Meta-Adaptive Nonlinear Control: Theory and Algorithms

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present an online multi-task learning approach for adaptive nonlinear control, which we call Online Meta-Adaptive Control (OMAC). The goal is to control a nonlinear system subject to adversarial disturbance and unknown *environment-dependent* nonlinear dynamics, under the assumption that the environment-dependent dynamics can be well captured with some shared representation. Our approach is motivated by robot control, where a robotic system encounters a sequence of new environmental conditions that it must quickly adapt to. A key emphasis is to integrate online representation learning with established methods from control theory, in order to arrive at a unified framework that yields both control-theoretic and learning-theoretic guarantees. We provide instantiations of our approach under varying conditions, leading to the first non-asymptotic convergence guarantee for multi-task adaptive nonlinear control as well as integration with deep learning. Experiments show that OMAC significantly outperforms conventional adaptive control approaches which do not learn the shared representation.

1 Introduction

One important goal in autonomy and artificial intelligence is to enable autonomous robots to learn from prior experience to quickly adapt to new tasks and environments. Examples abound in robotics, such as a drone flying in different wind conditions [1], a manipulator throwing varying objects [2], or a quadruped walking over changing terrains [3]. Though those examples provide encouraging empirical evidence, when designing such adaptive systems, two important theoretical challenges arise, as discussed below.

First, from a learning perspective, the system should be able to learn an "efficient" representation from prior tasks, thereby permitting faster future adaptation, which falls into the categories of representation learning or meta-learning. Recently, a line of work has shown theoretically that learning representations (in the standard supervised setting) can significantly reduce sample complexity on new tasks [4, 5]. Empirically, deep representation learning has achieved success in many applications [6], including control, in the context of meta-reinforcement learning [7–9]. However, theoretical benefits of representation learning or meta-learning for adaptive control remain unclear.

Second, from a control perspective, the agent should be able to handle parametric model uncertainties with control-theoretic guarantees such as stability and tracking error convergence, which is a common adaptive control problem [10]. For classic adaptive control algorithms, theoretical analysis often involves the use of Lyapunov stability and asymptotic convergence [10, 11]. Moreover, many recent studies aim to integrate ideas from learning, optimization and control theory to design and analyze adaptive controllers using learning-theoretic metrics. Typical results guarantee non-asymptotic convergence in finite time horizons, such as regret [12–15] and dynamic regret [16]. However, these

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

results focus on a single environment or task. A multi-task extension, especially whether and how prior experience could benefit the adaptation in new tasks, remains an open problem.

Main contributions. In this paper, we address both learning and control challenges in a unified framework. We derive a new method of Online Meta-Adaptive Control (OMAC) that controls uncertain nonlinear systems under a sequence of new environmental conditions. The underlying assumption is that the environment-dependent unknown dynamics can well be captured by a shared representation, which OMAC learns using a *meta-adapter*. OMAC then performs environment-specific updates using an *inner-adapter*.

We provide different instantiations of OMAC under varying assumptions and conditions. In the jointly and element-wise convex cases, we show sublinear cumulative control error bounds, which to our knowledge is the first non-asymptotic convergence result for multi-task nonlinear control. Compared to standard adaptive control approaches that do not have a meta-adapter, we show that OMAC possesses both stronger guarantees and empirical performance. We finally show how to integrate OMAC with deep representation learning, which further improves empirical performance.

2 Problem statement

We consider the setting where a controller encounters a sequence of N environments, with each environment lasting T time steps. We use *outer iteration* to refer to the iterating over the N environments, and *inner iteration* to refer to the T time steps within an environment.

Notations: We use superscripts (e.g., (i) in $x_t^{(i)}$) to denote the index of the outer iteration where $1 \leq i \leq N$, and subscripts (e.g., t in $x_t^{(i)}$) to denote the time index of the inner iteration where $1 \leq t \leq T$. We use $\texttt{step}(i, t)$ to refer to the inner time step t at the ith outer iteration. $\|\cdot\|$ denotes the 2-norm of a vector or the spectral norm of a matrix. $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. $\text{vec}(\cdot) \in \mathbb{R}^{mn}$ denotes the vectorization of a $m \times n$ matrix, and $\otimes$ denotes the Kronecker product. Finally, we use $u_{1:t}$ to denote a sequence $\{u_1, u_2, \cdots, u_t\}$. All proofs can be found in Appendix A.

We consider a discrete-time nonlinear control-affine system [17, 18] with environment-dependent uncertainty $f(x, c)$. The dynamics model at the ith outer iteration is:

$$x_{t+1}^{(i)} = f_0(x_t^{(i)}) + B(x_t^{(i)})u_t^{(i)} - f(x_t^{(i)}, c^{(i)}) + w_t^{(i)}, \quad 1 \leq t \leq T-1, \tag{1}$$

where the state $x_t^{(i)} \in \mathbb{R}^n$, the control $u_t^{(i)} \in \mathbb{R}^m$, $f_0 : \mathbb{R}^n \to \mathbb{R}^n$ is a known nominal dynamics model, $B : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ is a known state-dependent actuation matrix, $c^{(i)} \subset \mathbb{R}^h$ is the unknown parameter that indicates an environmental condition, $f : \mathbb{R}^n \times \mathbb{R}^h \to \mathbb{R}^n$ is the unknown $c^{(i)}$-dependent dynamics model, and $w_t^{(i)}$ is disturbance, potentially adversarial. For simplicity we define $B_t^{(i)} = B(x_t^{(i)})$ and $f_t^{(i)} = f(x_t^{(i)}, c^{(i)})$.

Interaction protocol. We study the following adaptive nonlinear control problem under N unknown environments. At the beginning of outer iteration i, the environment first selects $c^{(i)}$ (adaptively and adversarially), which is unknown to the controller, and then the controller makes decision $u_{1:T-1}^{(i)}$ under unknown dynamics $f(x_t^{(i)}, c^{(i)})$ and potentially adversarial disturbances $w_t^{(i)}$. To summarize:

1. **Outer iteration i.** A policy encounters environment i ($i \in \{1, \ldots, N\}$), associated with unobserved variable $c^{(i)}$ (e.g., the wind condition for a flying drone). Run inner loop (Step 2).
2. **Inner loop.** Policy interacts with environment i for T time steps, observing $x_t^{(i)}$ and taking action $u_t^{(i)}$, with state/action dynamics following (1).
3. Policy optionally observes $c^{(i)}$ at the end of the inner loop (used for some variants of the analysis).
4. Increment $i = i + 1$ and repeat from Step 1.

