Meta-Reinforcement Learning with Self-Modifying Networks

Mathieu Chalvidal
Artificial and Natural Intelligence Toulouse Institute
Universite de Toulouse, France
mathieu_chalvid@brown.edu

Thomas Serre
Carney Institute for Brain Science
Brown University, U.S.
thomas_serre@brown.edu

Rufin VanRullen
Centre de Recherche Cerveau & Cognition
CNRS, Universite de Toulouse, France
rufin.vanrullen@cnrs.fr

Abstract

Deep Reinforcement Learning has demonstrated the potential of neural networks tuned with gradient descent for solving complex tasks in well-delimited environments. However, these neural systems are slow learners producing specialized agents with no mechanism to continue learning beyond their training curriculum. On the contrary, biological synaptic plasticity is persistent and manifold, and has been hypothesized to play a key role in executive functions such as working memory and cognitive flexibility, potentially supporting more efficient and generic learning abilities. Inspired by this, we propose to build networks with dynamic weights, able to continually perform self-reflexive modification as a function of their current synaptic state and action-reward feedback, rather than a fixed network configuration. The resulting model, MetODS (for Meta-Optimized Dynamical Synapses) is a broadly applicable meta-reinforcement learning system able to learn efficient and powerful control rules in the agent policy space. A single layer with dynamic synapses can perform one-shot learning, generalize navigation principles to unseen environments and demonstrate a strong ability to learn adaptive motor policies.

1 Introduction

The algorithmic shift from hand-designed to learned features characterizing modern Deep Learning has been transformative for Reinforcement Learning (RL), allowing to solve complex problems ranging from video games [1, 2] to multiplayer contests [3] or motor control [4, 5]. Yet, "deep" RL has mostly produced specialized agents unable to cope with rapid contextual changes or tasks with novel or compositional structure [6–8]. The vast majority of models have relied on gradient-based optimization to learn static network parameters adjusted during a predefined curriculum, arguably preventing the emergence of online adaptivity. A potential solution to this challenge is to *meta-learn* [9–11] computational mechanisms able to rapidly capture a task structure and automatically operate complex feedback control: Meta-Reinforcement Learning constitutes a promising direction to build more adaptive artificial systems [12] and identify key neuroscience mechanisms that endow humans with their versatile learning abilities [13].

In this work, we draw inspiration from biological fast synaptic plasticity, hypothesized to orchestrate flexible cognitive functions according to context-dependent rules [14–17]. By tuning neuronal selectivity at fast time scales – from fast neural signaling (milliseconds) to experience-based learning

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

(seconds and beyond) – fast plasticity can support in principle many cognitive faculties including motor and executive control. From a dynamical system perspective, fast plasticity can serve as an efficient mechanism for information storage and manipulation and has led to modern theories of working memory [18–22]. Despite the fact that the magnitude of the synaptic gain variations may be small, such modifications are capable of profoundly altering the network transfer function [23] and constitute a plausible mechanism for rapidly converting reward and choice history into tuned neural functions [24]. From a machine learning perspective, despite having a long history [25–29], fast weights have most often been investigated in conjunction with recurrent neural activations [30–33] and rarely as a function of an external reward signal or of the current synaptic state itself. Recently, new proposals have shown that models with dynamic modulation related to fast weights could yield powerful meta-reinforcement learners [34–36]. In this work, we explore an original self-referential update rule that allows the model to form synaptic updates conditionally on information present in its own synaptic memory. Additionally, environmental reward is injected continually in the model as a rich feedback signal to drive the weight dynamics. These features endow our model with a unique recursive control scheme, that support the emergence of a self-contained reinforcement learning program.

Contribution: We demonstrate that a neural network trained to continually self-modify its weights as a function of sensory information and its own synaptic state can produce a powerful reinforcement learning program. The resulting general-purpose meta-RL agent called 'MetODS' (for Meta-Optimized Dynamical Synapses) is theoretically presented as a model-free approach performing stochastic feedback control in the policy space. In our experimental evaluation, we investigate the reinforcement learning strategies implemented by the model and demonstrate that a single layer with lightweight parametrization can implement a wide spectrum of cognitive functions, from one-shot learning to continuous motor-control. We hope that MetODS inspires more works around self-optimizing neural networks.

The remainder of the paper is organised as follows: In Section 2 we introduce our mathematical formulation of the meta-RL problem, which motivates MetODS computational principles presented in Section 3. In Section 4 we review previous approaches of meta-reinforcement learning and we discuss other models of artificial fast plasticity and their relation to associative memory. In Section 5 we report experimental results in multiple contexts. Finally, in Section 6 we summarise the main advantages of MetODS and outline future work directions.

2 Background

2.1 Notation

Throughout, we refer to "tasks" as Markov decision processes (MDP) defined by the following tuple $\tau = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho_0)$, where $\mathcal{S}$ and $\mathcal{A}$ are respectively the state and action sets, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ refers to the state transition distribution measure associating a probability to each tuple (state, action, new state), $r : \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is a bounded reward function and ρ_0 is the initial state distribution. For simplicity, we consider finite-horizon MDP with T time-steps although our discussion can be extended to the infinite horizon case as well as partially observed MDP. We further specify notation when needed by subscripting symbols with the corresponding task τ or time-step t.

2.2 Meta-Reinforcement learning as an optimal transport problem:

Meta-Reinforcement learning considers the problem of building a program generating a distribution μ_π of policies $\pi \in \Pi$ that are "adapted" for a distribution μ_τ of task $\tau \in \mathbb{T}$ with respect to a given metric $\mathcal{R}$. For instance, $\mathcal{R}(\tau, \pi)$ can be the expected cumulative reward for a task τ and policy π, i.e where state transitions are governed by $\boldsymbol{s}_{t+1} \sim \mathcal{P}_\tau(.|\boldsymbol{s}_t, \boldsymbol{a}_t)$, actions are sampled according to the policy π: $\boldsymbol{a}_t \sim \pi$ and initial state $\boldsymbol{s}_0$ follows the distribution $\rho_{0,\tau}$. Provided the existence of an optimal policy π^* for any task $\tau \in \mathbb{T}$, we can define the distribution measure μ_{π^*} of these policies over Π. Arguably, an ideal system aims at associating to any task τ its optimal policy π^*, i.e, finding the transport plan $\boldsymbol{\gamma}$ in the space $\Gamma(\mu_\mathbb{T}, \mu_{\pi^*})$ of coupling distributions with marginals $\mu_\mathbb{T}$ and μ_{π^*} that maximizes $\mathcal{R}$.

$$\max_{\gamma \in \Gamma(\mu_\mathbb{T}, \mu_{\pi^*})} \mathbb{E}_{(\tau,\pi)\sim\gamma}\Big[\mathcal{R}(\tau, \pi)\Big] \quad \text{where} \quad \mathcal{R}(\tau, \pi) = \mathbb{E}_{\pi,\mathcal{P}_\tau}\Big[\sum_{t=0}^{T} r_\tau(\boldsymbol{a}_t, \boldsymbol{s}_t)\Big] \tag{1}$$

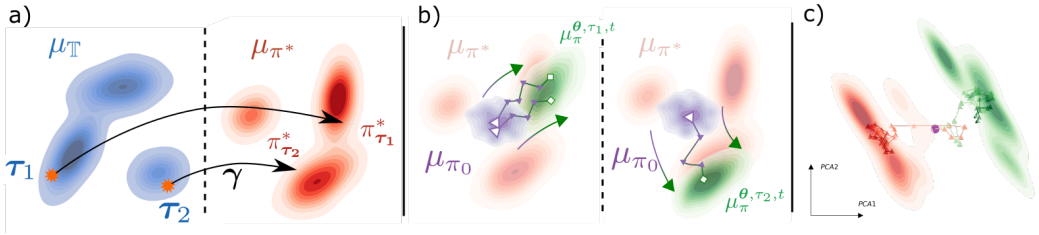

Figure 1: Meta-Reinforcement Learning as a transport problem and MetODS synaptic adaptation: **a)** Associating any task $\boldsymbol{\tau}$ in $\mathbb{T}$ to its optimal policy $\boldsymbol{\pi}^*_{\boldsymbol{\tau}}$ in $\boldsymbol{\Pi}$ can be regarded as finding an optimal transport plan from $\boldsymbol{\mu}_{\mathbb{T}}$ to $\boldsymbol{\mu}_{\boldsymbol{\pi}^*}$ with respect to the cost $-\boldsymbol{\mathcal{R}}$. Finding this transport plan is generally an intractable problem. **b)** Meta-RL approximates a solution by defining a stochastic flow in the policy space $\boldsymbol{\Pi}$ conditioned by the current task $\boldsymbol{\tau}$ and driving a prior distribution $\boldsymbol{\mu}_{\boldsymbol{\pi}_0}$ of policies $\boldsymbol{\pi}_0$ towards a distribution $\mu_{\pi}^{\boldsymbol{\theta},\tau,t}$ of policies with high score $\boldsymbol{\mathcal{R}}$. **c)** Density and mean trajectories of our model dynamic weights principal components over several episodes of the Harlow task (see section 5.1) reveal this policy specialization. Two modes colored with respect to whether the agent initial guess was good or bad emerge, corresponding to two different policies to solve the task.

