Meta-Learning Online Control for Linear Dynamical Systems

Deepan Muthirayan, Dileep Kalathil, and Pramod P. Khargonekar

***Abstract*—In this paper, we consider the problem of finding a meta-learning online control algorithm that can learn across the tasks when faced with a sequence of N (similar) control tasks. Each task involves controlling a linear dynamical system for a finite horizon of T time steps. The cost function and system noise at each time step are adversarial and unknown to the controller before taking the control action. Meta-learning is a broad approach where the goal is to prescribe an online policy for any new unseen task exploiting the information from other tasks and the similarity between the tasks. We propose a meta-learning online control algorithm for the control setting and characterize its performance by *meta-regret*, the average cumulative regret across the tasks. We show that when the number of tasks are sufficiently large, our proposed approach achieves a meta-regret that is smaller by a factor D/D^* compared to an independent-learning online control algorithm which does not perform learning across the tasks, where D is a problem constant and D^* is a scalar that decreases with increase in the similarity between tasks. Thus, when the sequence of tasks are similar the regret of the proposed meta-learning online control is significantly lower than that of the naive approaches without meta-learning. We also present experiment results to demonstrate the superior performance achieved by our meta-learning algorithm.**

I. Introduction

Meta-learning is a powerful paradigm in machine learning for *learning-to-learn* new tasks efficiently, e. g., with limited data [1]. Meta-learning is based on the intuitive idea that if the new task is similar to previous tasks, it can be learned very quickly by using the data and knowledge from previously encountered related tasks. Recently there has been tremendous progress in practical algorithms for meta-learning [2]–[4] with impressive performance in many applications such as image classification [5], natural language processing [6], and robotic control [7]. These algorithms, however, are in the *batch learning* setting, where data sets composed of different tasks are available for offline training. A meta-model (typically a neural network) is then trained using these data sets with the objective of fast adaptation to a new/unseen task at the test time using only a few data samples corresponding to that new task. Significantly different from the batch learning setting which are offline by nature, many learning algorithms have to operate in an *online* setting where the data samples are obtained in a sequential manner. For example, personalized recommendation systems [8], various applications in robotics [9]–[13], demand response management in smart grid [14], and load balancing in data centers [15] require online learning.

Online convex optimization (OCO) [16], [17] focuses on developing algorithms for online learning setting where the loss functions are sequentially revealed and the learner is trained as well as tested at each round. The standard OCO objective is to minimize the regret which is defined as the difference between the cumulative cost incurred by the online algorithm and the optimal policy from a certain class of policies. Even though the OCO approach offers a fundamental theoretical framework to analyze a variety of online learning scenarios, the existing works do not consider how the past experience can be used to accelerate adaptation to a new task, which is the key idea behind meta-learning. There are many works in the area of online control algorithms for dynamical systems with uncertain/unknown disturbances, system parameters and cost functions. The online control literature extends the OCO approach to problems with dynamics [18]–[21]. However, these existing works only consider the problem of learning *within* a task assuming that the task is fixed. In particular, they do not consider the possibility of *learning across the tasks* when faced with a sequence of similar control tasks.

In this paper, we consider the problem of finding a meta-learning online control algorithm that learns across the tasks when faced with a sequence of N (similar) control tasks. Each task involves controlling a linear dynamical system for T time steps. The cost function and system noise at each time step are adversarial and unknown to the algorithm before taking the control action. The primary role of a meta-learning algorithm is to prescribe an online control policy for any new unseen task exploiting the information from prior tasks and the similarity between the tasks. We characterize the performance of a meta-learning online control algorithm by *meta-regret*, the average (taken over the tasks) cumulative regret across the tasks. Our goal is to develop a meta-learning online control algorithm that can achieve superior performance, in theory and practice, over an independent-learning online control algorithm which applies a standard online control algorithm to each task without performing any learning across the tasks.

Our approach is motivated by some recent works in online meta-learning [22]–[24] which combine the meta-learning idea with the OCO framework. In [22], the authors extend the model-agnostic meta-learning (MAML) approach [2] to the online setting. Their goal is to learn a good meta-policy parameter that allows fast adaptation to all the previously seen tasks by taking only a few gradient steps from this meta-policy parameter. The work that is closest ours is [23], which proposes the Follow-the-Meta-Regularized-Leader

This work is supported in part by the National Science Foundation under Grant ECCS-1839429 and NSF- CAREER-EPCN-2045783. D. Muthirayan and P. P. Khargonekar are with the Department of Electrical Engineering and Computer Sciences, University of California Irvine, Irvine, CA (emails: deepan.m@uci.edu, pramod.khargonekar@uci.edu). Dileep Kalathil is with the Department of Electrical and Computer Engineering, Texas A&M University (email:dileep.kalathil@tamu.edu).

(FTMRL) approach. FTMRL learns a meta-intialization for a task specific OCO algorithm such that the individual task regret of these algorithms improves with the similarity of the online tasks. However, these works consider only the online optimization setting without state evolution. In particular, they do not consider the more challenging problem of online control for uncertain dynamical systems.