We use average control error (ACE) as our performance metric:

Definition 1 (Average control error)**.** The average control error (ACE) of N outer iterations (i.e., N environments) with each lasting T time steps, is defined as $\mathsf{ACE} = \frac{1}{TN}\sum_{i=1}^{N}\sum_{t=1}^{T}\|x_t^{(i)}\|$.

ACE can be viewed as a non-asymptotic generalization of steady-state error in control [19]. We make the following assumptions on the actuation matrix B, the nominal dynamics f_0, and disturbance $w_t^{(i)}$:

Assumption 1 (Full actuation, bounded disturbance, and e-ISS assumptions)**.** *We consider fully-actuated systems, i.e., for all x, $\mathrm{rank}(B(x)) = n$. $\|w_t^{(i)}\| \leq W, \forall t, i$. Moreover, the nominal dynamics f_0 is exponentially input-to-state stable (e-ISS): let constants $\beta, \gamma \geq 0$ and $0 \leq \rho < 1$. for a sequence $v_{1:t-1} \in \mathbb{R}^n$, consider the dynamics $x_{k+1} = f_0(x_k) + v_k, 1 \leq k \leq t-1$. x_t satisfies:*

$$\|x_t\| \leq \beta \rho^{t-1} \|x_1\| + \gamma \sum_{k=1}^{t-1} \rho^{t-1-k} \|v_k\|. \tag{2}$$

With the e-ISS property in Assumption 1, we have the following bound that connects ACE with the average squared loss between $B_t^{(i)} u_t^{(i)} + w_t^{(i)}$ and $f_t^{(i)}$.

Lemma 1. *Assume $x_1^{(i)} = 0, \forall i$. The average control error (*ACE*) is bounded as:*

$$\frac{\sum_{i=1}^N \sum_{t=1}^T \|x_t^{(i)}\|}{TN} \leq \frac{\gamma}{1-\rho} \sqrt{\frac{\sum_{i=1}^N \sum_{t=1}^T \|B_t^{(i)} u_t^{(i)} - f_t^{(i)} + w_t^{(i)}\|^2}{TN}}. \tag{3}$$

The proof can be found in the Appendix A.1. We assume $x_1^{(i)} = 0$ for simplicity: the influence of non-zero and bounded $x_1^{(i)}$ is a constant term in each outer iteration, from the e-ISS property.

Remark. Note that a stable linear system $f_0(x_t) = Ax_t$ with A's spectral radius < 1 satisfies the exponential ISS (e-ISS) assumption. For nonlinear systems, the e-ISS property of f_0 is usually achieved by applying some stable feedback controller to the system[1], i.e., f_0 is the closed-loop dynamics [10]. e-ISS assumption is standard in both online adaptive linear control [20, 14, 21] and nonlinear control [12]. In ACE we consider a regulation task, but it can also capture trajectory tracking task with time-variant nominal dynamics f_0 under incremental stability assumptions [12]. We only consider the regulation task in this paper for simplicity.

Generality. We would like to emphasize the generality of our model (1). The nominal control-affine part can model general fully-actuated robotic systems via Euler-Langrange equations [17, 18], and the unknown part $f(x, c)$ is nonlinear in x and c. We only need to assume the disturbance $w_t^{(i)}$ is bounded, which is more general than stochastic settings in linear [14, 13] and nonlinear [12] cases. For example, $w_t^{(i)}$ can model extra (x, u, c)-dependent uncertainties or adversarial disturbances. Moreover, the environment sequence $c^{(1:N)}$ could also be adversarial.

3 Online meta-adaptive control (OMAC) algorithm

The design of our online meta-adaptive control (OMAC) approach comprises four pieces: the policy class, the function class, the inner loop (within environment) adaptation algorithm $\mathcal{A}_2$, and the outer loop (between environment) online learning algorithm $\mathcal{A}_1$.

Policy class. We focus on the class of certainty-equivalence controllers [12, 22, 13], which is a general class of model-based controllers that also includes linear feedback controllers commonly studied in online control [22, 13]. After a model is learned from past data, and then a controller is designed by treating the learned model as the truth [11]. Formally, at `step`(i, t), the controller first estimates $\hat{f}_t^{(i)}$ (an estimation of $f_t^{(i)}$) based on past data, and then executes $u_t^{(i)} = B_t^{(i)\dagger} \hat{f}_t^{(i)}$, where $(\cdot)^\dagger$ is the pseudo inverse. Note that from Lemma 1, the average control error of the *omniscient controller* using $\hat{f}_t^{(i)} = f_t^{(i)}$ (i.e., the controller perfectly knows $f(x, c)$) is upper bounded as:[2]

$$\mathsf{ACE}(\text{omniscient}) \leq \gamma W / (1-\rho).$$

[1]For example, consider $x_{t+1} = \frac{3}{2} x_t + 2 \sin x_t + \bar{u}_t$. With a feedback controller $\bar{u}_t = u_t - x_t - 2 \sin x_t$, the closed-loop dynamics is $x_{t+1} = \frac{1}{2} x_t + u_t$, where $f_0(x) = \frac{1}{2} x$ is e-ISS.

[2]This upper bound is tight. Consider a scalar system $x_{t+1} = ax_t + u_t - f(x_t) + w$ with $|a| < 1$ and w a constant. In this case $\rho = |a|, \gamma = 1$, and the omniscient controller $u_t = f(x_t)$ yields $\mathsf{ACE} = \gamma |w| / (1-\rho)$.

$\mathsf{ACE}(\text{omniscient})$ can be viewed as a fundamental limit of the certainty-equivalence policy class.