Most generally, problem (1) is intractable, since μ_{π^*} is unknow or has no explicit form. Instead, previous approaches optimize a surrogate problem, by defining a parametric specialization procedure which builds for any task τ, a sequence $(\pi_t)_t$ of improving policies (see Fig. 1). Defining $\boldsymbol{\theta}$ some meta-parameters governing the evolution of the sequences $(\pi_t)_t$ and $\mu_{\pi}^{\boldsymbol{\theta},\tau,t}$ the distribution measure of the policy π_t after learning task τ during some period t, the optimization problem amounts to finding the meta-parameters $\boldsymbol{\theta}$ that best adapt $\pi_t \sim \mu_{\pi}^{\boldsymbol{\theta},\tau,t}$ over the task distribution.

$$\max_{\boldsymbol{\theta}} \;\; \mathbb{E}_{\tau\sim\mu_{\mathbb{T}}}\left[\mathbb{E}_{\pi\sim\mu_{\pi}^{\boldsymbol{\theta},\tau,t}}\left[\mathcal{R}(\tau,\pi)\right]\right] \tag{2}$$

Equation (2) cast meta-Reinforcement learning as the problem of building automatic policy control in Π. We discuss below desirable properties of such control for which we show that our proposed meta-learnt synaptic rule has a good potential.

$\diamond$ **Efficiency:** How "fast" the distribution $\mu_{\pi}^{\boldsymbol{\theta},\tau,t}$ is transported towards a distribution of high-performing policies. Arguably, an efficient learning mechanism should require few interaction steps t with the environment to identify the task rule and adapt its policy accordingly. This can be seen as minimizing the agent's cumulative regret during learning [37]. This notion is also connected to the policy exploration-exploitation trade-off [38, 39], where the agent learning program should foster rapid discovery of policy improvements while retaining performance gains acquired through exploration. We show that our learnt synaptic update rule is able to change the agent transfer function drastically in a few updates, and settle to a policy when task structure has been identified, thus supporting one-shot learning of a task-contingent association rule in the Harlow task, adapting a motor policy in a few steps or exploring original environments quickly in the Maze experiment.

$\diamond$ **Capacity:** That property defines the learner sensitivity to particular task structures and its ability to convert them into precise states in the policy space, which determines the achievable level of performance for a distribution of tasks μ_{τ}. This notion is linked to the sensitivity of the learner, i.e how the agent captures and retains statistics and structures describing a task. Precedent work has shown that memory-based Meta-RL models operate a Bayesian task regression problem [40], and further fostering task identification through diverse regularizations [41, 42, 39] benefited performance of the optimized learning strategy. Because our mechanism is continual, it allows for constant tracking of the environment information and policy update similar to memory-based models. We particularly test this property in the maze experiment in Section 5, showing that tuned online synaptic updates obtain the best capacity under systematic variation of the environment.

$\diamond$ **Generality:** We refer here to the overall ability of the meta-learnt policy flow to drive $\mu_{\pi}^{\boldsymbol{\theta},\tau,t}$ towards high-performing policy regions for a diverse set of tasks (*generic trainability*) but also to how general is the resulting reinforcement learning program and how well it transfers to tasks unseen during training (*transferability*). In the former case. since the proposed synaptic mechanism is model-free, it allows for tackling diverse types of policy learning, from navigation to motor control. Arguably, to build reinforcing agents that learn in open situations, we should strive for generic and efficient computational mechanisms rather than learnt heuristics. *Transferability* corresponds to the ability of the meta-learned policy flow to generally yield improving updates even in unseen policy regions

of the space $\mathbf{\Pi}$ or conditioned by unseen task properties: new states, actions and transitions, new reward profile. etc. We show in a motor-control experiment using the Meta-World benchmark that meta-tuned synaptic updates are a potential candidate to produce a more systematic learner agnostic to environment setting and reward profile. The generality property remains the hardest for current meta-RL approaches, demonstrating the importance of building more stable and invariant control principles.

3 MetODS: Meta-Optimized Dynamical Synapses

3.1 Learning reflexive weights updates

What if a neural agent could adjust the rule driving the evolution of $\mu_\pi^{\boldsymbol{\theta},\tau,t}$ based on its own knowledge? In this work, we test a neural computational model that learns to compress experience of a task τ into its dynamic weights $\boldsymbol{W}_t$.

$$\forall t \leq T, \qquad \pi(\boldsymbol{a}|\boldsymbol{s}, \boldsymbol{W}_t) \sim \mu_\pi^{\boldsymbol{\theta},\tau,t} \tag{3}$$

Contrary to gradient-based rules whose analytical expression is a static and predefined function of activations error ($\Delta(\boldsymbol{W}_t) = \frac{\partial \boldsymbol{a}}{\partial \boldsymbol{W}} \cdot \frac{\partial \mathcal{L}}{\partial \boldsymbol{a}}$), we define an update rule that directly depends on the current weight state through a parameterized non-linear recursive scheme, making the expression of the update much more sensitive to weight content itself.

$$\forall t \leq T, \qquad \Delta(\boldsymbol{W}_t) = \mathcal{F}_\theta(\boldsymbol{W}_t)\Big|_{\boldsymbol{W}=\boldsymbol{W}_t} \tag{4}$$

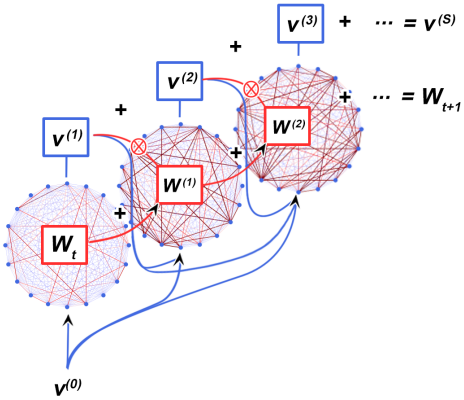

Figure 2: A layer of MetODS updates its weights by recursive applications of **read** and **write** operations based on neural activations $\boldsymbol{v}^{(s)}$ and synaptic traces $\boldsymbol{W}^{(s)}$.

This self-reflexive evolution of the synaptic configuration $\boldsymbol{W}_t$ allows for online variation of the learning rule during adaptation for a given task τ and opens an original repertoire of dynamics for the policy evolution $\mu_\pi^{\boldsymbol{\theta},\tau,t}$. Specifically, at every time-step t, network inference and learning consists in recursive applications of read and write operations that we define below.

◇ **Read-write operations:** The core mechanism consists in two simple operations that respectively linearly project neurons activation $\boldsymbol{v}$ through the dynamic weights $\boldsymbol{W}$ followed by an element-wise non-linearity, and build a local weight update with element-wise weighting $\boldsymbol{\alpha}$:

$$\begin{cases} \phi(\boldsymbol{W}, \boldsymbol{v}) = \sigma(\boldsymbol{W}.\boldsymbol{v}) & \textit{read} \\ \psi(\boldsymbol{v}) = \boldsymbol{\alpha} \odot \boldsymbol{v} \otimes \boldsymbol{v} & \textit{write} \end{cases} \tag{5}$$

Here, $\boldsymbol{\alpha}$ is a matrix of $\mathbb{R}^{N\times N}$, $\otimes$ denotes the outer-product, $\odot$ is the element-wise multiplication, σ is a non-linear activation function. The read and write operations are motivated by biological synaptic computation: Reading corresponds to the non-linear response of the neuron population to a specific activation pattern. Writing consists in an outer product that emulates a local Hebbian rule between neurons. The element-wise weighting $\boldsymbol{\alpha}$ allows locally tuning synaptic plasticity at every connection consistent with biology [43, 44] which generates a matrix update with potentially more than rank one as in the classic hebbian rule.

◇ **Recursive update:** While a single iteration of these two operations can only retrieve a similar activation pattern (reading) or add unfiltered external information into weights (writing), recursively applying these operations offers a much more potent computational mechanism that mix information between current neural activation and previous iterates. Starting from an initial activation pattern $\boldsymbol{v}^{(0)}$ and previous weight state $\boldsymbol{W}^{(0)} = \boldsymbol{W}_{t-1}$, the model recursively applies equations (5) for $s \in [1, S]$ on $\boldsymbol{v}^{(s)}$ and $\boldsymbol{W}^{(s)}$ such that:

$$\text{for } s \in [1, S] \quad : \quad \begin{cases} \boldsymbol{v}^{(s)} = \sum_{l=0}^{s-1} \boldsymbol{\kappa}_s^{(l)} \boldsymbol{v}^{(l)} + \boldsymbol{\kappa}_s^{(s)} \phi(\boldsymbol{W}^{(s-1)}, \boldsymbol{v}^{(s-1)}) \\ \boldsymbol{W}^{(s)} = \sum_{l=0}^{s} \boldsymbol{\beta}_s^{(l)} \boldsymbol{W}^{(l)} + \boldsymbol{\beta}_s^{(s)} \psi(\boldsymbol{v}^{(s-1)}) \end{cases} \tag{6}$$

Parameters $\boldsymbol{\kappa}_s^{(l)}$ and $\boldsymbol{\beta}_s^{(l)}$ are scalar values learnt along with plasticity parameters $\boldsymbol{\alpha}$, and correspond to delayed contributions of previous patterns and synaptic states to the current operations. This is motivated by biological evidence of complex cascades of temporal modulation mechanisms over synaptic efficacy [45, 46]. Finally, $(\boldsymbol{v}^{(S)}, \boldsymbol{W}^{(S)})$ are respectively used as activation for the next layer, and as the new synaptic state $\boldsymbol{W}_t$.