Our contributions: We consider the problem of developing a meta-learning online control algorithm for a sequence of similar control tasks. Each task involves controlling a linear dynamical system with adversarial cost functions and disturbances, which are unknown before taking the control action. Our algorithm has a two loop structure where the outer loop performs the meta-learning update to prescribe an initialization parameter for the task specific online control algorithm used in the inner loop. We show that when the number of tasks are sufficiently large the meta-regret of our proposed approach is smaller by a factor D/D^* compared to an independent-learning online control algorithm which does not perform learning across the tasks, where D is a problem constant and D^* is a scalar that represents the task similarity (D^* decreases with similarity between tasks). Therefore, when the sequence of tasks are similar, i.e., when $D^* \ll D$, we achieve a regret that is significantly lower than that of the naive approaches without meta-learning. We also present experiments results to demonstrate the superior performance of our meta-learning algorithm.

Our technical contribution lies in expanding the framework and technical analysis of online control to incorporate meta-learning. To the best of our knowledge, ours is the first work that combines the ideas of meta-learning and online control to develop a learning algorithm with provable guarantees for its performance. The conference version of this paper presents a simpler algorithm that assumes the knowledge of D^*. In this version, we introduce a general algorithm that does not require the knowledge of D^*.

Related Works:

Online Control: Substantial number of works have been published in the area of online control [18]–[21], [25]–[27]. Most of these works focus on developing online control algorithms for linear dynamical systems with provable guarantees for the regret. In our work we make use of the task specific online control algorithm proposed in [20]. This considers the control of a known linear dynamic system with adversarial disturbance and (convex) cost functions and shows that the proposed algorithm can achieve $\mathcal{O}(\sqrt{T})$ regret for a given task. Our meta-learning online control algorithm is developed by extending the task specific online control algorithm proposed in [20] with an additional outer loop for performing the meta-learning update and slightly modifying the task specific (inner loop) update.

Adaptive and Robust Control: Classical adaptive and robust control literature addresses the problem of control of systems with parametric, structural, modeling and disturbance uncertainties [28]–[31]. Typically, these classical approaches are concerned with stability and asymptotic performance guarantees of the systems. Online control literature focuses typically on the finite time regret performance of the algorithms. This is one of the key differences compared to the conventional adaptive and robust control literature, and it requires combining techniques from statistical learning, online optimization and control. In this work, we focus on the online control approach for developing our meta-learning algorithm.

Notations: Unless otherwise specified $\|\cdot\|$ denotes the Euclidean norm and the Frobenious norm for vectors and matrices respectively. We use $\mathcal{O}(\cdot)$ for the standard big-O notation while $\widetilde{\mathcal{O}}(\cdot)$ denotes the big-O notation neglecting the poly-log terms. We also use $o(\cdot)$ for the standard little-o notation. Further, when a function $g(n) = o_n(1)$, then $g(n) \to 0$ as $n \to \infty$. We denote the sequence $(x_{m_1}, x_{m_1+1}, \ldots, x_{m_2})$ compactly by $x_{m_1:m_2}$.

II. Problem Setting

We consider the problem of finding a meta-learning online control (M-OC) algorithm that learns across the tasks when faced with a sequence of (similar) control tasks. The sequence of tasks are denoted as $\tau_1, \tau_2, \ldots, \tau_N$. Each control task τ_i involves controlling a linear dynamical system for T time steps whose system dynamics is given by the equation

$$x_{i,t+1} = A_i x_{i,t} + B_i u_{i,t} + w_{i,t},\ 1 \le t \le T, \tag{1}$$

where $A_i \in \mathbb{R}^{n\times n}$ and $B_i \in \mathbb{R}^{n\times m}$ are the matrices that paramaterize the system, and $x_{i,t} \in \mathbb{R}^n$ is the state, $u_{i,t} \in \mathbb{R}^m$ is the action, $w_{i,t} \in \mathbb{R}^n$ is the system noise at time t. For conciseness we represent the system parameter for task τ_i as $\theta_i = [A_i, B_i]$. We assume that the systems noise is adversarial.

A control policy π for task τ_i selects a control action $u^\pi_{i,t}$ at each time t depending on the available information, resulting in a sequence of actions $u^\pi_{i,1:T}$ and the state trajectory $x^\pi_{i,1:T}$. The cumulative cost of a policy π under the system dynamics (1) is given by

$$J_i(\pi) = \sum_{t=1}^{T} c_{i,t}(x^\pi_{i,t}, u^\pi_{i,t}), \tag{2}$$

where $c_{i,t}$ is the cost function for task τ_i at time t. We assume that $c_{i,t}$s are arbitrary convex functions. The typical goal is to find the optimal policy $\pi^\star_i$ such that $\pi^\star_i = \arg\min_\pi J_i(\pi)$. Clearly, computing $\pi^\star_i$ requires the knowledge of the system parameter θ_i and the entire sequence of cost functions $c_{i,1:T}$.

The online control framework considers the more realistic setting where the future cost functions are not available for deciding the control action $u_{i,t}$ at time t. More precisely the policy π_i for task τ_i has only the following information at each time t for selecting the action $u_{i,t}$: (i) past and current state observations $x_{i,1:t}$, (ii) past control actions $u_{i,1:t-1}$, (iii) past cost functions $c_{i,1:t-1}$. We also assume that the system parameter θ_i is known to the control policy. The **task regret** of the control policy π_i for the task τ_i is defined as

$$R^i_T(\pi_i) = J_i(\pi_i) - \min_{\pi\in\Pi} J_i(\pi), \tag{3}$$

where Π is the class of control policies. The objective is to find a policy that minimizes the task regret assuming that the task is fixed. In particular the existing online control algorithms

do not consider *learning across tasks* when faced with a sequences of similar control tasks.