Function class F for representation learning. In OMAC, we need to define a function class $F(\phi(x;\hat{\Theta}),\hat{c})$ to compute $\hat{f}_t^{(i)}$ (i.e., $\hat{f}_t^{(i)} = F(\phi(x_t^{(i)};\hat{\Theta}),\hat{c})$), where ϕ (parameterized by $\hat{\Theta}$) is a representation shared by all environments, and $\hat{c}$ is an environment-specific latent state. From a theoretical perspective, the main consideration of the choice of $F(\phi(x),\hat{c})$ is on how it effects the resulting learning objective. For instance, ϕ represented by a Deep Neural Network (DNN) would lead to a highly non-convex learning objective. In this paper, we focus on the setting $\hat{\Theta} \in \mathbb{R}^p, \hat{c} \in \mathbb{R}^h$, and $p \gg h$, i.e., it is much more expensive to learn the shared representation ϕ (e.g., a DNN) than "fine-tuning" via $\hat{c}$ in a specific environment, which is consistent with meta-learning [8] and representation learning [4, 6] practices.

Inner loop adaptive control. We take a modular approach in our algorithm design, in order to cleanly leverage state-of-the-art methods from online learning, representation learning, and adaptive control. When interacting with a single environment (for T time steps), we keep the learned representation ϕ fixed, and use that representation for adaptive control by treating $\hat{c}$ as an unknown low-dimensional parameter. We can utilize any adaptive control method such as online gradient descent, velocity gradient, or composite adaptation [23, 12, 10, 1].

Outer loop online learning. We treat the outer loop (which iterates between environments) as an online learning problem, where the goal is to learn the shared representation ϕ that optimizes the inner loop adaptation performance. Theoretically, we can reason about the analysis by setting up a hierarchical or nested online optimization procedure (adaptive control nested within online learning).

Design goal. Our goal is to design a meta-adaptive controller that has low ACE, ideally converging to $\mathsf{ACE}(\text{omniscient})$ as $T, N \to \infty$. In other words, OMAC should converge to performing as good as the omniscient controller with perfect knowledge of $f(x,c)$.

Algorithm 1 describes the OMAC algorithm. Since ϕ is environment-invariant and $p \gg h$, we only adapt $\hat{\Theta}$ at the end of each outer iteration. On the other hand, because $c^{(i)}$ varies in different environments, we adapt $\hat{c}$ at each inner step. As shown in Algorithm 1, at $\texttt{step}(i,t)$, after applying $u_t^{(i)}$, the controller observes the next state $x_{t+1}^{(i)}$ and computes: $y_t^{(i)} \triangleq f_0(x_t^{(i)}) + B_t^{(i)} u_t^{(i)} - x_{t+1}^{(i)} = f_t^{(i)} - w_t^{(i)}$, which is a disturbed measurement of the ground truth $f_t^{(i)}$. We then define $\ell_t^{(i)}(\hat{\Theta},\hat{c}) \triangleq \|F(\phi(x_t^{(i)};\hat{\Theta}),\hat{c}) - y_t^{(i)}\|^2$ as the observed loss at $\texttt{step}(i,t)$, which is a squared loss between the disturbed measurement of $f_t^{(i)}$ and the model prediction $\hat{f}_t^{(i)}$.

Algorithm 1: Online Meta-Adaptive Control (OMAC) algorithm

Input: Meta-adapter $\mathcal{A}_1$; inner-adapter $\mathcal{A}_2$; model $F(\phi(x;\hat{\Theta}),\hat{c})$; Boolean `ObserveEnv`
for $i = 1, \cdots, N$ **do**
  The environment select $c^{(i)}$
  for $t = 1, \cdots, T-1$ **do**
    Compute $\hat{f}_t^{(i)} = F(\phi(x_t^{(i)};\hat{\Theta}^{(i)}),\hat{c}_t^{(i)})$
    Execute $u_t^{(i)} = B_t^{(i)\dagger} \hat{f}_t^{(i)}$ `//certainty-equivalence policy`
    Observe $x_{t+1}^{(i)}, y_t^{(i)} \triangleq f(x_t^{(i)}, c^{(i)}) - w_t^{(i)}$, and $\ell_t^{(i)}(\hat{\Theta},\hat{c}) \triangleq \|F(\phi(x_t^{(i)};\hat{\Theta}),\hat{c}) - y_t^{(i)}\|^2$
      `//`$y_t^{(i)}$ `is a noisy measurement of` f `and` $\ell_t^{(i)}$ `is the observed loss`
    Construct an inner cost function $g_t^{(i)}(\hat{c})$ by $\mathcal{A}_2$ `//`$g_t^{(i)}$ `is a function of` $\hat{c}$
    Inner-adaptation: $\hat{c}_{t+1}^{(i)} \leftarrow \mathcal{A}_2(\hat{c}_t^{(i)}, g_{1:t}^{(i)})$
  if `ObserveEnv` **then** Observe $c^{(i)}$ `//only used in some instantiations`
  Construct an outer cost function $G^{(i)}(\hat{\Theta})$ by $\mathcal{A}_1$ `//`$G^{(i)}$ `is a function of` $\hat{\Theta}$
  Meta-adaptation: $\hat{\Theta}^{(i+1)} \leftarrow \mathcal{A}_1(\hat{\Theta}^{(i)}, G^{(1:i)})$

Instantiations. Depending on $\{F(\phi(x;\hat{\Theta}),\hat{c}), \mathcal{A}_1, \mathcal{A}_2, \texttt{ObserveEnv}\}$, we consider three settings:

- **Convex case** (Section 4.1): The observed loss $\ell_t^{(i)}$ is convex with respect to $\hat{\Theta}$ and $\hat{c}$.
- **Element-wise convex case** (Section 4.2): Fixing $\hat{\Theta}$ or $\hat{c}$, $\ell_t^{(i)}$ is convex with respect to the other.
- **Deep learning case** (Section 5): In this case, we use a DNN with weight $\hat{\Theta}$ to represent ϕ.

4 Main theoretical results

4.1 Convex case

In this section, we focus on a setting where the observed loss $\ell_t^{(i)}(\hat{\Theta}, \hat{c}) = \|F(\phi(x; \hat{\Theta}), \hat{c}) - y_t^{(i)}\|^2$ is convex with respect to $\hat{\Theta}$ and $\hat{c}$. We provide the following example to illustrate this case and highlight the difference between conventional adaptive control (e.g., [12, 11]).

