (1-\gamma)^i$; and $w_\tau = 0, \ \forall \tau < 0$.*

Per [20], x_t and u_t are linear in $\{M_0, \ldots, M_t\}$; therefore, the cost function $c_t(x_t, u_t)$ is convex in $\{M_0, \ldots, M_t\}$.

To present the notion of *truncated loss*, we use:

- $x_t\left(M_{0:t-1}\right)$ is the state reached by applying the DAC policy $\{\pi_\tau\left(K_\tau, M_\tau\right)\}_{\tau=0,\ldots,t-1}$;
- $u_t\left(M_{0:t}\right)$ is the control action at state $x_t\left(M_{0:t-1}\right)$ per the DAC policy $\pi_t(K_t, M_t)$;
- $y_t\left(M_{t-1-H:t-1}\right)$ is the state reached from $x_{t-1-H} = 0$ by applying $\{\pi_\tau\left(K_\tau, M_\tau\right)\}_{\tau=t-H-1,\ldots,t-1}$ and experiencing the noise sequence $\{w_\tau\}_{\tau=t-H-1,\ldots,t-1}$;
- $v_t\left(M_{t-1-H:t}\right)$ is the control input that would have been executed if the state at time t was the $y_t\left(M_{t-1-H:t-1}\right)$.

Definition 4 (Truncated Loss)**.** *Given DAC policies $\{\pi_\tau\left(K_\tau, M_\tau\right)\}_{\tau=0,\ldots,t}$ with memory length H, the induced* truncated loss *$f_t : \mathcal{M}^{H+2} \mapsto \mathbb{R}$ is defined as $f_t\left(M_{t-1-H:t}\right) \triangleq c_t\left(y_t\left(M_{t-1-H:t-1}\right), v_t\left(M_{t-1-H:t}\right)\right)$.*

Thereby, the truncated loss $f_t\left(M_{t-1-H:t}\right)$ depends only on the last $H+2$ time steps of the DAC policy. That is, f_t has a fixed memory length $H+2$, for all $t = 1, \ldots, T$.

All in all, Problem 2 can be reduced to OCO-M when the decision variables are the M_t, and the loss functions are the truncated losses $f_t\left(M_{t-1-H:t}\right)$, for all $t = 1, \ldots, T$.

B. **Meta-OFW** *for Online Non-Stochastic Control*

We present **Meta-OFW**'s application to the online non-stochastic control problem (Algorithm 3). Particularly, Algorithm 3 initializes the number of base-learners, their corresponding step sizes, and their initial weights, per the following equations, similarly to **Meta-OFW**:

$$N = \left\lceil \frac{1}{2} \log_2\left(\frac{2\beta D^2 T + \phi}{\sigma}\right) \right\rceil + 1 = \mathcal{O}(\log T), \quad (17)$$

$$\mathcal{H} = \left\{ \eta_i \mid \eta_i = 2^{i-1} \sqrt{\frac{\sigma}{\zeta T D_f}} \leq 1, i \in \{1, \ldots, N\} \right\}, \quad (18)$$

$$p_{0,i} = \frac{1}{i(i+1)} \cdot \frac{N+1}{N}, \text{ for any } i \in \{1, \ldots, N\}, \quad (19)$$

where $\sigma \triangleq 4\beta D^2$, $\phi \triangleq \sigma + 2\beta D^2$, $\zeta \triangleq (H+2)^2 L_f$, and D_f, L_f, G_f defined as in Lemma 9 in [22, Appendix D.7].

The algorithm also sets the step size of meta-learner as

$$\epsilon = \sqrt{2 / \left((2\zeta + G_f)(\zeta + G_f) D_f^2 T\right)}. \quad (20)$$

At each iteration t, Algorithm 3 receives $M_{t,i}$ from all base-learners (line 4). Then, Algorithm 3 calculates $M_t = \sum_{i=1}^{N} p_{t,i} M_{t,i}$ and outputs the control actions $u_t = -K_t x_t + \sum_{i=1}^{H} M_t^{[i-1]} w_{t-i}$ (lines 5-6), after which the cost function is revealed and the algorithm suffers a loss of $c_t(x_t, u_t)$

[5] The DAC policy depends on the past noise, which can be obtained from eq. (15) once the next state is observed; specifically, at time $t+1$, it holds true that $w_t = x_{t+1} - A_t x_t - B_t u_t$.

Algorithm 3: Meta-OFW for Non-Stochastic Control.

Input: Time horizon T; number of base-learners N per eq. (17); step size pool $\mathcal{H}$ per eq. (18); initial weight of base-learners $\boldsymbol{p}_0$ per eq. (19); learning rate ϵ of meta-algorithm per eq. (20).
Output: Control u_t at each time step $t = 1, \ldots, T$.