Our goal is to find a meta-policy π^{m} that can learn across the tasks when faced with a sequence of (similar) control tasks $\tau_1, \tau_2, \ldots, \tau_N$ and minimize the task regret for individual tasks. A meta-policy π^{m} produces a sequence of task specific policies $\pi_i^{\mathrm{m}}, 1 \leq i \leq N$, by learning across the tasks. For deciding the task specific policy π_i^{m} for task τ_i the meta-policy π^{m} makes use of the observation available from the previous tasks: the state observations, cost functions, and task specific policies for all previous tasks $j \leq i-1$. Since the objective of the meta-policy is to generate task specific policies which can do well on individual tasks, the performance of the meta-policy is characterized by the metric **meta-regret**, formally defined as

$$R_N^{\mathrm{meta}}(\pi^{\mathrm{m}}) = \frac{1}{N}\sum_{i=1}^{N} R_T^i(\pi_i^{\mathrm{m}}). \tag{4}$$

Our objective is to find a meta-policy that performs better than an *independent-learning online control algorithm* which applies a standard online control algorithm independently to each task without performing any learning across the tasks.

We make the following assumptions. Please note that the assumptions stated below are standard in the (task specific) online control literature [20] and no further assumptions are made.

Assumption 1 (System Model): (i) The system matrices for each task are bounded, $\|A_i\| \leq \kappa_A$, and $\|B_i\| \leq \kappa_B$, where κ_A and κ_B are constants. (ii) The disturbance at time t of any task is bounded, $\|w_{i,t}\| \leq \kappa_w$, where κ_w is a constant.

Assumption 2 (Cost Functions): For all tasks $i, 1 \leq i \leq N$ and all time steps $t, 1 \leq t \leq T$, (i) the costs functions $c_{i,t}$s are convex, (ii) for any x and u with $\|x\| \leq S, \|u\| \leq S$,

$$\|c_{i,t}(x,u)\| \leq \beta S^2, \|\nabla_x c_{i,t}(x,u)\|, \|\nabla_u c_{i,t}(x,u)\| \leq GS,$$

The above formulation can be extended to the problem of controlling an output instead of a state just as in [21]. In this setting ([21]), the cost function is a function of the output and the control input, instead of the state and the control input.

III. Review: Online Control Algorithm

In this section we give a brief description of the task specific online control (OC) algorithm proposed in [20]. We drop the task subscript i because the discussion here is for a single task. Our meta-learning online control algorithm is developed by extending the task specific OC algorithm with an additional outer loop for performing the meta-learning update and appropriately modifying the task specific (inner loop) update.

The OC algorithm proposed in [20] uses a control policy parameterized by two matrices, a fixed matrix K and a time varying matrix $M_t = (M_t^{[1]}, M_t^{[2]}, \ldots, M_t^{[H]})$. The control action u_t at time t by this OC algorithm is given by

$$u_t = -Kx_t + \sum_{k=1}^{H} M_t^{[k]} w_{t-k}. \tag{5}$$

Thus, the control action is a linear map of the current state and the past disturbances up to a certain history. This property is convenient as it permits efficient optimization of the costs. We note that, since the state is fully observable, the past disturbances can be precisely estimated using the information at time t.

The parameter K is selected by the OC algorithm as a (κ,γ)-strongly stable linear feedback control matrix for the underlying system. A linear feedback control policy specified by the gain K is (κ,γ)-strongly stable if there exists matrices L, H satisfying $A - BK = HLH^{-1}$ such that the following two conditions are met: (i) $\|L\| \leq 1-\gamma$, and (ii) $\|K\| \leq \kappa, \|H\|, \|H^{-1}\| \leq \kappa$. The OC algorithm considers the class Π of all (κ,γ)-strongly stable linear feedback controllers for characterizing its regret performance according to (3).

The OC algorithm uses the framework of Online Convex Optimization (OCO) to update the parameters M_t at each time step. The key idea of the algorithm is to design a sequence of cost functions $f_{1:T}$ in terms of the parameters $M_{1:T}$ while correctly representing the actual cost incurred by the true cost functions $c_{1:T}$. This is achieved by defining an idealized state s_t and idealized control input a_t as follows. The idealized state s_t is the state the system would have reached if the controller had executed the policy with parameters $(M_{t-H}, \ldots, M_{t-1})$ from time step $t-H$ to time step $t-1$, assuming that the state at $t-H$ is 0. The idealized action a_t is the action that would have been executed at time t if the state observed at time t is s_t. We can then define the idealized cost as $f_t(M_{t-H}, \ldots, M_t) = c_t(s_t, a_t)$.

The complete OC algorithm proposed in [20] is shown in Algorithm 1. An Online Gradient Descent (OGD) approach updates the parameters M_t by the gradient of the idealized cost function. The algorithm requires the specification of a (κ,γ)-strongly stable matrix K. Such a matrix can be calculated offline before the task using an Semi-Definite Programming (SDP) relaxation as described in [32].