Example 1. We consider a model $F(\phi(x; \hat{\Theta}), \hat{c}) = Y_1(x)\hat{\Theta} + Y_2(x)\hat{c}$ to estimate f:

$$\hat{f}_t^{(i)} = Y_1(x_t^{(i)})\hat{\Theta}^{(i)} + Y_2(x_t^{(i)})\hat{c}_t^{(i)}, \tag{4}$$

where $Y_1 : \mathbb{R}^n \to \mathbb{R}^{n\times p}, Y_2 : \mathbb{R}^n \to \mathbb{R}^{n\times h}$ are two known bases. Note that conventional adaptive control approaches typically concatenate $\hat{\Theta}$ and $\hat{c}$ and adapt on both at each time step, regardless of the environment changes. Since $p \gg h$, such concatenation is computationally much more expensive than OMAC, which only adapts $\hat{\Theta}$ in outer iterations.

Because $\ell_t^{(i)}(\hat{\Theta}, \hat{c})$ is *jointly* convex with respective to $\hat{\Theta}$ and $\hat{c}$, the OMAC algorithm in this case falls into the category of Nested Online Convex Optimization (Nested OCO) [24]. The choice of $g_t^{(i)}, G^{(i)}, \mathcal{A}_1, \mathcal{A}_2$ and `ObserveEnv` are depicted in Table 1. Note that in the convex case OMAC does not need to known $c^{(i)}$ in the whole process (`ObserveEnv` $=$ False).

$F(\phi(x; \hat{\Theta}), \hat{c})$	Any F model such that $\ell_t^{(i)}(\hat{\Theta}, \hat{c})$ is convex (e.g., Example 1)
$g_t^{(i)}(\hat{c})$	$\nabla_{\hat{c}}\ell_t^{(i)}(\hat{\Theta}^{(i)}, \hat{c}_t^{(i)}) \cdot \hat{c}$
$G^{(i)}(\hat{\Theta})$	$\sum_{t=1}^{T} \nabla_{\hat{\Theta}}\ell_t^{(i)}(\hat{\Theta}^{(i)}, \hat{c}_t^{(i)}) \cdot \hat{\Theta}$
$\mathcal{A}_1$	With a convex set $\mathcal{K}_1$, $\mathcal{A}_1$ initializes $\hat{\Theta}^{(1)} \in \mathcal{K}_1$ and returns $\hat{\Theta}^{(i+1)} \in \mathcal{K}_1, \forall i$. $\mathcal{A}_1$ has sublinear regret, i.e., the total regret of $\mathcal{A}_1$ is $T \cdot o(N)$ (e.g., online gradient descent)
$\mathcal{A}_2$	With a convex set $\mathcal{K}_2$, $\forall i$, $\mathcal{A}_2$ initializes $\hat{c}_1^{(i)} \in \mathcal{K}_2$ and returns $\hat{c}_{t+1}^{(i)} \in \mathcal{K}_2, \forall t$. $\mathcal{A}_2$ has sublinear regret, i.e., the total regret of $\mathcal{A}_2$ is $N \cdot o(T)$ (e.g., online gradient descent)
`ObserveEnv`	False

Table 1: OMAC with convex loss

As shown in Table 1, at the end of `step`(i, t) we fix $\hat{\Theta} = \hat{\Theta}^{(i)}$ and estimate $\hat{c}_{t+1}^{(i)} \in \mathcal{K}_2$ using $\mathcal{A}_2(\hat{c}_t^{(i)}, g_{1:t}^{(i)})$, which is an OCO problem with linear costs $g_{1:t}^{(i)}$. On the other hand, at the end of outer iteration i, we estimate $\hat{\Theta}^{(i+1)} \in \mathcal{K}_1$ using $\mathcal{A}_1(\hat{\Theta}^{(i)}, G^{(1:i)})$, which is another OCO problem with linear costs $G^{(1:i)}$. From [24], we have the following regret relationship:

Lemma 2 (Nested OCO regret bound, [24])**.** *OMAC (Algorithm 1) specified by Table 1 has regret:*

$$\begin{aligned}
&\sum_{i=1}^{N}\sum_{t=1}^{T} \ell_t^{(i)}(\hat{\Theta}^{(i)}, \hat{c}_t^{(i)}) - \min_{\Theta\in\mathcal{K}_1}\sum_{i=1}^{N}\min_{c^{(i)}\in\mathcal{K}_2}\sum_{t=1}^{T}\ell_t^{(i)}(\Theta, c^{(i)}) \\
&\quad\le \underbrace{\sum_{i=1}^{N} G^{(i)}(\hat{\Theta}^{(i)}) - \min_{\Theta\in\mathcal{K}_1}\sum_{i=1}^{N} G^{(i)}(\Theta)}_{\textit{the total regret of } \mathcal{A}_1, T\cdot o(N)} + \underbrace{\sum_{i=1}^{N}\sum_{t=1}^{T} g_t^{(i)}(\hat{c}_t^{(i)}) - \sum_{i=1}^{N}\min_{c^{(i)}\in\mathcal{K}_2}\sum_{t=1}^{T} g_t^{(i)}(c^{(i)})}_{\textit{the total regret of } \mathcal{A}_2, N\cdot o(T)}.
\end{aligned} \tag{5}$$

With Lemmas 1 and 2, we have the following guarantee for the average control error.

Theorem 3 (OMAC ACE bound with convex loss)**.** *Assume the unknown dynamics model is $f(x, c) = F(\phi(x; \Theta), c)$. Assume the true parameter $\Theta \in \mathcal{K}_1$ and $c^{(i)} \in \mathcal{K}_2, \forall i$. Then OMAC (Algorithm 1) specified by Table 1 ensures the following* ACE *guarantee:*

$$\mathsf{ACE} \le \frac{\gamma}{1-\rho}\sqrt{W^2 + \frac{o(T)}{T} + \frac{o(N)}{N}}.$$

To further understand Theorem 3 and compare with conventional adaptive control approaches, we provide the following corollary for Example 1.