◇ **Computational interpretation:** We note that if S=1 in equation (6), the operation boils down to a simple hebbian update with a synapse-specific weighting $\boldsymbol{\alpha}^{i,j}$. This perspective makes MetODS an original form of modern Hopfield network [47] with hetero-associative memory that can dynamically access and edit stored representations driven by observations, rewards and actions. While pattern retrieval from Hopfield networks has a dense litterature, our recursive scheme is an original proposal to learn automatic updates able to articulate representations across timesteps. We believe that this mechanism is particularly benefitial for meta-RL to filter external information with respect to past experience. The promising results shown in our experimental section suggest that such learnt updates can generate useful self-modifications to sequentially adapt to incoming information at runtime.

In this work, we test a single dynamic layer for MetODS and leave the extension of the synaptic plasticity to the full network for future work. In order for the model to learn a credit assignment strategy, state transition and previous reward information $[\boldsymbol{s}_t, \boldsymbol{a}_{t-1}, \boldsymbol{r}_{t-1}]$ are embedded into a vector $\boldsymbol{v}_t$ by a feedforward map $\boldsymbol{f}$ as in previous meta-RL approaches [48, 49]. Action and advantage estimate are read-out by a feedforward policy map $\boldsymbol{g}$. We meta-learn the plasticity and update coefficients, as well as the embedding and read-out function altogether: $\boldsymbol{\theta} = [\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{\alpha}, \boldsymbol{\kappa}, \boldsymbol{\beta}]$. Additionally, the initial synaptic configuration $\boldsymbol{W}_0$ can be learnt, fixed a priori or sampled from a specified distribution.

Algorithm 1: MetODS synaptic learning

Require: $\boldsymbol{\theta} = [\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{\alpha}, \boldsymbol{\kappa}, \boldsymbol{\beta}]$ and $\boldsymbol{W}_0$
for $1 \leq t \leq T$ **do**
  $\boldsymbol{v}^{(0)} \leftarrow \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{a}_{t-1}, \boldsymbol{r}_{t-1})$
  $\boldsymbol{W}^{(0)} \leftarrow \boldsymbol{W}_{t-1}$
  for $1 \leq s \leq S$ **do**
    $\boldsymbol{v}^{(s)} \leftarrow \sum_{l=0}^{s-1} \boldsymbol{\kappa}_s^{(l)} \boldsymbol{v}^{(l)} + \boldsymbol{\kappa}_s^{(s)} \sigma(\boldsymbol{W}^{(s-1)}.\boldsymbol{v}^{(s-1)})$
    $\boldsymbol{W}^{(s)} \leftarrow \sum_{l=0}^{s-1} \boldsymbol{\beta}_s^{(l)} \boldsymbol{W}^{(l)} + \boldsymbol{\beta}_s^{(s)} \big(\boldsymbol{\alpha} \odot \boldsymbol{v}^{(s-1)} \otimes \boldsymbol{v}^{(s-1)}\big)$
  end for
  $\boldsymbol{a}_t, \boldsymbol{v}_t \leftarrow \boldsymbol{g}(\boldsymbol{v}^{(s)})$
  $\boldsymbol{W}_t \leftarrow \boldsymbol{W}^{(s)}$
end for

4 Related work

◇ **Meta-Reinforcement learning** has recently flourished into several different approaches aiming at learning high-level strategies for capturing task rules and structures. A direct line of work consists in automatically meta-learning components or parameters of the RL arsenal to improve over heuristic settings [50–52]. Orthogonally, work building on the Turing-completeness of recurrent neural networks has shown that simple recurrent neural networks can be trained to store past information in their persistent activity state to inform current decision, in such a way that the network implements a form of reinforcement learning over each episode [53, 48, 54]. It is believed that vanilla recurrent networks alone are not sufficient to meta-learn the efficient forms of episodic control found in biological agents [55, 13]. Hence additional work has tried to enhance the system with a better episodic memory model [56–58] or by modeling a policy as an attention module over an explicitely stored set of past events [49]. Optimization based approaches have tried to cast episodic adaptation as an explicit optimization procedure either by treating the optimizer as a black-box system [59, 60] or by learning a synaptic configuration such that one or a few gradient steps are sufficient to adapt the input/output mapping to a specific task [61].

◇ **Artificial fast plasticity:** Networks with dynamic weights that can adapt as a function of neural activation have shown promising results over regular recurrent neural networks to handle sequential data [62, 29, 31, 63, 32, 34]. However, contrary to our work, these models postulate a persistent neural activity orchestrating weights evolution. On the contrary, we show that synaptic states are the sole persistent components needed to perform fast adaptation. Additionally, the possibility of optimizing synaptic dynamics with evolutionary strategies in randomly initialized networks [64] or through gradient descent [65] has been demonstrated, as well as in a time-continuous setting

[66]. Recent results have shown that plasticity rules differentially tuned at the synapse level allow to dynamically edit and query networks memory [31, 67]. However another specificity of this work is that our model synaptic rule is a function of reward and synaptic state, allowing to drive weight dynamics conditionally on both an external feedback signal and the current model belief.

◇ **Associative memory:** As discussed above, efficient memory storage and manipulation is a crucial feature for building rapidly learning agents. To improve over vanilla recurrent neural network policies [48], some models have augmented recurrent agents with content-addressable dictionaries able to reinstate previously encoded patterns given the current state [68–70, 13]. However these slot-based memory systems are subject to interference with incoming inputs and their memory cost grows linearly with experience. Contrastingly, attractor networks can be learnt to produce fast compression of sensory information into a fixed size tensorial representation [71, 72]. One class of such network are Hopfield networks [73–76] which benefit from a large storage capacity [76], can possibly perform hetero-associative concept binding [77, 34] and produce fast and flexible information retrieval [47].

5 Experiments

In this section, we explore the potential of our meta-learnt synaptic update rule with respect to the three properties of the meta-RL problem exposed in section 2. Namely, 1) *efficiency*, 2) *capacity* and 3) *generality* of the produced learning algorithm. We compare it with three state-of-the-art meta-RL models based on different adaptation mechanisms: RL^2 [48] a memory-based algorithm based on training a GRU-cell to perform Reinforcement Learning within the hidden space of its recurrent unit, MAML [61] which performs online gradient descent on the weights of three-layer MLP and PEARL [78], which performs probabilistic task inference for conditioning policy. Details for each experimental setting are further discussed in S.I. and the code can be found at `https://github.com/mathieuchal/metods2022`.

5.1 Efficiency: One-shot reinforcement learning and rapid motor control

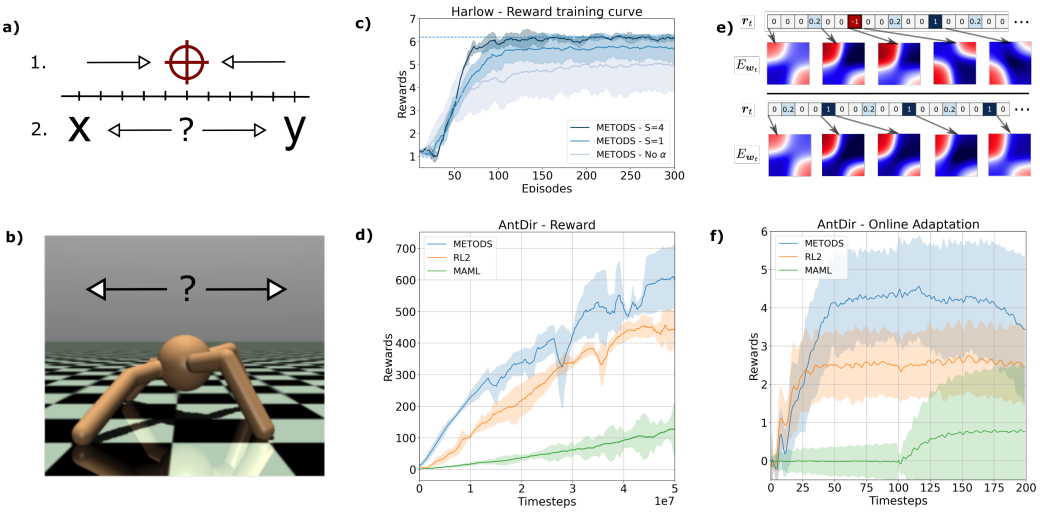

Figure 3: a-b) Schemas of the Harlow and Mujoco *Ant-directional* locomotion task. **c-d)** Evolution of accumulated reward over training. In the Harlow task, we conduct an ablation study by either reducing the number of recursive iterations (S=1) or removing the trainable plasticity weights $\boldsymbol{\alpha}$ resulting in sub-optimal policy. In *Ant-dir* we compare our agent training profile against MAML and RL^2. **e)** We can interpret the learned policy in terms of a Hopfield energy adapting with experience. We show horizontally two reward profiles of different episodes and the energy $E_{\boldsymbol{W}_t}(v_1, v_2) = -v_1^T \boldsymbol{W}_t v_2$ along two principal components of the vector trajectory $\boldsymbol{v_t}$. In the first episode, the error in the first presentation (red square) transforms the energy landscape which changes the agent policy, while on the other episode, the model belief does not change over time. Note the two modes for every energy map, which allows the model to handle the potential position permutation of the presented values. **f)** Average rewards per timestep during a single episode of the *Ant-dir* task.