Algorithm 1 Online Control (OC) Algorithm

Input: Step size η, parameters $\kappa_B, \kappa, \gamma, T$, (κ,γ)-strongly stable control matrix K
Define $H = \log T/(\log(1/1-\gamma))$
Define $\mathcal{M} = \{M = (M^{[1]}, \ldots, M^{[H]}) : \|M^{[k]}\| \leq \kappa^3 \kappa_B (1-\gamma)^k\}$
Define $g_t(M) = f_t(M, \ldots, M)$
Initialize $M_1 \in \mathcal{M}$
for *t = 1,...,T* **do**
  Choose the action $u_t = -Kx_t + \sum_{k=1}^{H} M_t^{[k]} w_{t-k}$
  Observe the new state x_{t+1}, and $w_t = x_{t+1} - Ax_t - Bu_t$
  Update $M_{t+1} = \mathrm{Proj}_{\mathcal{M}}(M_t - \eta \nabla g_t(M_t))$
end

A regret guarantee of Algorithm 1 is provided in [20]:

Theorem 1 (Theorem 5.1, [20]): Suppose Assumptions 1-2 hold, $\eta = \frac{D}{\sqrt{G_f(G_f/2+LH^2)T}}$, and $D = \frac{\kappa_B \kappa^3 \sqrt{d}}{\gamma}$. Then, under Algorithm 1,

$$R_T \leq \frac{3D\sqrt{G_f(G_f/2+LH^2)T}}{2} + \widetilde{\mathcal{O}}(1), \quad \text{where}$$

$$L = 2G\widetilde{D}\kappa_w\kappa_B\kappa^3, \quad G_f = G\widetilde{D}\kappa_w H d\left(\frac{2\kappa_B\kappa^3}{\gamma} + H\right),$$

$$\widetilde{D} = \frac{\kappa_w(\kappa^2 + H\kappa_B^2\kappa^5)}{\gamma(1-\kappa^2(1-\gamma)^{H+1})} + \frac{\kappa_B\kappa^3\kappa_w}{\gamma}.$$

The notation Π is used as a general notiation for a benchmark class of policies. The set $\mathcal{M}$ (used above) is a specific class of control policies as defined in Algorithm 1. In the regret presented in the paper $\Pi = \mathcal{M}$. We have made this clarification in the revised version.

Remark 1 (Diameter of the domain): It can be shown that [20, Theorem 5.1] the multiplicative constant D in the above regret bound is the diameter of the domain $\mathcal{M}$ of the control policy parameters, i.e., $D = \max_{M_1,M_2\in\mathcal{M}}\|M_1 - M_2\|$. In the next section we show that our meta-learning approach can significantly reduce this multiplicative constant by learning across the tasks.

IV. Meta-learning Online Control Algorithm

Our meta-learning online control (M-OC) algorithm builds on the simple, yet a powerful idea of meta-initialization. In the standard OC algorithms, the initialization parameter for the control policy is selected arbitrarily from the domain of possible parameters. So, inevitably the regret guarantee for such algorithms includes a multiplicative constant that is of the order of the radius of the domain (see Remark 1), which can be very large in many problems. Similarly, when an independent-learning OC algorithm is applied to a sequence of tasks the parameters of the control policy for each task are initialized arbitrarily ignoring the similarities and the benefit of learning across tasks. When the tasks are similar, the optimal parameters for the individual tasks are closer to each other, and the optimal parameters for the earlier tasks in the sequence can be used to improve the learning in a new upcoming task. Our M-OC algorithm translates this intuitive idea into providing a clever initialization for the control policy for the current task by learning from the previous tasks. This results in a multiplicative constant (in the regret) that is proportional to the diameter D^* of a much smaller subset that contains the parameters of the optimal control policies of the individual tasks, instead of the diameter of the generic domain. This scenario is illustrated in Fig. 1, where the diameter D of the original domain $\mathcal{M}$ is significantly larger than D^*, which is the diameter of the smaller set $\mathcal{M}^*$ that contains the optimal parameters corresponding to the similar tasks. Here the diameter D^* can be interpreted as the similarity of the sequence of tasks.

The architecture of our M-OC algorithm is given in Fig. 2. The meta-learning in the outer loop provides the meta-initialization for the task specific OC algorithm in the inner loop. The control policy for each specific task is of the same form as the independent learning OC algorithm (5). At the beginning of any task τ_i a (κ,γ) stabilizing feedback gain matrix K_i for the task τ_i is computed. During the task the algorithm updates the task specific policy parameters $M_{i,t}$ exactly as in Algorithm 1. The control action $u_{i,t}$ is computed using the parameters $M_{i,t}$ and the feedback gain matrix K_i with the same form as the independent learning OC algorithm (5). The difference between the M-OC algorithm and Algorithm 1 lies in the initialization of the parameter $M_{i,1}$.

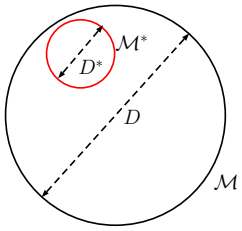

Fig. 1. Illustrative figure showing the domain $\mathcal{M}$ of the parameters of the online control policies and the set $\mathcal{M}^*$ of the optimal parameters of the control policies corresponding to a set of *similar* tasks.