Corollary 4. *Assume the unknown dynamics model is $f(x, c) = Y_1(x)\Theta + Y_2(x)c$, where $Y_1 : \mathbb{R}^n \to \mathbb{R}^{n\times p}, Y_2 : \mathbb{R}^n \to \mathbb{R}^{n\times h}$ are known bases. We assume $\|\Theta\| \le K_\Theta$ and $\|c^{(i)}\| \le K_c, \forall i$. Moreover, we assume $\|Y_1(x)\| \le K_1$ and $\|Y_2(x)\| \le K_2, \forall x$. Suppose in Table 1 we use $\hat{f}_t^{(i)} = Y_1(x_t^{(i)})\hat{\Theta}^{(i)} + Y_2(x_t^{(i)})\hat{c}_t^{(i)}$, and Online Gradient Descent (OGD) for both $\mathcal{A}_1$ and $\mathcal{A}_2$, with learning rate $\bar{\eta}^{(i)}$ and $\eta_t^{(i)}$ respectively. We set $\mathcal{K}_1 = \{\hat{\Theta} : \|\hat{\Theta}\| \le K_\Theta\}$ and $\mathcal{K}_2 = \{\hat{c} : \|\hat{c}\| \le K_c\}$. If we schedule the learning rates as:*

$$\bar{\eta}^{(i)} = \frac{K_\Theta}{\underbrace{(4K_1^2K_\Theta + 4K_1K_2K_c + 2K_1W)}_{C_1}T\sqrt{i}}, \ \eta_t^{(i)} = \frac{K_c}{\underbrace{(4K_2^2K_c + 4K_1K_2K_\Theta + 2K_2W)}_{C_2}\sqrt{t}},$$

then the average control performance is bounded by:

$$\mathsf{ACE}(\mathrm{OMAC}) \le \frac{\gamma}{1-\rho}\sqrt{W^2 + \frac{3}{2}\left(K_\Theta C_1\frac{1}{\sqrt{N}} + K_cC_2\frac{1}{\sqrt{T}}\right)}.$$

Moreover, the baseline adaptive control which uses OGD to adapt $\hat{\Theta}$ and $\hat{c}$ at each time step satisfies:

$$\mathsf{ACE}(\text{baseline adaptive control}) \le \frac{\gamma}{1-\rho}\sqrt{W^2 + \frac{3}{2}\sqrt{K_\Theta^2 + K_c^2}\sqrt{C_1^2 + C_2^2}\frac{1}{\sqrt{T}}}.$$

If $p \gg h$, we potentially have $K_1 \gg K_2$ and $K_\Theta \gg K_c$, so $C_1 \gg C_2$. Therefore, the ACE upper bound of OMAC is better than the baseline adaptation if $N > T$, which is consistent with recent representation learning results [4, 5]. It is also worth noting that the baseline adaptation is much more computationally expensive, because it needs to adapt both $\hat{\Theta}$ and $\hat{c}$ at each time step. Intuitively, OMAC improves convergence because the meta-adapter $\mathcal{A}_1$ approximates the environment-invariant part $Y_1(x)\Theta$, which makes the inner-adapter $\mathcal{A}_2$ much more efficient.

4.2 Element-wise convex case

In this subsection, we consider the setting where the loss function $\ell_t^{(i)}(\hat{\Theta}, \hat{c})$ is element-wise convex with respect to $\hat{\Theta}$ and $\hat{c}$, i.e., when one of $\hat{\Theta}$ or $\hat{c}$ is fixed, $\ell_t^{(i)}$ is convex with respect to the other one. For instance, F could be function as depicted in Example 2. Outside the context of control, such bilinear models are also studied in representation learning [4, 5] and factorization bandits [25, 26].

Example 2. We consider a model $F(\phi(x; \hat{\Theta}), \hat{c}) = Y(x)\hat{\Theta}\hat{c}$ to estimate f:

$$\hat{f}_t^{(i)} = Y(x_t^{(i)})\hat{\Theta}^{(i)}\hat{c}_t^{(i)}, \tag{6}$$

where $Y : \mathbb{R}^n \to \mathbb{R}^{n\times\bar{p}}$ is a known basis, $\hat{\Theta}^{(i)} \in \mathbb{R}^{\bar{p}\times h}$, and $\hat{c}_t^{(i)} \in \mathbb{R}^h$. Note that the dimensionality of $\hat{\Theta}$ is $p = \bar{p}h$. Conventional adaptive control typically views $\hat{\Theta}\hat{c}$ as a vector in $\mathbb{R}^{\bar{p}}$ and adapts it at each time step regardless of the environment changes [12].

Compared to Section 4.1, the element-wise convex case poses new challenges: i) the coupling between $\hat{\Theta}$ and $\hat{c}$ brings inherent non-identifiability issues; and ii) statistical learning guarantees typical need i.i.d. assumptions on $c^{(i)}$ and $x_t^{(i)}$ [5, 4]. These challenges are further amplified by the underlying nonlinear system. Therefore in this section we set `ObserveEnv` $=$ True, i.e., the controller has access to the true environmental condition $c^{(i)}$ at the end of the ith outer iteration, which is practical in many systems such as drones with wind disturbances [1, 27].

$F(\phi(x; \hat{\Theta}), \hat{c})$	Any F model such that $\ell_t^{(i)}(\hat{\Theta}, \hat{c})$ is element-wise convex (e.g., Example 2)
$g_t^{(i)}(\hat{c})$	$\ell_t^{(i)}(\hat{\Theta}^{(i)}, \hat{c})$
$G^{(i)}(\hat{\Theta})$	$\sum_{t=1}^T \ell_t^{(i)}(\hat{\Theta}, c^{(i)})$
$\mathcal{A}_1$ & $\mathcal{A}_2$	Same as Table 1
`ObserveEnv`	True

Table 2: OMAC with element-wise convex loss

The inputs to OMAC for the element-wise convex case are specified in Table 2. Compared to the convex case in Table 1, difference includes i) the cost functions $g_t^{(i)}$ and $G^{(i)}$ are convex, not necessarily linear; and ii) since $\texttt{ObserveEnv} = \text{True}$, in $G^{(i)}$ we use the true environmental condition $c^{(i)}$. With inputs specified in Table 2, Algorithm 1 has ACE guarantees in the following theorem.

Theorem 5 (OMAC ACE bound with element-wise convex loss)**.** *Assume the unknown dynamics model is $f(x, c) = F(\phi(x; \Theta), c)$. Assume the true parameter $\Theta \in \mathcal{K}_1$ and $c^{(i)} \in \mathcal{K}_2, \forall i$. Then OMAC (Algorithm 1) specified by Table 2 ensures the following* ACE *guarantee:*

$$\mathsf{ACE} \leq \frac{\gamma}{1-\rho}\sqrt{W^2 + \frac{o(T)}{T} + \frac{o(N)}{N}}.$$

4.2.1 Faster convergence with sub Gaussian and environment diversity assumptions

Since the cost functions $g_t^{(i)}$ and $G^{(i)}$ in Table 2 are not necessarily strongly convex, the ACE upper bound in Theorem 5 is typically $\frac{\gamma}{1-\rho}\sqrt{W^2 + O(1/\sqrt{T}) + O(1/\sqrt{N})}$. However, for the bilinear model in Example 2, it is possible to achieve faster convergence with extra sub Gaussian and *environment diversity* assumptions.