To first illustrate that learnt synaptic dynamics can support fast behavioral adaptation, we use a classic experiment from the neuroscience literature originally presented by Harlow [79] and recently reintroduced in artificial meta-RL in [54] as well as a heavily-benchmarked MuJoCo directional locomotion task (see Fig. 3). To behave optimally in both settings, the agent must quickly identify the task rule and implement a relevant policy : The Harlow task consists of five sequential presentations of two random variables placed on a one-dimensional line with random permutation of their positions that an agent must select by reaching the corresponding position. One value is associated with a positive reward and the other with a negative reward. The five trials are presented in alternance with periods of fixation where the agent should return to a neutral position between items. In the MuJoCo robotic *Ant-dir* experiment, a 4-legged agent must produce a locomotive policy given a random rewarded direction.

◇ **Harlow**: Since values location are randomly permuted across presentations within one episode, the learner cannot develop a mechanistic strategy to reach high rewards based on initial position. Instead, to reach the maximal expected reward over the episode, the agent needs to perform one-shot learning of the task-contingent association rule during the first presentation. We found that even a very small network of N=20 neurons proved to be sufficient to solve the task perfectly. We investigated the synaptic mechanism encoding the agent policy. A principal component analysis reveals a policy differentiation with respect to the initial value choice outcome supported by synaptic specialization in only a few time-steps (see Figure 1-c). We can further interpret this adaptation in terms of sharp modifications of the Hopfield energy of the dynamic weights (Figure 3-e). Finally, the largest synaptic variations measured by the sum of absolute synaptic variations occur for states that carry a non-null reward signal (see S.I). These results suggest that the recursive hebbian update combined with reward feedback is sufficient to support one-shot reinforcement learning of the task association rule.

◇ **MuJoCo Ant-dir**: We trained models for performing the task on an episode of 200 timesteps and found that MetODS can adapt in a few time-steps, similar to memory-based models such as RL^2, (Figure 3-f) thanks to its continual adaptation mechanism. By design, MAML and PEARL do not present such a property, and they need multiple episodes before being able to perform adaptation correctly. We still report MAML performance after running its gradient adaptation at timestep t=100. We further note that our agent overall performance in a single episode is still superior to MAML performance reported in [61, 49] when more episodes are accessible for training.

5.2 Capacity : Maze exploration task

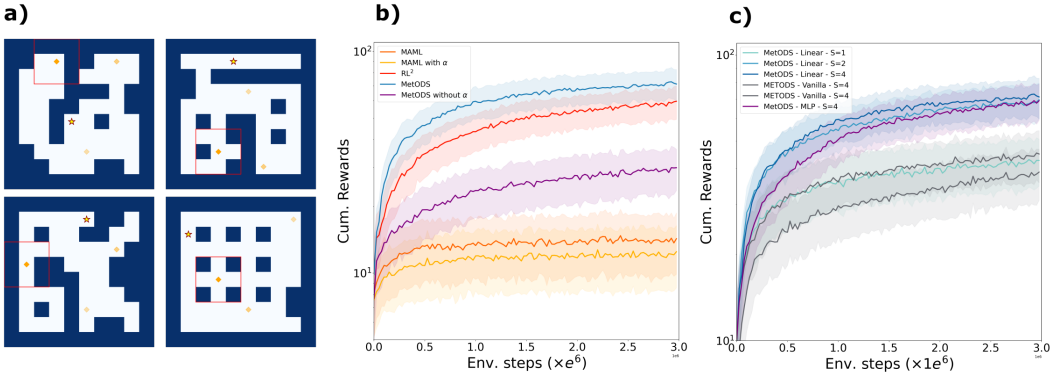

Figure 4: **a)** Examples of generated maze configurations. The mazes consist in 8 × 8 pix. areas, with walls and obstacles randomly placed according to a variation of Prim's algorithm [80] with target location randomly selected for the entire duration of the episode (star). The agent receptive field is highlighted in red. **b)** Comparisons with MAML and RL^2 and effect of element-wise plasticity $\boldsymbol{\alpha}$. **c)** Variations of MetODS writing mechanism as well as depth S of recursivity yield different performances (see. S.I for further details)

We further tested the systematicity of MetODS learnt reinforcement program on a partially observable Markov decision process (POMDP): An agent must locate a target in a randomly generated maze while starting from random locations and observing a small portion of its environment (depicted in Figure 4). While visual navigation has been previously explored in meta-RL [48, 49, 31], we here focus on the mnemonic component of navigation by complexifying the task in two ways, we reduce the agent's visual field to a small size of 3x3 pix. and randomize the agent's position after every reward encounter. The agent can take discrete actions in the set {up,down,left,right} which moves

it accordingly by one coordinate. The agent's reward signal is solely received by hitting the target location, thus receiving a reward of 10. Each time the agent hits the target, its position is randomly reassigned on the map (orange) and the exploration resumes until 100 steps are accumulated during which the agent must collect as much reward as possible. Note that the reward is invisible to the agent, and thus the agent only knows it has hit the reward location because of the activation of the reward input. The reduced observability of the environment and the sparsity of the reward signal (most of the state transitions yield no reward) requires the agent to perform logical binding between distant temporal events to navigate the maze. Again, this setting rules out PEARL since its latent context encoding mechanism erases crucial temporal dependencies between state transitions for efficient exploration. Despite having no particular inductive bias for efficient spatial exploration or path memorization, a strong policy emerges spontaneously from training.

Agent	1st rew.* ($\downarrow$)	Success ($\uparrow$)	Cum. Rew. ($\uparrow$)	Cum. Rew (Larger) ($\uparrow$)
Random	96.8 ± 0.5	$5\ \%$	3.8 ± 8.9	3.7 ± 6.4
MAML	64.3 ± 39.3	45.2%	14.95 ± 4.5	5.8 ± 10.3
RL^2	16.2 ± 1.1	96.2%	77.7 ± 46.5	28.1 ± 29.7
MetODS	$\mathbf{14.7 \pm 1.4}$	$\mathbf{96.6\%}$	$\mathbf{86.5 \pm 46.8}$	$\mathbf{34.9 \pm 34.9}$

Figure 5: Performance of Meta-RL models tested at convergence ($1e^7$ env. steps). MetODS better explores the maze as measured by the average number of steps before 1st reward and the success rate in finding the reward at least once. It then better exploits the maze as per the accumulated reward. (* We assign 100 to episodes with no reward encounter.)

$\diamond$ **Ablation study and variations:** We explored the contribution of the different features combined in MetODS update rule, showing that they all contribute to the final performance of our meta-learning model. First, we tested the importance of the element-wise tuning of plasticity in weight-based learning models and note that while it adversely affects MAML gradient update, it greatly improves MetODS performance, suggesting different forms of weights update. Second, we verified that augmenting recursivity depth S was beneficial to performance, consistently with Harlow results. Third, we noted that transforming the rightmost vector of the writing equation in 5 with a linear projection (*Linear* in figure 4, see S.I for full experimental details) yields major improvement while non-linear (*MLP*) does not improve performance. Finally, we additionally test the capability of the learnt navigation skills to generalize to a larger maze size of 10×10 pix. unseen during training. We show that MetODS is able to retain its advantage (see figure 4 and table 5 for results).

5.3 Generality : Motor control

Finally, we test the generality of the reinforcement learning program learnt by our model for different continuous control tasks:

MetaWorld: First, we use the dexterous manipulation benchmark proposed in [81] using the benchmark suite [82], in which a Sawyer robot is tasked with diverse operations. A full adaptation episode consists in N=10 rollouts of 500 timesteps of the same task across which dynamic weights are carried over. Observation consists in the robot's joint angles and velocities, and the actions are its joint torques. We compare MetODS with baseline methods in terms of meta-training and meta-testing success rate for 3 settings, *push, reach* and *ML-10*. We show in Fig. 6 the meta-training results for all the methods in the MetaWorld environments. Due to computational resource constraints, we restrict our experiment to a budget of 10M steps per run. While we note that our benchmark does not reflect the final performance of previous methods reported in [81] at 300M steps, we note that MetODS test performance outperforms these methods early in training and keeps improving at 10M steps, potentially leaving room for improvement with additional training. (see S.I for additional discussion). Finally, we note that all tested approaches performed modestly on ML10 for test tasks, which highlights the limitation of current methods. We conjecture that this might be due to the absence of inductive bias for sharing knowledge between tasks or fostering systematic exploration of the environment of the tested meta-learning algorithms.

Robot impairment: We also tested the robustness of MetODS learnt reinforcement programs by evaluating the agent ability to perform in a setting not seen during training: specifically, when partially impairing the agent motor capabilities. We adopt the same experimental setting as section 5.1 for the *Ant* and *Cheetah* robots and evaluate the performance when freezing one of the robots

torque. We show that our model policy retains a better proportion of its performance compared to other approaches. These results suggest that fast synaptic dynamics are not only better suited to support fast adaptation of a motor policy in the continuous domain, but also implement a more robust reinforcement learning program when impairing the agent motor capabilities.