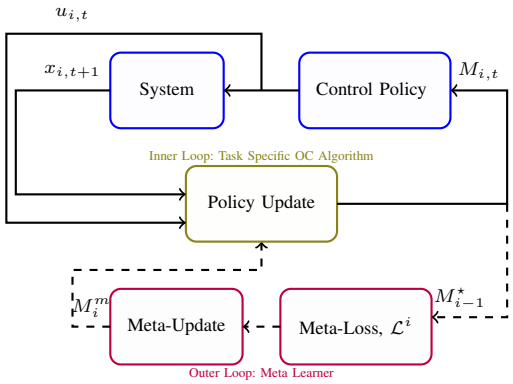

Fig. 2. Meta-Learning Online Control (M-OC) Algorithm Architecture. Solid line: within task signals. Dashed line: signals that are constant within a task but that can change across the tasks.

In particular, Algorithm 1 selects $M_{i,1}$ arbitrarily from the domain $\mathcal{M}$, whereas the outer loop of meta-learner provides the initialization M_i^{m} for each task τ_i.

Specifically, the inner loop updates the control policy parameter $M_{i,t}$ within each task τ_i by

$$M_{i,t+1} = \mathrm{Proj}_{\mathcal{M}}\left(M_{i,t} - \nabla g_{i,t}(M_{i,t})\right), \; M_{i,1} = M_i^{\mathrm{m}}. \quad (6)$$

In the outer loop, the meta-learner computes the initialization parameter M_i^{m} for the inner loop as follows. Let $M_i^\star$ the optimal parameter in hindsight for task τ_i, i.e.,

$$M_i^\star = \arg\min_{M\in\mathcal{M}} \sum_{t=1}^{T} g_{i,t}(M). \quad (7)$$

We note that $M_i^\star$ is computable at the end of task τ_i. Given that $g_{i,t}$s are convex functions, finding $M_i^\star$ is a convex optimization problem, and thus can be solved efficiently. We define the meta-learner's loss for task i as

$$\mathcal{L}^i(M^{\mathrm{m}}) = \frac{1}{2}\|M^{\mathrm{m}} - M_i^\star\|^2. \quad (8)$$

The meta-learner performs an online gradient descent step to find the initialization M_{i+1}^{m} for task τ_{i+1} as

$$M_{i+1}^{\mathrm{m}} = \mathrm{Proj}_{\mathcal{M}}\left(M_i^{\mathrm{m}} - \frac{1}{i}\nabla\mathcal{L}^i(M_i^{\mathrm{m}})\right). \quad (9)$$

We note that performing the naive initialization $M_{i+1}^{\mathrm{m}} = M_i^\star$ does not improve the regret optimally as this will effectively throw away the information from all the previous tasks. Instead the meta-learner solves an online convex optimization problem with N steps with the cost function at each step i given by $\mathcal{L}^i$. Since the online gradient descent approach solves

this problem efficiently with provable guarantees for the regret performance, we adapt this approach as the meta-learning algorithm in the outer loop. We present two variations of the algorithm: (i) a simpler algorithm which assumes knowledge of the diameter D^*, and (ii) a complete algorithm that does not require the knowledge of D^*.

A. Algorithm with the Knowledge of D^*

We first present the algorithm with the knowledge of D^* for easier understanding of the idea and the technical analysis. The key advantage of this assumption is that we can set the learning rate η in the inner loop proportional to D^* in addition to updating the meta-initialization according to (9). We emphasize that setting $\eta \propto D^*$ is the optimal way to set the rate, which follows from how η is set in Theorem 1 for the independent learning OC algorithm. The assumption of knowledge of D^* simplifies the algorithm, which otherwise requires setting the learning rate adaptively. We present the more general algorithm in the next section. The algorithm with the knowledge of D^* is presented below.

Algorithm 2 Meta-learning Online Control (M-OC-1) Algorithm

Input: Number of tasks N, the diameter D^*, inner loop step size η, parameters $\kappa_B, \kappa, \gamma, T$

Define $\mathcal{M} = \{M = (M^{[1]}, \dots, M^{[H]}) : \|M^{[k]}\| \leq \kappa^3 \kappa_B (1-\gamma)^k\}$. Initialize $M_1^{\mathrm{m}} \in \mathcal{M}$ arbitrarily

for *i = 1,…,N* **do**

- For task τ_i, set the initialization $M_{i,1} = M_i^{\mathrm{m}}$ for the OC Algorithm (Algorithm 1) in the inner loop
- Execute the OC Algorithm (Algorithm 1) for task τ_i
- Compute $M_i^\star$ as in (7)
- Update M_{i+1}^{m} as in (8)-(9)

end

We now present our main result which characterizes the performance of Algorithm 2.

Theorem 2: Suppose Assumptions 1-2 hold, and $\eta = \frac{D^*}{\sqrt{G_f(G_f/2+LH^2)T}}$. Then, under the M-OC-1 Algorithm (Algorithm 2)

$$R_N^{\mathrm{meta}} \leq \left(\frac{(GD)^2 \log N}{D^* N} + \frac{\overline{D}}{2} + D^* \right) \sqrt{\widetilde{G}^2 T},$$

where, $\overline{D}^2 = \frac{1}{N}\sum_{i=1}^N \left(M_i^\star - \widetilde{M}^\star\right)^2$, $\widetilde{M}^\star = \frac{1}{N}\sum_{i=1}^N M_i^\star$, $\widetilde{G}^2 = G_f\left(\frac{G_f}{2} + LH^2\right)$.