1: Set $M_\tau = \mathbf{0}$ and $w_\tau = 0$, $\forall \tau < 0$;
2: Initialize $M_{0,i} \in \mathcal{M}$, $\forall i \in \{1, \ldots, N\}$;
3: **for** each time step $t = 0, \ldots, T$ **do**
4: Receive $M_{t,i}$ from base-learner $\mathcal{B}_i$ for all i;
5: Calculate $M_t = \sum_{i=1}^{N} p_{t,i} M_{t,i}$;
6: Output $u_t = -K_t x_t + \sum_{i=1}^{H} M_t^{[i-1]} w_{t-i}$;
7: Observe the loss function $c_t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \mapsto \mathbb{R}$ and suffer the loss $c_t(x_t, u_t)$;
8: Construct the truncated loss $f_t(M_{t-H-1}, \ldots, M_t) : \mathcal{M}^{H+2} \mapsto \mathbb{R}$;
9: Construct the linearized loss
$$g_t(M) = \left\langle \nabla_M \widetilde{f}_t\left(M_t\right), M \right\rangle_{\mathrm{F}};$$
10: Construct the switching-cost-regularized surrogate loss $\boldsymbol{\ell}_t \in \mathbb{R}^N$ with
$$\ell_{t,i} = g_t(M_{t,i}) + \zeta \left\| M_{t,i} - M_{t-1,i} \right\|_{\mathrm{F}};$$
11: Update the weight of base-learners $\boldsymbol{p}_{t+1} \in \Delta_N$ by
$$p_{t+1,i} = \frac{p_{t,i} e^{-\epsilon l_{t,i}}}{\sum_{j=1}^{N} p_{t,j} e^{-\epsilon l_{t,j}}};$$
12: **for** each base-learner $\mathcal{B}_i$ **do**
13: Compute
$$M'_{t,i} = \arg\min_{M \in \mathcal{M}} \left\langle \nabla_M \widetilde{f}_t(M_t), M \right\rangle_{\mathrm{F}};$$
14: Update $M_{t+1,i} = (1 - \eta_i) M_{t,i} + \eta_i M'_{t,i}$;
15: **end for**
16: Observe the state x_{t+1} and calculate the noise $w_t = x_{t+1} - A_t x_t - B_t u_t$;
17: **end for**

(line 7). Next, Algorithm 3 constructs the truncated loss $f_t(M_{t-H-1}, \ldots, M_t)$, linearized loss $g_t(M)$, and switching-cost-regularized loss $\boldsymbol{\ell}_t$ (lines 8-10). The meta-learner and base-learners update the weights $\boldsymbol{p}_{t+1}$ and $M_{t+1,i}$ for the next iteration (lines 11-15). Finally, the noise w_t is calculated when x_{t+1} is observed (line 16).

C. Dynamic Regret Guarantee of Algorithm 3

Theorem 3 (Dynamic Policy Regret Bound of Algorithm 3)**.** *Algorithm 3 ensures that* [6]

$$\text{Regret-NSC}_T^D \leq \tilde{\mathcal{O}}\left(\sqrt{T\left(1 + V_T + \bar{D}_T + C_T\right)}\right). \quad (21)$$

Remark 6 (Novelty of Theorem 3)**.** *Theorem 3 guarantees a dynamic regret bound against an optimal* time-varying linear feedback policy *in hindsight,* i.e.*, against* $\{\pi_\tau(K_\tau^*, 0)\}_{\tau=0,\ldots,t}$*, per Definition 2. This is different than competing against an optimal time-varying DAC policy* $\{\pi_\tau(K_\tau, M_\tau^*)\}_{\tau=0,\ldots,t}$ *with pre-specified stabilizing control gains* K_τ *as in [16], or an optimal time-invariant linear feedback policy* $\{\pi(K^*, 0)\}$ *over the entire horizon or any time interval as in [19], [20], [34], [36], [37]. To achieve this, we show a DAC policy* $\{\pi_\tau(K_\tau, M_\tau)\}_{\tau=0,\ldots,t}$ *can approximate any time-varying linear feedback policy (Proposition 2 in [22, Appendix D.5]), where the optimal time-varying linear feedback control policy does not depend on pre-specified stabilizing feedback control gains, in contrast to [16].*

D. Numerical Evaluations

We evaluate **Meta-OFW** (Algorithm 3) in simulated scenarios of online control of linear time-invariant systems. Additional experiments that demonstrate the computational efficiency of **Meta-OFW** compared to the state-of-the-art OCO-M algorithm [16] are available in [22, Appendix E]). Our code is open-sourced at: https://github.com/UM-iRaL/Non-Stochastic-Control.

Compared Algorithms. We compare **Meta-OFW** with the **OGD** [8], **Ader** [15], and **Scream** [16] algorithms. All algorithms rely on the DAC policy [19]. To run the algorithms, we use the default step-sizes and parameters given in the experiment of [16].