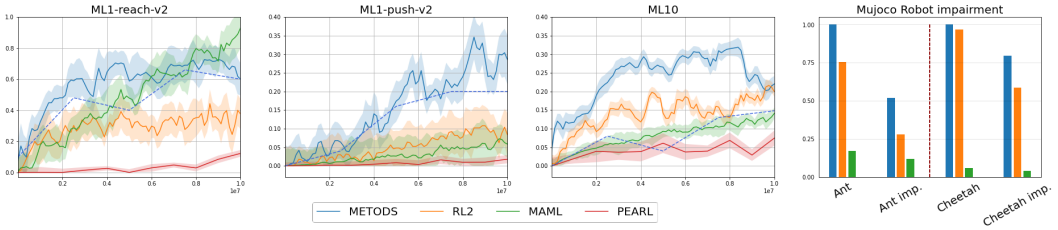

Figure 6: Left Meta-training results for MetaWorld benchmarks. Subplots show tasks success rate over training timesteps. Average meta-test results for MetODS is shown in dotted line. **Right** Cumulative reward of the *Ant* and *Cheetah* directional locomotion task. For each condition, results are normalized against the best performing policy

6 Discussion

In this work, we introduce a novel meta-RL system, MetODS, which leverages a self-referential weight update mechanism for rapid specialization at the episodic level. Our approach is generic and supports discrete and continuous domains, giving rise to a promising repertoire of skills such as one-shot adaptation, spatial navigation or motor coordination. MetODS demonstrates that locally tuned synaptic updates whose form depends directly on the network configuration can be meta-learnt for solving complex reinforcement learning tasks. We conjecture that further tuning the hyperparameters as well as combining MetODS with more sophisticated reinforcement learning techniques can boost its performance. Generally, the success of the approach provides evidence for the benefits of fast plasticity in artificial neural networks, and the exploration of self-referential networks.

7 Broader Impact

Our work explore the emergence of reinforcement-learning programs through fast synaptic plasticity. The proposed evidence that information memorization and manipulation as well as behavioral specialization can be supported by this computational principles 1) suggests that alternative paradigms to gradient descent might exists for efficient artificial neural network control and 2) might help question the functional role of such mechanisms observed in biology. Additionally, the proposed method is of interest for interactive machine learning systems that operates in quickly changing environments and under uncertainty (bayesian optimization, active learning and control theory). For instance, the meta RL approach proposed in this work could be applied to brain-computer interfaces for tuning controllers to rapidly drifting neural signals. Improving medical applications and robot control promises positive impact, however deeper theoretical understanding and careful deployment monitoring are required to avoid misuse.

8 Acknowledgments and Disclosure of Funding

This work was supported by ANR-3IA Artificial and Natural Intelligence Toulouse Institute (ANR-19-PI3A-0004), OSCI-DEEP ANR (ANR-19-NEUC-0004), ONR (N00014-19-1-2029), and NSF (IIS-1912280 and EAR-1925481). Additional support provided by the Carney Institute for Brain Science and the Center for Computation and Visualization (CCV) via NIH Office of the Director grant S10OD025181. The authors would like to thank the anonymous reviewers for their thorough comments and suggestions that led to an improved version of this work.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *ArXiv*, abs/1312.5602, 2013.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[3] Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

[4] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[5] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *ICLR (Poster)*, 2016.

[6] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017.

[7] Brenden Lake and Marco Baroni. Still not systematic after all these years: On the compositional skills of sequence-to-sequence recurrent networks, 2018.

[8] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning, 2020.

[9] Jürgen Schmidhuber, Jieyu Zhao, and Marco Wiering. Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. *Machine Learning*, 28, 01 1997.

[10] Sebastian Thrun. *Lifelong Learning Algorithms*, pages 181–209. Springer US, Boston, MA, 1998.

[11] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artif. Intell. Rev.*, 18(2):77–95, oct 2002.

[12] Jeff Clune. Ai-gas: Ai-generating algorithms, an alternate paradigm for producing general artificial intelligence. *CoRR*, abs/1905.10985, 2019.

[13] Matthew Botvinick, Sam Ritter, Jane X. Wang, Zeb Kurth-Nelson, Charles Blundell, and Demis Hassabis. Reinforcement learning, fast and slow. *Trends in Cognitive Sciences*, 23(5):408–422, 2019.

[14] S. J. Martin, P. D. Grimwood, and R. G. M. Morris. Synaptic plasticity and memory: An evaluation of the hypothesis. *Annual Review of Neuroscience*, 23(1):649–711, 2000.

[15] LF Abbott and Wade G Regehr. Synaptic computation. *Nature*, 431(7010):796–803, 2004.

[16] Wade G Regehr. Short-term presynaptic plasticity. *Cold Spring Harbor perspectives in biology*, 4(7):a005702, 2012.

[17] Natalia Caporale and Yang Dan. Spike timing–dependent plasticity: A hebbian learning rule. *Annual Review of Neuroscience*, 31(1):25–46, 2008. PMID: 18275283.

[18] Gianluigi Mongillo, Omri Barak, and Misha Tsodyks. Synaptic theory of working memory. *Science*, 319(5869):1543–1546, 2008.

[19] Nicolas Masse, Guangyu Yang, H. Song, Xiao-Jing Wang, and David Freedman. Circuit mechanisms for the maintenance and manipulation of information in working memory, 04 2018.

[20] Omri Barak and Misha Tsodyks. Working models of working memory. *Current Opinion in Neurobiology*, 25:20–24, 2014. Theoretical and computational neuroscience.

[21] Mark G. Stokes. 'activity-silent' working memory in prefrontal cortex: a dynamic coding framework. *Trends in Cognitive Sciences*, 19(7):394–405, 2015.

[22] Sanjay G. Manohar, Nahid Zokaei, Sean J. Fallon, Tim P. Vogels, and Masud Husain. Neural mechanisms of attending to items in working memory. *Neuroscience and Biobehavioral Reviews*, 101:1–12, 2019.

[23] Pierre Yger, Marcel Stimberg, and Romain Brette. Fast learning with weak synaptic plasticity. *Journal of Neuroscience*, 35(39):13351–13362, 2015.

[24] Răzvan V Florian. Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural computation*, 19(6):1468–1502, 2007.

[25] Geoffrey E Hinton and David C Plaut. Using fast weights to deblur old memories. In *Proceedings of the 9th Annual Conference of the Cognitive Science Society*, pages 177–186, 1987.

[26] Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

[27] Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4:131–139, 1992.

[28] Christoph von der Malsburg. *The Correlation Theory of Brain Function*, pages 95–119. Springer New York, New York, NY, 1994.

[29] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[30] Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. *Advances in Neural Information Processing Systems*, 29:4331–4339, 2016.

[31] Thomas Miconi, Kenneth Stanley, and Jeff Clune. Differentiable plasticity: training plastic neural networks with backpropagation. In *International Conference on Machine Learning*, pages 3559–3568. PMLR, 2018.

[32] Imanol Schlag and Jürgen Schmidhuber. Learning to reason with third-order tensor products. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10003–10014, 2018.

[33] Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. Going beyond linear transformers with recurrent fast weight programmers. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[34] Imanol Schlag, Tsendsuren Munkhdalai, and Jürgen Schmidhuber. Learning associative inference using fast weight memory. In *International Conference on Learning Representations*, 2020.

[35] Elad Sarafian, Shai Keynan, and Sarit Kraus. Recomposing the reinforcement learning building blocks with hypernetworks. In *International Conference on Machine Learning*, pages 9301–9312. PMLR, 2021.

[36] Luckeciano C Melo. Transformers are meta-reinforcement learners. In *International Conference on Machine Learning*, pages 15340–15359. PMLR, 2022.

[37] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[38] Shin Ishii, Wako Yoshida, and Junichiro Yoshimoto. Control of exploitation–exploration meta-parameter in reinforcement learning. *Neural networks*, 15(4-6):665–687, 2002.

[39] Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

[40] Pedro A Ortega, Jane X Wang, Mark Rowland, Tim Genewein, Zeb Kurth-Nelson, Razvan Pascanu, Nicolas Heess, Joel Veness, Alex Pritzel, Pablo Sprechmann, et al. Meta-learning of sequential strategies. *arXiv preprint arXiv:1905.03030*, 2019.

[41] Jan Humplik, Alexandre Galashov, Leonard Hasenclever, Pedro A. Ortega, Yee Whye Teh, and Nicolas Heess. Meta reinforcement learning as task inference. *CoRR*, abs/1905.06424, 2019.

[42] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *CoRR*, abs/1912.01603, 2019.

[43] Larry F Abbott and Sacha B Nelson. Synaptic plasticity: taming the beast. *Nature neuroscience*, 3(11):1178–1183, 2000.

[44] Tuning into diversity of homeostatic synaptic plasticity. *Neuropharmacology*, 78:31–37, 2014. Homeostatic Synaptic Plasticity.

[45] Chun Yun Chang. Impact of second messenger modulation on activity-dependent and basal properties of excitatory synapses. 2010.

[46] Rylan S. Larsen, Ikuko T. Smith, Jayalakshmi Miriyala, Ji Eun Han, Rebekah J. Corlew, Spencer L. Smith, and Benjamin D. Philpot. Synapse-specific control of experience-dependent plasticity by presynaptic nmda receptors. *Neuron*, 83(4):879–893, 2014.

[47] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, et al. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2020.

[48] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[49] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018.

[50] Rein Houthooft, Richard Y Chen, Phillip Isola, Bradly C Stadie, Filip Wolski, Jonathan Ho, and Pieter Abbeel. Evolved policy gradients. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5405–5414, 2018.