Remark 2 (Comparison with independent-learning online control algorithm): Under our M-OC-1 algorithm, when N is sufficiently large, the multiplicative constant in the regret upper bound is approximately equal to $\frac{\overline{D}}{2}+D^*$. When the tasks are similar $D^* \ll D$, and by definition $\overline{D} \leq D^*$. Therefore, when the tasks are similar the regret our algorithm achieves is significantly better compared to the independent learning OC algorithm. This clearly shows that M-OC-1 is able to learn across tasks, which by default the independent learning OC algorithm cannot do. This fact is verified by our numerical simulations also; see Section V.

Remark 3 (Achievability by meta-learning): We note that the meta-regret scaling with respect to the duration T of a control task is $\widetilde{\mathcal{O}}(\sqrt{T})$, which is same as the scaling achieved by the independent learning OC algorithm. This aspect is consistent with the existing theoretical results in online meta-learning [22]–[24]. This is expected, as the meta-learner will never be able to learn an initialization that does not require further adaptation, especially, since the cost functions and the disturbances are arbitrary. Furthermore, as pointed in [23, Theorem 2.2], even in the simpler OCO setting, reductions to the multiplicative constant are the best that can be achieved.

Remark 4 (Knowledge of D^* vs $\mathcal{M}^*$): We emphasize that our algorithm only assumes the knowledge of a scalar D^*, and not of the entire multi-dimensional set $\mathcal{M}^*$. Assuming the knowledge of $\mathcal{M}^*$ is not realistic in most practical problems.

Remark 5 (Extension to Output Feedback): In the output feedback setting of [21], the cost function is a function of the output and the control input. The output is an observable and also the variable to be controlled. This leads to the following difference in the policy parameter update. The cost f_t, in this case, is a function of the idealized output and the idealized action, instead of the idealized state and action. This is the only difference. The rest of the update equations are the same.

B. Algorithm without the Knowledge of D^*

In this subsection, we present a general version of our algorithm which does not assume the knowledge of D^*. As mentioned earlier, without the knowledge of D^*, requires setting the learning rate adaptively.

Our approach is motivated by the idea proposed in [24], but we present a simpler algorithm which lends itself to a simpler proof. We set the learning rate for task τ_i as $\eta = \frac{D_i}{\sqrt{G_f(G_f/2+LH^2)T}}$, where D_i is an estimate of the diameter of the smallest bounding circle of the region $\mathcal{M}^*$. We update D_i whenever there is evidence that D_i is smaller that D^*. The idea is to start D_i from a guess (a small number ϵ) of D^* and increase this guess by a factor $\zeta > 1$ whenever $\|M_i^\star - \widetilde{M}_{i-1}^{\mathrm{m}}\| > D_i$, where $\widetilde{M}_i^{\mathrm{m}} = \frac{1}{i}\sum_{j=1}^i M_i^\star$. The term $\|M_i^\star - \widetilde{M}_{i-1}^{\mathrm{m}}\|$ is the deviation of the optimal parameter for a new task i from the average of the optimal parameters of the previous tasks. Thus, this term is indicative of how smaller D_i is, and thus can be used to increase D_i by comparing with it. In addition, since $\widetilde{M}_{i-1}^{\mathrm{m}}$ is equal to the output of the meta-learner in Eq. (9) with M_1^{m} set to zero, we use $\widetilde{M}_{i-1}^{\mathrm{m}}$ itself as the meta-initialization for the task τ_i. The complete algorithm is shown in Algorithm 3.

We now present our main result which characterizes the performance of Algorithm 3.

Theorem 3: Suppose Assumptions 1-2 hold, $\epsilon < D^*$, and $\zeta = (1+\log(T))/\log(T)$. Then, under the M-OC-2 Algorithm (Algorithm 3)

$$R_N^{\mathrm{meta}} \leq \left(\frac{(GD)^2 \log N}{D^* N} + \frac{4D^3}{\epsilon^2 N} + \frac{\overline{D}}{2} + D^* + o_T(1) \right) \sqrt{\widetilde{G}^2 T}$$

where, $\overline{D}^2 = \frac{1}{N}\sum_{i=1}^N \left(M_i^\star - \widetilde{M}^\star\right)^2$, $\widetilde{M}^\star = \frac{1}{N}\sum_{i=1}^N M_i^\star$, $\widetilde{G}^2 = G_f\left(\frac{G_f}{2} + LH^2\right)$

Algorithm 3 Meta-learning Online Control (M-OC-2) Algorithm

Input: Number of tasks N, parameters $\kappa_B, \kappa, \gamma, T, \epsilon, \zeta > 1$
Define $\mathcal{M} = \{M = (M^{[1]}, \ldots, M^{[H]}) : \|M^{[k]}\| \leq \kappa^3 \kappa_B (1-\gamma)^k\}$. Set M_1^{m} to the origin. Initialize $D_1 = \epsilon$, $k = 0$.
for i = *1,...,N* **do**
  Set $\eta = \frac{D_i}{\sqrt{G_f(G_f/2+LH^2)T}}$
  For task τ_i, set the initialization $M_{i,1} = M_i^{\mathrm{m}}$ for the OC Algorithm (Algorithm 1) in the inner loop
  Execute the OC Algorithm (Algorithm 1) for task τ_i
  Compute $M_i^\star$ as in (7)
  Set $M_{i+1}^{\mathrm{m}} = \frac{1}{i}\sum_{j=1}^{i} M_j^\star$
  if $i > 1$ **then**
    if $\|M_i^\star - M_i^{\mathrm{m}}\| > D_i$ **then**
      $k \leftarrow k+1$
    end
  end
  $D_{i+1} = \zeta^k \epsilon$
end