$F(\phi(x; \hat{\Theta}), \hat{c})$	The bilinear model in Example 2
$g_t^{(i)}(\hat{c})$	$\ell_t^{(i)}(\hat{\Theta}^{(i)}, \hat{c})$
$G^{(i)}(\hat{\Theta})$	$\lambda\|\hat{\Theta}\|_F^2 + \sum_{j=1}^{i}\sum_{t=1}^{T} \ell_t^{(i)}(\hat{\Theta}, c^{(j)})$ with some $\lambda > 0$
$\mathcal{A}_1$	$\hat{\Theta}^{(i+1)} = \arg\min_{\hat{\Theta}} G^{(i)}(\hat{\Theta})$ (i.e., Ridge regression)
$\mathcal{A}_2$	Same as Table 1
`ObserveEnv`	True

Table 3: OMAC with bilinear model

With a bilinear model, the inputs to the OMAC algorithm are shown in Table 3. With extra assumptions on $w_t^{(i)}$ and the environment, we have the following ACE guarantees.

Theorem 6 (OMAC ACE bound with bilinear model)**.** *Consider an unknown dynamics model $f(x, c) = Y(x)\Theta c$. Assume each component of the disturbance $w_t^{(i)}$ is R-sub-Gaussian, $\|\Theta\|_F \leq K_\Theta$, $\|c^{(i)}\| \leq K_c, \forall i$, and $\|Y(x)\|_F \leq K_Y, \forall x$. Define $Z_t^{(j)} = c^{(j)\top} \otimes Y(x_t^{(j)}) \in \mathbb{R}^{n \times ph}$ and assume environment diversity: $\lambda_{\min}(\sum_{j=1}^{i}\sum_{t=1}^{T} Z_t^{(j)\top} Z_t^{(j)}) = \Omega(i)$. Then OMAC (Algorithm 1) specified by Table 3 has the following* ACE *guarantee (with probability at least $1-\delta$):*

$$\mathsf{ACE} \leq \frac{\gamma}{1-\rho}\sqrt{W^2 + \frac{o(T)}{T} + O(\frac{\log(NT/\delta)\log(N)}{N})}. \tag{7}$$

If we relax the environment diversity condition to $\Omega(\sqrt{i})$, the last term becomes $O(\frac{\log(NT/\delta)}{\sqrt{N}})$.

The sub-Gaussian assumption is widely used in statistical learning theory to obtain concentration bounds [28, 4]. The environment diversity assumption states that $c^{(i)}$ provides "new information" in every outer iteration such that the minimum eigenvalue of $\sum_{j=1}^{i}\sum_{t=1}^{T} Z_t^{(j)\top} Z_t^{(j)}$ grows as i increases. Note that we do not need $\lambda_{\min}(\sum_{j=1}^{i}\sum_{t=1}^{T} Z_t^{(j)\top} Z_t^{(j)})$ to increase as T goes up. Moreover, if we relax the condition to $\Omega(\sqrt{i})$, the ACE bound becomes worse than the general element-wise convex case (the last term is $O(1/\sqrt{N})$), which implies the importance of "linear" environment diversity ($\Omega(i)$). Task diversity has been shown to be important for representation learning [4, 29].

5 Deep OMAC and experiments

We now introduce deep OMAC, a deep representation learning based OMAC algorithm. Table 4 shows the instantiation. As shown in Table 4, Deep OMAC employs a DNN to represent ϕ, and uses

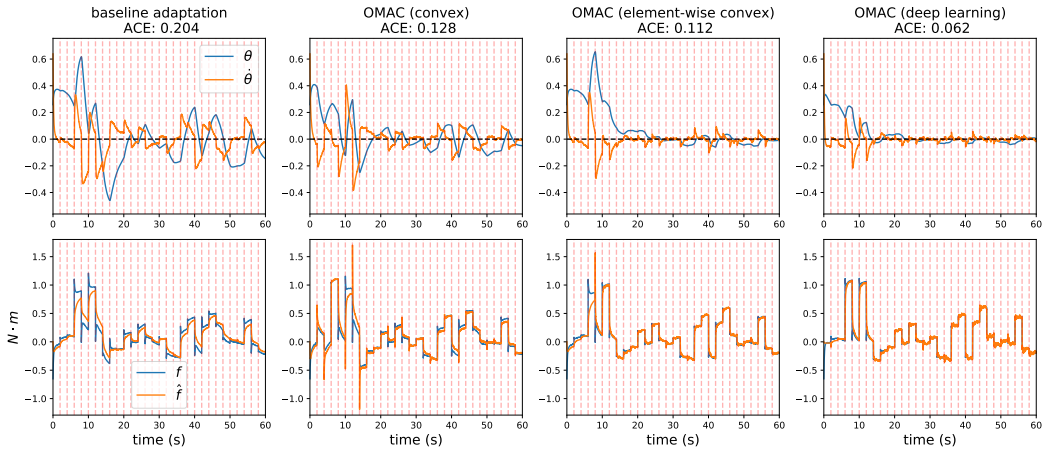

Figure 1: Trajectories of state (top) and $f_t^{(i)}, \hat{f}_t^{(i)}$ (bottom) from different methods. The environment (wind) is switched every 2 s, indicated by the red dashed lines.

gradient descent for optimization. With the model[3] $\phi(x;\hat{\Theta}) \cdot \hat{c}$, the meta-adapter $\mathcal{A}_1$ updates the representation ϕ via deep learning, and the inner-adapter $\mathcal{A}_2$ updates a linear layer $\hat{c}$.