[51] Zhongwen Xu, Hado van Hasselt, and David Silver. Meta-gradient reinforcement learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2402–2413, 2018.

[52] Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. *Advances in Neural Information Processing Systems*, 31:5302–5311, 2018.

[53] Sepp Hochreiter, A Steven Younger, and Peter R Conwell. Learning to learn using gradient descent. In *International Conference on Artificial Neural Networks*, pages 87–94. Springer, 2001.

[54] Jane X. Wang, Zeb Kurth-Nelson, Dharshan Kumaran, Dhruva Tirumala, Hubert Soyer, Joel Z. Leibo, Demis Hassabis, and Matthew Botvinick. Prefrontal cortex as a meta-reinforcement learning system. 2018.

[55] Máté Lengyel and Peter Dayan. Hippocampal contributions to control: The third way. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

[56] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

[57] Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adria Puigdomenech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *International Conference on Machine Learning*, pages 2827–2836. PMLR, 2017.

[58] Samuel Ritter, Jane X Wang, Zeb Kurth-Nelson, and M Botvinick. Episodic control as meta-reinforcement learning. *bioRxiv*, page 360537, 2018.

[59] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1842–1850. JMLR.org, 2016.

[60] Sachin Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017.

[61] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 06–11 Aug 2017.

[62] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International Conference on Machine Learning*, pages 2554–2563. PMLR, 2017.

[63] Thomas Miconi, Aditya Rawal, Jeff Clune, and Kenneth O Stanley. Backpropamine: training self-modifying neural networks with differentiable neuromodulated plasticity. In *International Conference on Learning Representations*, 2018.

[64] Elias Najarro and Sebastian Risi. Meta-learning through hebbian plasticity in random networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20719–20731. Curran Associates, Inc., 2020.

[65] Thomas Miconi. Learning to learn with backpropagation of hebbian plasticity. *arXiv: Neural and Evolutionary Computing*, 2016.

[66] Krzysztof Choromanski, Jared Davis, Valerii Likhosherstov, Xingyou Song, Jean-Jacques E. Slotine, Jacob Varley, Honglak Lee, Adrian Weller, and Vikas Sindhwani. An ode to an ode. *ArXiv*, abs/2006.11421, 2020.

[67] Mathieu Chalvidal, Matthew Ricci, Rufin VanRullen, and Thomas Serre. Go with the flow: Adaptive control for neural ODEs. In *International Conference on Learning Representations*, 2021.

[68] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[69] Wojciech Zaremba and Ilya Sutskever. Reinforcement learning neural turing machines - revised, 2016.

[70] Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2827–2836. PMLR, 06–11 Aug 2017.

[71] Sergey Bartunov, Jack Rae, Simon Osindero, and Timothy Lillicrap. Meta-learning deep energy-based memory models. In *International Conference on Learning Representations*, 2019.

[72] Wei Zhang and Bowen Zhou. Learning to update auto-associative memory in recurrent neural networks for improving sequence memorization, 2017.

[73] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[74] Pascal Koiran. Dynamics of discrete time, continuous state hopfield networks. *Neural Computation*, 6(3):459–468, 1994.

[75] Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 02 2017.

[76] Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. In *NIPS*, 2016.

[77] Imanol Schlag and Jürgen Schmidhuber. Learning to reason with third order tensor products. In *NeurIPS*, pages 10003–10014, 2018.

[78] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

[79] Harry Frederick Harlow. The formation of learning sets. *Psychological review*, 56 1:51–65, 1949.

[80] Robert Clay Prim. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957.

[81] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

[82] The garage contributors. Garage: A toolkit for reproducible reinforcement learning research. `https://github.com/rlworkgroup/garage`, 2019.

[83] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[84] Michael Betancourt, Charles C Margossian, and Vianey Leos-Barajas. The discrete adjoint method: Efficient derivatives for functions of discrete sequences. *arXiv preprint arXiv:2002.00326*, 2020.

[85] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[86] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[87] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018.

Supplementary Information : Meta-Reinforcement Learning with Self-Modifying Networks

9 Optimization

Defining the weight parameters $\boldsymbol{W}$ of MetODS as dynamic variables lifts the optimization problem (2) into a functional space of control functions parameterized by $\boldsymbol{\theta}$. Hence, meta-optimizing the control necessitates the estimation of gradients with respect to $\boldsymbol{\theta}$ over the space $\mathbb{T}$ and for any possible trajectory π_t in Π. Interestingly, previous meta RL approaches have performed policy gradient optimization by sampling a single policy trajectory $\pi_t \sim \mathcal{M}_{\boldsymbol{\theta}}(\tau)$ over M multiple tasks, showing that it is sufficient to obtain correct gradient estimates on $\boldsymbol{\theta}$. We proceed in the same way, by estimating the gradient policy update integrated over the space of tasks as mini-batches over tasks.

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_{\tau \sim \mu_{\mathbb{T}}} \left[\mathbb{E}_{\pi \sim \mu_{\pi}^{\boldsymbol{\theta},\tau,t}} \left[\mathcal{R}(\tau, \pi) \right] \right] \quad \approx \sum_{\tau_1,\ldots,\tau_n} \sum_{t=0}^{T} \frac{\partial \log \pi_t(\boldsymbol{a}_t | \boldsymbol{W}_t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} r_{\tau_i}(\boldsymbol{a}_t, \boldsymbol{s}_t) \tag{7}$$

Additionally, the memory cost of storing synaptic weights trajectories instead of hidden activity in a network of N neurons is $\mathcal{O}(N^2)$ instead of $\mathcal{O}(N)$. This might lead to prohibitively large memory requirements for training with BPTT [83] over long episodes. We present in S.I an alternative solution to train the model through the discrete adjoint sensitivity method, leveraging the work of [84] yielding a memory cost of $\mathcal{O}(1)$. The agent's log-policy total derivative with respect to $\boldsymbol{\theta}$ can be computed as the solution of an augmented adjoint problem [67].

9.1 Gradient policy update

We define the evolution of the agent policy in task τ up to step T as the stochastic policy process $(\pi_t)_{t \leq T}$ in the space of policy Π with measure $\mu_{\pi}^{\boldsymbol{\theta},\tau,1\cdots T}$ and write $(\boldsymbol{W}_t)_{t \leq T}$ the trajectory of weights, such that:

$$\big(\pi_t\big)_{t \leq T} = \big(\pi_t(\cdot | \boldsymbol{W}_t, \boldsymbol{s}_t, \boldsymbol{\theta})\big)_{t \leq T} \sim \mu_{\pi}^{\boldsymbol{\theta},\tau,1\cdots T} \tag{8}$$

We also recall the definition of $\mathcal{R}(\tau, \pi)$ as the average accumulated reward under the realisation of $(\pi_t)_{t \leq T}$ for task τ. Then, The policy gradient update used to train MetODS can be written as the average gradient of $\boldsymbol{\theta}$ with respect to $\mu_{\mathbb{T}}$ and $\mu_{\pi}^{\boldsymbol{\theta},\tau,1\cdots T}$:

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_{\tau \sim \mu_{\mathbb{T}}} \left[\mathbb{E}_{\pi \sim \mu_{\pi}^{\boldsymbol{\theta},\tau,1\cdots T}} \left[\mathcal{R}(\tau, \pi) \right] \right] = \mathbb{E}_{\tau \sim \mu_{\mathbb{T}}} \left[\sum_{t=0}^{T} \frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_{\pi_t, \mathcal{P}_\tau} \left[r_\tau(\boldsymbol{a}_t, \boldsymbol{s}_t) \right] \right] \tag{9}$$

Then for any $t \leq T$, we can rewrite the gradient of the average, using the log-policy trick:

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_{\pi_t, \mathcal{P}_\tau} \left[r_\tau(\boldsymbol{a}_t, \boldsymbol{s}_t) \right] = \mathbb{E}_{\pi_t, \mathcal{P}_\tau} \left[r_\tau(\boldsymbol{a}_t, \boldsymbol{s}_t) \frac{\partial \log \pi_t(\boldsymbol{a}_t^\tau | \boldsymbol{W}_t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right] \tag{10}$$

As specified in section 3.3, we do not sample over the policy distribution π_t and probability transition $\mathcal{P}_\tau$ to estimate the inner expectation in (11). Instead, we rely on a single evaluation, which yield, combining (9) with (10):

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_{\tau \sim \mu_{\mathbb{T}}} \left[\mathbb{E}_{\pi \sim \mu_{\pi}^{\boldsymbol{\theta},\tau,1\cdots T}} \left[\mathcal{R}(\tau, \pi) \right] \right] \quad \approx \quad \sum_{t=0}^{T} \left[r_\tau(\boldsymbol{a}_t, \boldsymbol{s}_t) \frac{\partial \log \pi_t(\boldsymbol{a}_t^\tau | \boldsymbol{W}_t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right] \tag{11}$$

By sampling over tasks, this last equation allows us to write the following gradient estimator

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_{\tau \sim \mu_{\mathbb{T}}} \left[\mathbb{E}_{\pi \sim \mu_{\pi}^{\boldsymbol{\theta},\tau,1\cdots T}} \left[\mathcal{R}(\tau, \pi) \right] \right] \quad \approx \quad \frac{1}{M} \sum_{i=0}^{M} \sum_{t=0}^{T} \frac{\partial \log \pi_t(\boldsymbol{a}_t^{\tau_i} | \boldsymbol{W}_t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} r(\boldsymbol{a}_t^{\tau_i}, \boldsymbol{s}_t^{\tau_i}) \tag{12}$$