Simulation Setup. We follow the setup as [39] and consider linear systems of the form

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + w_t \\ &= Ax_t + Bu_t + (\Delta_{t,A} x_t + \Delta_{t,B} u_t + \tilde{w}_t), \end{aligned} \quad (22)$$

where $\tilde{w}_t$ and the elements of $\Delta_{t,A}$ and $\Delta_{t,B}$ are sampled from various distributions, specifically, Gaussian, Uniform, Gamma, Beta, Exponential, and Weibull distributions. The loss function has the form $c_t(x_t, u_t) = q_t x_t^\top x_t + r_t u_t^\top u_t$, where q_t and r_t are time-varying weights. Particularly, we consider two cases:

1) Sinusoidal weights defined as

$$q_t = \sin(t/10\pi), \; r_t = \sin(t/20\pi). \quad (23)$$

2) Step weights defined as

$$(q_t, r_t) = \begin{cases} \left(\frac{\log(2)}{2}, 1\right), & t \leq T/5, \\ (1, 1), & T/5 < t \leq 2T/5, \\ \left(\frac{\log(2)}{2}, \frac{\log(2)}{2}\right), & 2T/5 < t \leq 3T/5, \\ \left(1, \frac{\log(2)}{2}\right), & 3T/5 < t \leq 4T/5, \\ \left(\frac{\log(2)}{2}, 1\right), & 4T/5 < t \leq T. \end{cases} \quad (24)$$

Results. We first compare **Meta-OFW** with the **OGD**, **Ader**, and **Scream** algorithms in terms of cumulative loss. The results are summarized in Table III, showing that **Meta-OFW** achieved the lowest cumulative loss across all tested cases, except under gamma distribution with sinusoidal weights; in the best-case —exponential distribution— **Meta-OFW** is 52 times better than **Scream**.

VII. Conclusion

We provided **Meta-OFW** (Algorithm 2), a projection-free algorithm with bounded dynamic regret for OCO-M in time-varying environments (Theorem 2). To develop **Meta-OFW**, we employed the projection-free algorithm **OFW** and **Hedge**. Further, we applied **Meta-OFW** to the online non-stochastic control problem to control linear time-varying systems corrupted with unknown and unpredictable noise (Algorithm 3). We thus developed a (projection-free) OCO-M controller with memory and bounded dynamic regret against any linear time-varying control policy (Theorem 3), instead of against only static linear control policies. To this end, we also

[6] The path length is defined as $C_T \triangleq \sum_{t=2}^{T} \left\| M_{t-1}^* - M_t^* \right\|_{\mathrm{F}}$.

TABLE II: Comparison of the **OGD** [8], **Ader** [15], **Scream** [16], and **Meta-OFW** algorithms in terms of cumulative loss for 10000 time steps. The blue numbers correspond to the best performance and the red numbers correspond to the worse.

Noise Distribution	Sinusoidal Weights (eq. (139))				Step Weights (eq. (140))			
	Meta-OFW	**Scream**	**Ader**	**OGD**	**Meta-OFW**	**Scream**	**Ader**	**OGD**
Gaussian	15625	19725	21052	33574	9496	10704	11453	26790
Uniform	18299	93987	107096	30419	13395	39057	35313	39885
Gamma	16239	16138	18039	17484	9184	61989	75505	45398
Beta	21448	34146	30990	30253	15982	29301	30799	28859
Exponential	10621	254815	252227	28859	4366	227860	204844	53626
Weibull	14068	91474	94040	38549	5623	182887	993734	92341

proved that the DAC policy class [19] can approximate linear time-varying feedback controllers (Proposition 2 in [22, Appendix D.5]).

REFERENCES

[1] S. Shalev-Shwartz *et al.*, "Online learning and online convex optimization," *Foundations and Trends® in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[2] E. Hazan *et al.*, "Introduction to online convex optimization," *Foundations and Trends in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.

[3] N. Cesa-Bianchi and G. Lugosi, *Prediction, learning, and games*. Cambridge university press, 2006.

[4] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural lander: Stable drone landing control using learned dynamics," in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9784–9790.

[5] Z. Xu, H. Zhou, and V. Tzoumas, "Online submodular coordination with bounded tracking regret: Theory, algorithm, and applications to multi-robot coordination," *IEEE Robotics and Automation Letters (RAL)*, 2023.

[6] A. Romero, R. Penicka, and D. Scaramuzza, "Time-optimal online replanning for agile quadrotor flight," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7730–7737, 2022.

[7] D. S. Kalhan, A. S. Bedi, A. Koppel, K. Rajawat, H. Hassani, A. K. Gupta, and A. Banerjee, "Dynamic online learning via frank-wolfe algorithm," *IEEE Transactions on Signal Processing (TSP)*, vol. 69, pp. 932–947, 2021.