Remark 6 (Comparison with independent-learning online control algorithm and M-OC-1 algorithm): Under M-OC-2 algorithm, when N is sufficiently large, the multiplicative constant in the regret upper bound is approximately equal to $\frac{\overline{D}}{2} + D^*$. We recall from Remark 2 that $\overline{D} \leq D^*$ (by definition), and when the tasks are similar $D^* \ll D$. Therefore, when the tasks are similar, we observe that the regret M-OC-2 achieves is significantly better compared to the independent learning OC algorithm. We also observe that the M-OC-2 algorithm has an additional term $\frac{D^2}{\epsilon N}$ compared to the M-OC-1 algorithm. This indicates that when the initial guess ϵ is very small, the number of tasks N that M-OC-2 observes has to be sufficiently large. This is expected as, when ϵ is much smaller compared to D^* meta-learning will necessarily require more experience to improve the initial guess $D_i = \epsilon$.

Please refer to the arxiv version [33] for the proofs of Theorem 2 and Theorem 3.

V. Numerical Experiments

In this section, we present numerical experiments to demonstrate the benefits of our proposed meta-learning online control algorithm. We consider only the M-OC-1 algorithm for the simplicity of illustration. In our experiments, each task τ_i is the problem of regulating a linear dynamical system given in 1 with dimensions $n = 2, m = 1$. The system model A_i in each task τ_i is selected as a random matrix: a perturbation around a nominal matrix. In particular, we set $A_i = \frac{1}{2n} I + \frac{1}{5n} W_i$, where W_i is a random matrix with the value of each element generated uniformly from the interval $[0, 1]$. This structure implicitly incorporates the idea of task similarity. The cost functions $c_{i,t}$s are selected as quadratic cost functions $c_{i,t}(x,u) = x^\top Q_t x + u^\top R_t u$, where Q_t and R_t are randomly chosen diagonal matrices with each diagonal element chosen randomly from the range $[0.375, 0.625]$. The other parameters are selected as $\kappa_a = \kappa_b = \kappa_w = 1, \kappa = \sqrt{nm}, \gamma = 0.5$.

In our experiments, we compare the performance of our M-OC algorithm with the following benchmarks:

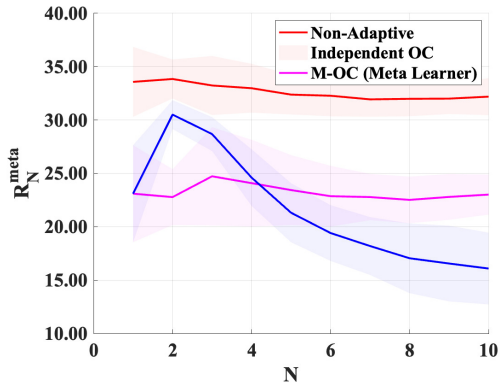

Fig. 3. Plot of R_N^{meta} versus the number for tasks N. The plot shows a patch of one standard deviation for each controller.

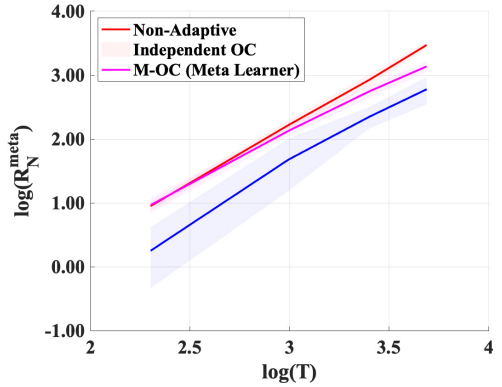

Fig. 4. Plot of $\log R_N^{\mathrm{meta}}$ vs $\log T$. The plot shows a patch of one standard deviation for each controller.

(i) *Non-adaptive control algorithm* which employs the control policy $u_{i,t} = -K_i x_t$, where K_i is a stabilizing controller for task τ_i with system parameter $\theta_i = [A_i, B_i]$. We select K_i by solving a standard linear matrix inequality (LMI) for finding a stabilizing controller. We call this non-adaptive control because the control policy is invariant over the duration of the control tasks. Moreover, there is no learning across the tasks.

(ii) *Independent-learning online control algorithm* employs the task specific OC algorithm (Algorithm 1) independently to each control task. While this approach is capable of learning within a task, it does not perform any meta-learning across the tasks.

Different from these benchmarks, our M-OC algorithm can learn within and across the tasks.

Figure 3 shows the meta-regret R_N^{meta} as a function of the number of tasks N with $T = 25$ for all tasks. Note that meta-regret is equivalent to the average (averaged over the tasks) cumulative regret of the tasks; see (4). Since the non-adaptive control algorithm and the independent-learning OC algorithm do not perform any learning across the tasks, their meta-regret does not improve with the number of tasks. In stark difference, the meta-regret of our M-OC algorithm decreases with the number of tasks; see Remark 2 also. This is because our M-OC algorithm is designed to perform meta-learning across

the tasks. This clearly demonstrates the superior performance of the M-OC algorithm over the benchmarks without meta-learning.