9.2 Discrete adjoint system

With the previous notation, let us define the total gradient function $\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}$ as the gradient of our objective function:

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}) = \frac{1}{M} \sum_{t=0}^{T} \sum_{i=0}^{M} \frac{\partial \log \pi_t(\boldsymbol{a}_t^{\tau_i} | \boldsymbol{W}_t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} r(\boldsymbol{a}_t^{\tau_i}, \boldsymbol{s}_t^{\tau_i}) \tag{13}$$

To clarify, we shall introduce the intermediary cost notation:

$$c_t(\boldsymbol{W}, \boldsymbol{\theta}, t) = \sum_{i=0}^{M} \log \pi_t(\boldsymbol{a}_t^{\tau_i} | \boldsymbol{W}_t, \boldsymbol{\theta}) r(\boldsymbol{a}_t^{\tau_i}, \boldsymbol{s}_t^{\tau_i}) \tag{14}$$

and we note the following update equation

$$\boldsymbol{W}_{t+1} = \delta(\boldsymbol{W}_t, \boldsymbol{\theta}) \tag{15}$$

This identify a discrete dynamical system with finite sum and differentiable cost, whose gradient can be computed mediating the introduction of an *adjoint* dynamical system presented in section 2 of [84] . Defining $(\boldsymbol{\nu}_t)_{t \leq T}$ the adjoint sequence, the general gradient equation can be computed as:

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}) = \left[\frac{\partial c_0}{\partial \boldsymbol{W}_0} - \boldsymbol{\nu}_0 \cdot \frac{\partial \delta}{\partial \boldsymbol{W}_0} - \boldsymbol{\nu}_0 \right]^{\dagger} \frac{\partial \boldsymbol{W}_0}{\partial \boldsymbol{\theta}} + \sum_{i=1}^{T} \left[\frac{\partial c_t}{\partial \boldsymbol{\theta}} - \boldsymbol{\nu}_t^{\dagger} \cdot \frac{\partial \delta}{\partial \boldsymbol{\theta}} \right] \tag{16}$$

Applying formula (16) to MetODS yields the following gradient formula:

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}) = \left[\frac{\partial c_0}{\partial \boldsymbol{W}_0} - \boldsymbol{\nu}_0 \right] + \sum_{i=1}^{T} \left[\frac{\partial c_t}{\partial \boldsymbol{\theta}} - \boldsymbol{\nu}_t^{\dagger} \cdot \frac{\partial \delta}{\partial \boldsymbol{\theta}} \right] \tag{17}$$

where $(\boldsymbol{\nu}_t)_{t \leq T}$ follows the following update rule backwards:

$$\begin{cases} \boldsymbol{\nu}_T = 0 \\ \boldsymbol{\nu}_{t-1} = \boldsymbol{\nu}_t - \frac{\partial c_t}{\partial \boldsymbol{W}_t} \end{cases} \tag{18}$$

10 Experiment details

General information: As specified in section **3** and **5**, we test a single model definition for all experiments in this work, with one layer of dynamic weights $\boldsymbol{W}_t$. This layer consists in a dense matrix of size $\boldsymbol{n} \in \mathbb{N}$ with learnt or random initialization. Our model lightweight parametrization of the synaptic update rule makes it a very parameter efficient technique, which can perform batch-computation and be ported to GPU hardware to accelerate training. We refer readers to table 1 for specific details of each experiment presented in section 5. In addition to this *pdf* file, we provide the code for training METODS agents in experiments presented in section 5 at https:/

Feature - Experiment	Harlow	Gym Mujoco	Maze	Meta-World
Dynamic layer size $\boldsymbol{n}$	20	200	200	100
Input size $\boldsymbol{i}$	12	134 (Ant) / 27 (Cheetah)	15	1
Output size $\boldsymbol{o}$	2	8 (Ant)/ 6 (Cheetah)	4	8
Embedding $\boldsymbol{f}$	$[\boldsymbol{i} \times 32, \sigma, 32 \times \boldsymbol{n}]$	$[\boldsymbol{i} \times 64, \sigma, 64 \times \boldsymbol{n}]$	$[\boldsymbol{i} \times 32, \sigma, 32 \times \boldsymbol{n}]$	$[\boldsymbol{i} \times 32, \sigma, 32 \times 64, \sigma, 64 \times \boldsymbol{n}]$
Read-out $\boldsymbol{g}$	$[\boldsymbol{n} \times 32, \sigma, 32 \times \boldsymbol{o}]$	$[\boldsymbol{n} \times 64, \sigma, 64 \times \boldsymbol{o}]$	$[\boldsymbol{n} \times 32, \sigma, 32 \times \boldsymbol{o}]$	$[\boldsymbol{n} \times 64, \sigma, 64 \times \boldsymbol{o}]$
Non-linearities	tanh	tanh	tanh	tanh
Init. $\boldsymbol{W}_0$	$\mathcal{N}(0, 1e-3)$	Learned	Learned	Learned
# Num. of episodes	5 (max)	1	1	10
Lenght of 1 episode	250 (max)	200	100	500
Meta-training Alg.	A2C	A2C	A2C	PPO
learning rate	7.5e-4	5e-4	5e-4	5e-4
meta-batch-size	50	50	20	25
discount factor λ	9e-1	9e-1	9.9e-1	9.9e-1
GAE	1.	9.5E-1	9.5e-1	9.5e-1
value function coeff.	4e-1	1e-1	4e-1	-
entropy reg. factor	3e-2	3e-2	1e-2	1e-2

Table 1: Summary of training hyper-parameters for the four experiments presented in this work.

Meta-training algorithm: We show in our experiments, that the all meta-parameters $\boldsymbol{\theta} = [\boldsymbol{\alpha}, \boldsymbol{\kappa}, \boldsymbol{\beta}, \boldsymbol{f}, \boldsymbol{g},]$ can be jointly optimized with two policy gradient algortihms. We use policy gradient methods to meta-train the synaptic parameters: Advantage Actor-critic algorithm (A2C) [85] where we consider that temporal dependancies are crucial to solve the task, and we use PPO [86] as a sequential policy optimization over fixed rollouts for the motor control experiment. Additionally, to reduce noise in policy gradient updates, we further show that it is possible to learn a dynamic advantage estimate of the Generalized Advantage Estimation (GAE) [87] and that we can meta-learn it as a second head of the MetODS layer output $\boldsymbol{v}^{(s)}$. This fact confirms that tuned hebbian-updates are also a sufficient mechanism to keep track of a policy value estimate.

Plasticity parameters: In all our experiments, $\boldsymbol{\alpha}$ consists in a real-valued matrix of $\mathbb{R}^{\boldsymbol{n}\times\boldsymbol{n}}$ initialized with independent normal distribution $\mathcal{N}(\mu = 0, \sigma = 1e-3)$. Similarly, multi-step weighting parameters $(\boldsymbol{\kappa}_s^{(l)})_{l\leq s}$ and $(\boldsymbol{\beta}_s^{(l)})_{l\leq s}$ can be stored as entries of triangular inferior matrices of $\mathbb{R}^{\boldsymbol{S}\times\boldsymbol{S}}$ and are initialized with $\mathcal{N}(\mu = 0, \sigma = 1e-2)$.

Embedding and read-out: At each time-steps, inputs to the feed-forward embedding function $\boldsymbol{f}$ consist in a concatenation of new observable/state $\boldsymbol{s}_t$ as well as previous action and reward $\boldsymbol{a}_{t-1}$ and $\boldsymbol{r}_{t-1}$. After adaptation procedure, policy is read-out from the last activation vector $\boldsymbol{v}^{(S)}$ by a feed-forward function $\boldsymbol{g}$ which output statistics of a parameterized distribution in the action space (categorical with Softmax normalization in the discrete case or Gaussian in the continuous case). Both input and output mappings $\boldsymbol{f}$ and $\boldsymbol{g}$ consist in 2-layer Perceptrons with dense connections initialized with orthogonal initialization and hyperbolic tangent non-linearities.