$F(\phi(x;\hat{\Theta}),\hat{c})$	$\phi(x;\hat{\Theta}) \cdot \hat{c}$, where $\phi(x;\hat{\Theta}) : \mathbb{R}^n \to \mathbb{R}^{n\times h}$ is a neural network with weight $\hat{\Theta}$
$g_t^{(i)}(\hat{c})$	$\ell_t^{(i)}(\hat{\Theta}^{(i)},\hat{c})$
$\mathcal{A}_1$	Gradient descent: $\hat{\Theta}^{(i+1)} \leftarrow \hat{\Theta}^{(i)} - \eta\nabla_{\hat{\Theta}} \sum_{t=1}^{T} \ell_t^{(i)}(\hat{\Theta},\hat{c}_t^{(i)})$
$\mathcal{A}_2$	Same as Table 1
`ObserveEnv`	False

Table 4: OMAC with deep learning

In the experiments, we consider a nonlinear pendulum model with external wind disturbance, gravity mismatch, and unknown damping. The continuous-time model is $ml^2\ddot{\theta} - ml\hat{g}\sin\theta = u + f(\theta,\dot{\theta},c)$, where θ is the pendulum angle, $\dot{\theta}$ is the angular velocity, m is the pendulum mass and l is the length, $\hat{g}$ is the gravity estimation, c is the external unknown 2D wind, and $f(\theta,\dot{\theta},c)$ is the unknown dynamics model which depends on c, but also includes c-invariant terms such as damping and gravity mismatch. Note that for OMAC, the dimension of $\hat{c}$ is not necessarily equal to the dimension of c, which is typically unknown. We consider 6 different controllers in the experiment (see more details about the dynamics model and controllers in the Appendix A.7):

- **No-adapt** is simply using $\hat{f}_t^{(i)} = 0$, and **omniscient** is using $\hat{f}_t^{(i)} = f_t^{(i)}$.
- **OMAC (convex)** is based on Example 1, where $\hat{f}_t^{(i)} = Y_1(x_t^{(i)})\hat{\Theta}^{(i)} + Y_2(x_t^{(i)})\hat{c}_t^{(i)}$. We use random Fourier features [30, 12] to generate both $Y_1 \in \mathbb{R}^{30}$ and $Y_2 \in \mathbb{R}^{20}$. We use OGD for both $\mathcal{A}_1$ and $\mathcal{A}_2$ in Table 1.
- **OMAC (element-wise convex)** is based on Example 2, where $\hat{f}_t^{(i)} = Y(x_t^{(i)})\hat{\Theta}^{(i)}\hat{c}_t^{(i)}$, $\hat{\Theta}^{(i)} \in \mathbb{R}^{30\times 20}$. Similarly, we use random Fourier features to generate $Y \in \mathbb{R}^{30}$. Although the theoretical result in Section 4.2 requires `ObserveEnv` $=$ True, we relax this in our experiments and use $G^{(i)}(\hat{\Theta}) = \sum_{t=1}^{T} \ell_t^{(i)}(\hat{\Theta},\hat{c}_t^{(i)})$ in Table 2, instead of $\sum_{t=1}^{T} \ell_t^{(i)}(\hat{\Theta},c^{(i)})$. We also deploy OGD for $\mathcal{A}_1$ and $\mathcal{A}_2$. **Baseline** has the same procedure except with $\hat{\Theta}^{(i)} \equiv \hat{\Theta}^{(1)}$, i.e., it calls the inner-adapter $\mathcal{A}_2$ at every step and does not update $\hat{\Theta}$, which is standard in adaptive control [12, 11].
- **OMAC (deep learning)** is based on Table 4. We use a three-layer DNN with spectral normalization [31–33] to represent ϕ, and use the `Adam` optimizer [34] for $\mathcal{A}_1$. Same as other methods, we use $h = 20$, i.e., $\hat{c} \in \mathbb{R}^{20}$. $\mathcal{A}_2$ is also an OGD algorithm.

[3]The intuition behind this structure is that, any analytic function $f(x,c)$ can be approximated by $\phi(x)\hat{c}(c)$ with a universal approximator ϕ. We provide a detailed and theoretical justification in the Appendix A.7.

no-adapt	baseline	OMAC (convex)
0.663 ± 0.142	0.311 ± 0.112	0.147 ± 0.047
OMAC (element-wise convex)	OMAC (deep learning)	omniscient
0.129 ± 0.044	0.093 ± 0.027	0.034 ± 0.017

Table 5: ACE results in experiments

For all methods, we fix random seeds and randomly switch the environment (wind) c every 2 s. The total horizon is 60 s, i.e., $N = 30$. To make a fair comparison, except no-adapt or omniscient, all methods have the same learning rate for the inner-adapter $\mathcal{A}_2$ and the dimensions of $\hat{c}$ are also same. ACE results from 10 random seeds are depicted in Table 5. Figure 1 visualizes the trajectories of four methods. We observe that i) OMAC significantly outperforms the baseline; ii) OMAC adapts faster and faster as it encounters more environments but baseline cannot benefit from prior experience (see Figure 1), and iii) Deep OMAC achieves the best ACE due to the representation power of DNN.

6 Related work

Meta-learning and representation learning. Empirically, representation learning and meta-learning have shown great success in various domains [6]. In terms of control, meta-reinforcement learning is able to solve challenging mult-task RL problems [8, 9, 7]. We remark that learning representation for control also refers to learn *state* representation from rich observations [35–37], but we consider *dynamics* representation in this paper. On the theoretic side, [4, 5, 29] have shown representation learning reduces sample complexity on new tasks, and "task diversity" is critical. Consistently, we show OMAC enjoys better convergence theoretically (Corollary 4) and empirically, and Theorem 6 also implies the importance of *environment diversity*.

Adaptive control and online control. There is a rich body of literature studying Lyapunov stability and asymptotic convergence in adaptive control theory [10, 11]. Recently, there has been increased interest in studying online adaptive control with non-asymptotic metrics (e.g., regret) from learning theory, largely for settings with linear systems such as online LQR or LQG with unknown dynamics [14, 13, 15, 21, 20]. The most relevant work [12] gives the first regret bound of adaptive nonlinear control with unknown nonlinear dynamics and stochastic noise. Another relevant work studies online robust control of nonlinear systems with a mistake guarantee on the number of robustness failures [38]. However, all these results focus on the single-task case. To our knowledge, we show the first non-asymptotic convergence result for multi-task adaptive control. On the empirical side, [1, 27] combine adaptive nonlinear control with meta-learning, yielding encouraging experimental results.