Figure 4 shows the variation of the meta-regret with $N = 15$ tasks as a function of the duration T of each control task. We see that, when the task duration is small, the M-OC outperforms independent learning OC by a notable margin. This indeed is the very purpose of meta-learning, i.e., to improve adaptation when the data or experience available for online learning is limited. The performance of the independent OC approaches that of M-OC as T becomes large. This is because, the initialization has an effect only on the shorter time scales and not on the longer time scales.

VI. Conclusion

In this paper, we address the problem of developing a meta-learning online control algorithm for a sequence of similar control tasks. We focus on the setting where each task is the problem of controlling a linear dynamical system with arbitrary disturbances and arbitrarily time varying cost functions. We propose a meta-learning online control algorithm that provably achieves a superior performance compared to the standard online control algorithm which does not use meta-learning. We also present numerical experiments to demonstrate the superior performance of our algorithm. In the future work, we plan to extend this approach to the setting where the system parameters θ_is are also unknown.

References

[1] S. Thrun and L. Pratt, *Learning to learn*. Springer Science & Business Media, 2012.

[2] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning (ICML)*, 2017.

[3] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *International Conference on Learning Representations (ICLR)*, 2017.

[4] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[5] X. Wang, T. Huang, J. Gonzalez, T. Darrell, and F. Yu, "Frustratingly simple few-shot object detection," in *International Conference on Machine Learning*, 2020.

[6] J. Bragg, A. Cohan, K. Lo, and I. Beltagy, "Flex: Unifying evaluation for few-shot nlp," in *Neural Information Processing Systems (NeurIPS)*, 2021.

[7] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn, "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2019.

[8] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *International conference on World Wide Web*, 2010, pp. 661–670.

[9] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez, "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," *Information fusion*, vol. 58, pp. 52–68, 2020.

[10] T. Petrič, A. Gams, L. Žlajpah, and A. Ude, "Online learning of task-specific dynamics for periodic tasks," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 1790–1795.

[11] F. Alambeigi, Z. Wang, R. Hegeman, Y.-H. Liu, and M. Armand, "A robust data-driven approach for online learning and manipulation of unmodeled 3-d heterogeneous compliant objects," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4140–4147, 2018.

[12] D. Romeres, M. Zorzi, R. Camoriano, S. Traversaro, and A. Chiuso, "Derivative-free online learning of inverse dynamics models," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 816–830, 2019.

[13] S. Tesfazgi, A. Lederer, J. F. Kunz, A. J. Ordóñez-Conejo, and S. Hirche, "Personalized rehabilitation robotics based on online learning control," *arXiv preprint arXiv:2110.00481*, 2021.

[14] D. Kalathil and R. Rajagopal, "Online learning for demand response," in *Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2015, pp. 218–222.

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

[16] S. Shalev-Shwartz *et al.*, "Online learning and online convex optimization," *Foundations and trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2011.

[17] E. Hazan, "Introduction to online convex optimization," *arXiv preprint arXiv:1909.05207*, 2019.

[18] S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu, "Regret bounds for robust adaptive control of the linear quadratic regulator," in *Neural Information Processing Systems (NeurIPS)*, 2018.

[19] H. Mania, S. Tu, and B. Recht, "Certainty equivalence is efficient for linear quadratic control," in *Neural Information Processing Systems (NeurIPS)*, 2019.

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

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

[22] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine, "Online meta-learning," in *International Conference on Machine Learning (ICML)*, 2019, pp. 1920–1930.

[23] M.-F. Balcan, M. Khodak, and A. Talwalkar, "Provable guarantees for gradient-based meta-learning," *International Conference on Machine Learning (ICML)*, pp. 424–433, 2019.

[24] M. Khodak, M.-F. F. Balcan, and A. S. Talwalkar, "Adaptive gradient-based meta-learning methods," *Neural Information Processing Systems (NeurIPS)*, 2019.

[25] A. Cohen, T. Koren, and Y. Mansour, "Learning linear-quadratic regulators efficiently with only $\sqrt{T}$ regret," *International Conference on Machine Learning (ICML)*, pp. 1300–1309, 2019.

[26] M. Simchowitz and D. Foster, "Naive exploration is optimal for online lqr," *International Conference on Machine Learning (ICML)*, pp. 8937–8948, 2020.

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

[28] S. Sastry and M. Bodson, *Adaptive control: stability, convergence and robustness*. Dover Publications, 2011.

[29] K. J. Åström and B. Wittenmark, *Adaptive control*. Courier Corporation, 2013.

[30] P. A. Ioannou and J. Sun, *Robust adaptive control*. Dover Publications, 2012.

[31] K. Zhou, J. Doyle, and K. Glover, *Robust and optimal control*. Prentice hall, 1996.

[32] A. Cohen, A. Hasidim, T. Koren, N. Lazic, Y. Mansour, and K. Talwar, "Online linear quadratic control," in *International Conference on Machine Learning (ICML)*, 2018, pp. 1029–1038.

[33] D. Muthirayan, D. Kalathil, and P. P. Khargonekar, "Meta-learning online control for linear dynamical systems," pp. 1435–1440, 2022.