10.1 Harlow task

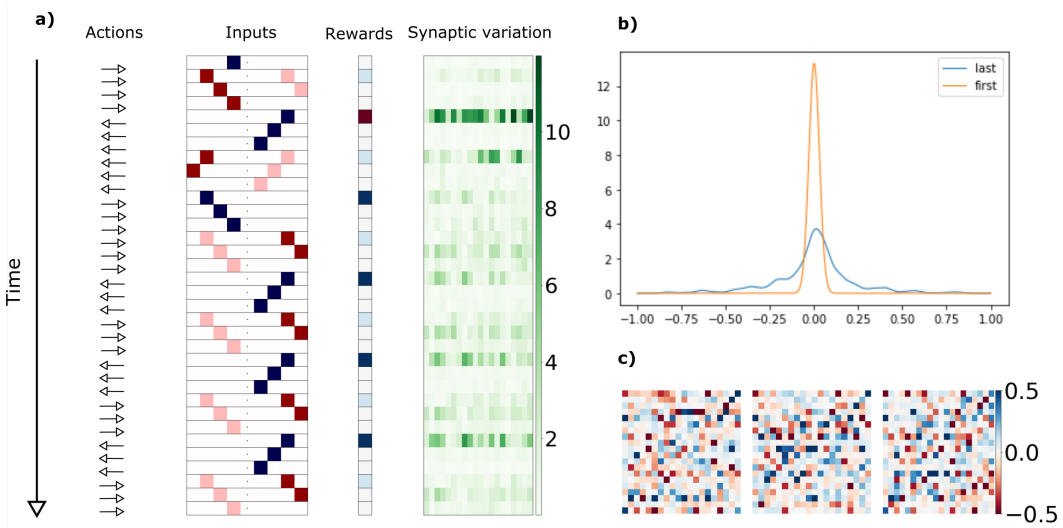

Figure 7: Harlow task: a) An episode of the Harlow task. From left to right: Actions, visual field of the agent (blue squares:fixation target, red squares: values), Rewards (red;negative,light blue: fixation reward, dark blue:value reward) and sum of absolute synaptic variation $\|\sum \Delta \boldsymbol{W}\|$ per neuron. **b)** Distribution of weights before and after adaptation over 5 presentations.**c)** Three instances of the dynamic weights $\boldsymbol{W}_t$ after adaptation that solved the Harlow task. Every synaptic configuration presents differences but perform optimally.

This experiment consists in a 1-dimensional simplification of the task presented in [54] and inspired from https://github.com/bkhmsi/Meta-RL-Harlow. The action space is the discrete set $\{-1, 1\}$ moving the agent respectively to the left/right on a discretized line. The state space consist in 17 positions while the agent receptive field is eight dimensional. Values are placed at 3 positions from the fixation target position. The fixation position yields a reward of 0.2 while the values are drawn uniformly from $[\![0, 10]\!]$ and randomly associated with a reward of -1 and 1 at the beginning of each episode. The maximal duration for an episode is 250 steps, although the model solves the 5 trials in ~ 35 steps on average.

10.2 Gym Mujoco directional robot control

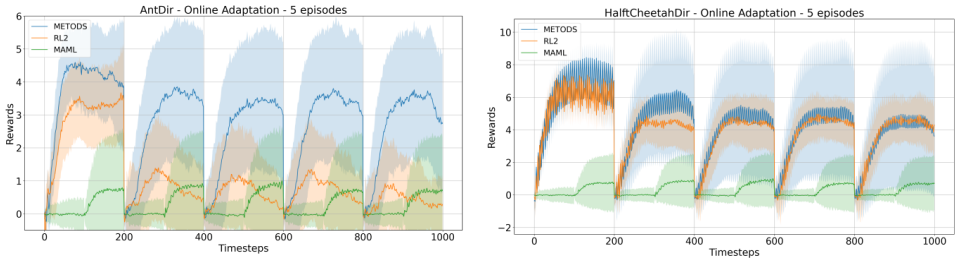

Figure 8: Reward profiles of MAML, RL^2 and MetODS over 5 consecutive episodes with randomly varying rewarded direction. In *Cheetah*, MetODS slightly overperforms RL^2 and in *Ant*, MetODS is the only approach able to reach performance comparable to first episode in subsequent episodes.

Setting: We consider the directional rewards task proposed in [61] with the *Ant* and *Cheetah* robots, as a more complex test of rapid adaptation. We apply standard RL practice for continuous control by parameterizing the stochastic policy as a product of independent normal distributions with fixed standard deviation $\sigma = 0.1$ and mean inferred by the agent network. A training meta-episode consists in a single rollout of 200 steps with random sampling of the reward direction for each episode.

Robot impairment: We partially impaired the agent motor capabilities by "freezing" one of the robot actuator. Namely we consistently passed a value of zero to the the *right_back_leg* in Ant (coordinate 8 in OpenAI Gym XML asset files) and *ffoot* in Cheetah (coordinate 6 in OpenAI Gym XML asset files).

Continual adaptation: Finally, we further investigated whether the adaptation mechanism was also adjustable after performing initial adaptation by testing trained agents over 5 consecutive episodes with randomly changing rewarded direction while retaining weight state across episodes. We show in Fig 10 the average reward profile of the three meta-RL agents over 1000 test episodes, where MetODS adapt faster and better with respect to reward variations.

10.3 Maze Navigation task

Maze generation: The maze environments are created following Prim's algorithm [80] which randomly propagates walls on a board of $N \times N$ cells. We additionally add walls to cell location were no walls has been created at any of the neighboring 8 cells to avoid null inputs to the agent. The reward and agent locations are selected at random at the beginning of an episode. The reward location does not change during the episode while the agent is restarting from a uniform random location after every encounter with reward.

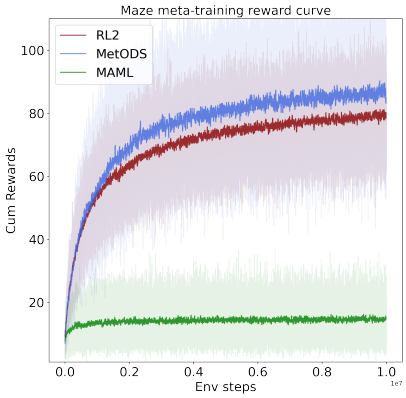

Figure 9: Meta-training rewards curves of the maze experiment presented in section 5.2

Models capacity: In this experiment, since model memory capacity seems to directly impact final performance, we specifically control for the number of learnable parameters in order to compare models and fix a training budget of 10M env. steps. Since MAML does not have a continual adaptation mechanism, we perform gradient adaptation every 20 time-steps during an episode, in order to balance noisiness of the gradient and rapidity of exploration. We show in Table 2 the mean over 1000 episodes of the accumulated reward of agents trained on 8×8 mazes and tested on different maze sizes $N \in (4, 6, 8, 10, 12)$. Agents performances are highly varying within a size setting due to differences in maze configurations and across maze sizes due to increasing complexity. However, MetODS and RL^2 are able to generalize at least partially to these new settings (see table 2). MetODS outperforms RL^2 in every setting, and generalizes better, retaining an advantage of up to 25% in accumulated reward in the biggest maze.

MAZE SIZE	MAML	RL^2	METODS	REL IMP. TO RL^2
6	21.0 ± 18.4	149.3 ± 66.7	$\mathbf{169.1 \pm 66.1}$	13%
8	14.9 ± 4.5	72.1 ± 45.6	$\mathbf{87.3 \pm 48.3}$	20%
10	5.7 ± 7.9	28.1 ± 29.7	$\mathbf{34.9 \pm 34.9}$	21%
12	3.9 ± 6.9	11.1 ± 15.8	$\mathbf{13.9 \pm 19.8}$	25%

Table 2: Average accumulated reward over 1000 episodes for different maze sizes for agents trained for N=8.

Additionally, we investigated the selectivity of neurons of MetODS plastic weight layer with respect to spatial location, to investigate whether there could be emergence of activation patterns resembling those of grid cells found in the mammalian entorhinal cortex. We measured the selectivity of each neuron for specific agent locations by measuring the average activation rate normalized by the number of agent passages in different maze configurations. Interestingly, we found that without any explicit regularization, some neurons displayed sparse activation and consistent remapping between maps.

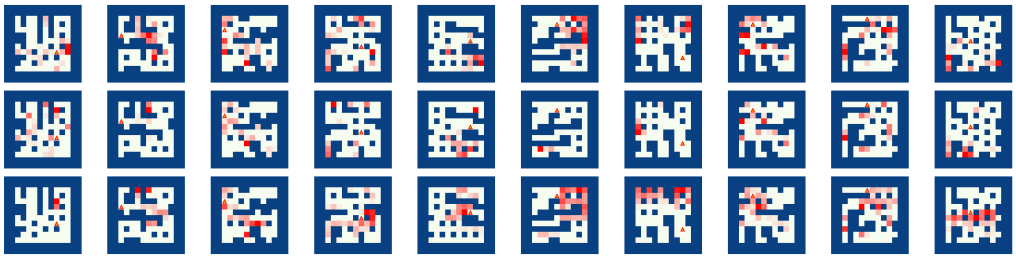

Figure 10: Heatmap of activation rate of three neurons of a trained model across 10 different maze maps normalized by agent passage. Activation patterns show selectivity to particular positions within an episode and strong remapping across maze configurations.

10.4 Meta-world

Experiments: In order to run experiments and baseline of MetaWorld [81], we used the training routines and functions offered in the python library GARAGE [82]. We ran two different types of experiments:

1. **ML1**: In this setting, we restrict the task distribution to a single type of robotic manipulation while varying the goal location. The meta-training "tasks" corresponds to 50 random initial agent and goal positions, and meta-testing to 50 heldout positions. We tested our model on the *reach-v2* and *push-v2* tasks.
2. **ML10**: This set-up tests generalization to new manipulation tasks, the benchmark provides 10 training tasks and holds out 5 meta-testing tasks.

Training details: We used PPO [86] for training our model, with clip range $r = 0.2$ and 10 inner optimization steps maximum, for its good empirical performance and in order to compare with other methods with recurrent computation scheme. We kept the experimental settings from the original paper with N=10 episodes of 500 time-steps and sampled proximal gradient update in a meta-batch size of 25 trajectories. To estimate value function, we both tested a feedforward neural network trained at each iteration of the meta-training or an additional MetODS network, which resulted in similar training results.