[8] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Interna. Conf. on Machine Learning (ICML)*, 2003, pp. 928–936.

[9] E. Hazan and C. Seshadhri, "Adaptive algorithms for online decision problems," in *Electronic colloquium on computational complexity (ECCC)*, vol. 14, no. 088, 2007.

[10] L. Zhang, "Online learning in changing environments," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence (IJCAI)*, 2020, pp. 5178–5182.

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

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

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

[14] T. Yang, L. Zhang, R. Jin, and J. Yi, "Tracking slowly moving clairvoyant: Optimal dynamic regret of online learning with true and noisy gradient," in *International Conference on Machine Learning (ICML)*. PMLR, 2016, pp. 449–457.

[15] L. Zhang, S. Lu, and Z.-H. Zhou, "Adaptive online learning in dynamic environments," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[16] P. Zhao, Y.-X. Wang, and Z.-H. Zhou, "Non-stationary online learning with memory and non-stochastic control," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2022, pp. 2101–2133.

[17] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[18] E. Hazan and S. Kale, "Projection-free online learning," *arXiv preprint:1206.4657*, 2012.

[19] N. Agarwal, B. Bullins, E. Hazan, S. Kakade, and K. Singh, "Online control with adversarial disturbances," in *International Conference on Machine Learning (ICML)*, 2019, pp. 111–119.

[20] P. Gradu, E. Hazan, and E. Minasyan, "Adaptive regret for control of time-varying dynamics," *arXiv preprint:2007.04393*, 2020.

[21] E. Hazan, S. Kakade, and K. Singh, "The nonstochastic control problem," in *Algorithmic Learning Theory (ALT)*, 2020, pp. 408–421.

[22] H. Zhou, Z. Xu, and V. Tzoumas, "Efficient online learning with memory via frank-wolfe optimization: Algorithms with bounded dynamic regret and applications to control," *arXiv preprint arXiv:2301.00497*, 2023.

[23] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, no. 5, pp. 877–898, 1976.

[24] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[25] M. Jaggi, "Revisiting frank-wolfe: Projection-free sparse convex optimization," in *International Conference on Machine Learning*. PMLR, 2013, pp. 427–435.

[26] D. Garber and E. Hazan, "Faster rates for the frank-wolfe method over strongly-convex sets," in *International Conference on Machine Learning*. PMLR, 2015, pp. 541–549.

[27] Y. Wan and L. Zhang, "Projection-free online learning over strongly convex sets," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 10 076–10 084.

[28] B. Kretzu and D. Garber, "Revisiting projection-free online learning: the strongly convex case," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 3592–3600.

[29] D. Garber and B. Kretzu, "New projection-free algorithms for online convex optimization with adaptive regret guarantees," *arXiv preprint arXiv:2202.04721*, 2022.

[30] Y. Wan, L. Zhang, and M. Song, "Improved dynamic regret for online frank-wolfe," *arXiv preprint arXiv:2302.05620*, 2023.

[31] T.-J. Chang and S. Shahrampour, "Unconstrained online optimization: Dynamic regret analysis of strongly convex and smooth problems," *arXiv preprint:2006.03912*, 2020.

[32] M. J. Weinberger and E. Ordentlich, "On delayed prediction of individual sequences," *IEEE Transactions on Information Theory (TIT)*, vol. 48, no. 7, pp. 1959–1976, 2002.

[33] O. Anava, E. Hazan, and S. Mannor, "Online learning for adversaries with memory: price of past mistakes," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, 2015.

[34] M. Simchowitz, K. Singh, and E. Hazan, "Improper learning for non-stochastic control," in *Conference on Learning Theory (COLT)*, 2020, pp. 3320–3436.

[35] K. J. Åström, *Introduction to stochastic control theory*. Courier Corporation, 2012.

[36] Y. Li, S. Das, and N. Li, "Online optimal control with affine constraints," in *AAAI Conference on Artificial Intelligence (AAAI)*, vol. 35, no. 10, 2021, pp. 8527–8537.

[37] Z. Zhang, A. Cutkosky, and I. Paschalidis, "Adversarial tracking control via strongly adaptive online learning with memory," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2022, pp. 8458–8492.

[38] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth, "How to use expert advice," *Journal of the ACM (JACM)*, vol. 44, no. 3, pp. 427–485, 1997.

[39] P. Zhao, Y.-H. Yan, Y.-X. Wang, and Z.-H. Zhou, "Non-stationary online learning with memory and non-stochastic control," *arXiv preprint arXiv:2102.03758*, 2021.

[40] N. K. Vishnoi, *Algorithms for convex optimization*. Cambridge University Press, 2021.