Online matrix factorization. Our work bears affinity to online matrix factorization, particularly the bandit collaborative filtering setting [26, 39, 25]. In this setting, one typically posits a linear low-rank projection as the target representation (e.g., a low-rank factorization of the user-item matrix), which is similar to our bilinear case. Setting aside the significant complexity introduced by nonlinear control, a key similarity comes from viewing different users as "tasks" and recommended items as "actions". Prior work in this area has by and large not been able to establish strong regret bounds, in part due to the non-identifiability issue inherent in matrix factorization. In contrast, in our setting, one set of latent variables (e.g., the wind condition) has a concrete physical meaning that we are allowed to measure (`ObserveEnv` in Algorithm 1), thus side-stepping this non-identifiability issue.

7 Concluding remarks

We have presented OMAC, a meta-algorithm for adaptive nonlinear control in a sequence of unknown environments. We provide different instantiations of OMAC under varying assumptions and conditions, leading to the first non-asymptotic convergence guarantee for multi-task adaptive nonlinear control, and integration with deep learning. We also validate OMAC empirically. We use the average control error (ACE) metric and focus on fully-actuated systems in this paper. Future work will seek to consider general cost functions and systems. It is also interesting to study end-to-end convergence guarantees of deep OMAC, with ideas from deep representation learning theories. Another interesting direction is to study how to incorporate other areas of control theory, such as roboust control.

Broader impacts. This work is primarily focused on establishing a theoretical understanding of meta-adaptive control. Such fundamental work will not directly have broader societal impacts.

References

[1] Michael O'Connell, Guanya Shi, Xichen Shi, and Soon-Jo Chung. Meta-learning-based robust adaptive flight control under uncertain wind conditions. *arXiv preprint arXiv:2103.01932*, 2021.

[2] Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36 (4):1307–1319, 2020.

[3] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47), 2020.

[4] Simon S Du, Wei Hu, Sham M Kakade, Jason D Lee, and Qi Lei. Few-shot learning via learning the representation, provably. *arXiv preprint arXiv:2002.09434*, 2020.

[5] Nilesh Tripuraneni, Chi Jin, and Michael I Jordan. Provable meta-learning of linear representations. *arXiv preprint arXiv:2002.11684*, 2020.

[6] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828, 2013.

[7] Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. *arXiv preprint arXiv:1802.07245*, 2018.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

[9] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

[10] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control*, volume 199. Prentice hall Englewood Cliffs, NJ, 1991.

[11] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

[12] Nicholas M Boffi, Stephen Tu, and Jean-Jacques E Slotine. Regret bounds for adaptive nonlinear control. *arXiv preprint arXiv:2011.13101*, 2020.

[13] Max Simchowitz and Dylan Foster. Naive exploration is optimal for online lqr. In *International Conference on Machine Learning*, pages 8937–8948. PMLR, 2020.

[14] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, pages 1–47, 2019.

[15] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Logarithmic regret bound in partially observable linear dynamical systems. *arXiv preprint arXiv:2003.11227*, 2020.

[16] Chenkai Yu, Guanya Shi, Soon-Jo Chung, Yisong Yue, and Adam Wierman. The power of predictions in online control. *Advances in Neural Information Processing Systems*, 33, 2020.

[17] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[18] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[19] Karl Johan Åström and Richard M Murray. *Feedback systems*. Princeton university press, 2010.

[20] Yasin Abbasi-Yadkori and Csaba Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 1–26. JMLR Workshop and Conference Proceedings, 2011.

[21] Alon Cohen, Avinatan Hasidim, Tomer Koren, Nevena Lazic, Yishay Mansour, and Kunal Talwar. Online linear quadratic control. In *International Conference on Machine Learning*, pages 1029–1038. PMLR, 2018.

[22] Horia Mania, Stephen Tu, and Benjamin Recht. Certainty equivalence is efficient for linear quadratic control. *arXiv preprint arXiv:1902.07826*, 2019.

[23] Elad Hazan. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

[24] Naman Agarwal, Elad Hazan, Anirudha Majumdar, and Karan Singh. A regret minimization approach to iterative learning control. *arXiv preprint arXiv:2102.13478*, 2021.

[25] Huazheng Wang, Qingyun Wu, and Hongning Wang. Factorization bandits for interactive recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[26] Jaya Kawale, Hung H Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in neural information processing systems*, pages 1297–1305, 2015.

[27] Spencer M Richards, Navid Azizan, Jean-Jacques E Slotine, and Marco Pavone. Adaptive-control-oriented meta-learning for nonlinear systems. *arXiv preprint arXiv:2103.04490*, 2021.

[28] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, volume 11, pages 2312–2320, 2011.

[29] Nilesh Tripuraneni, Michael I Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. *arXiv preprint arXiv:2006.11650*, 2020.

[30] Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, volume 3, page 5. Citeseer, 2007.

[31] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[32] Guanya Shi, Xichen Shi, Michael O'Connell, Rose Yu, Kamyar Azizzadenesheli, Animashree Anandkumar, Yisong Yue, and Soon-Jo Chung. Neural lander: Stable drone landing control using learned dynamics. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9784–9790. IEEE, 2019.

[33] Guanya Shi, Wolfgang Hönig, Xichen Shi, Yisong Yue, and Soon-Jo Chung. Neural-swarm2: Planning and control of heterogeneous multirotor swarms using learned interactions. *arXiv preprint arXiv:2012.05457*, 2020.

[34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[35] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018.

[36] Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deep-mdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, pages 2170–2179. PMLR, 2019.

[37] Amy Zhang, Zachary C Lipton, Luis Pineda, Kamyar Azizzadenesheli, Anima Anandkumar, Laurent Itti, Joelle Pineau, and Tommaso Furlanello. Learning causal state representations of partially observable environments. *arXiv preprint arXiv:1906.10437*, 2019.

[38] Dimitar Ho, Hoang Le, John Doyle, and Yisong Yue. Online robust control of nonlinear systems with large uncertainty. In *International Conference on Artificial Intelligence and Statistics*, 2021.

[39] Guy Bresler, Devavrat Shah, and Luis Filipe Voloch. Collaborative filtering with low regret. In *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, 2016.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 7.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 7.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] All proofs are in Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide all the code and instructions in the supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5 and Appendix A.7.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 5.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix A.7.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]