Mitigating Catastrophic Forgetting in Continual RL via Certified Alignment

Anonymous Authors[1]

Abstract

Continual adaptation enables reinforcement learning (RL) agents to learn new capabilities, sometimes at the cost of forgetting behaviour in previous tasks. We propose CERTALIGN, a certified alignment framework that constrains downstream policy updates to a parameter subspace in which any policy provably retains the source-task behaviour. The method uses a sound differentiable surrogate, enabling projected gradient adaptation with provable alignment guarantees. Experiments in Frozen Lake and Lunar Lander environments show that CERTALIGN is the only adaptation method that preserves source-task behaviour both provably and empirically, while retaining non-trivial downstream plasticity.

1. Introduction

While standard adaptation techniques allow AI systems to acquire skills for solving new tasks, they can also erase behaviours learned for completing previous tasks. In continual reinforcement learning (RL), this phenomenon is commonly known as *catastrophic forgetting* (Khetarpal et al., 2022). Existing approaches mitigate forgetting through regularisation (Kirkpatrick et al., 2017), replay (Rolnick et al., 2019), gradient constraints (Lopez-Paz & Ranzato, 2017), or architectural expansion (Rusu et al., 2016). In contrast, our contribution is aimed at *provably preserving* previously learned capabilities through the lens of *certified alignment*. We consider RL settings in which a source-trained NoAdapt policy is available and can be queried in the source task to generate demonstrations of the behaviour to retain. For this case, we develop a method that *guarantees* that the model remains aligned to that demonstrated behaviour during adaptation to a new (downstream) task. We make three contributions in our work:

1. Formulate the source-task behaviour alignment problem using a *specification function* over a behaviour demonstration dataset generated from the policy trained on a source task.

2. Introduce a *sound constraint* on a differentiable specification surrogate, which allows optimisation tools to build a *certified parameter space* in which alignment is guaranteed.

3. Apply certified alignment to continual RL and compare it to unconstrained adaptation and regularisation-based methods. Our experiments demonstrate that CERTALIGN is the only adaptation method that is *provably aligned* with the source-task behaviour while allowing sufficient plasticity for downstream adaptation.

2. Certified Alignment Methodology

We consider a setting where an RL policy π_θ parametrised with θ is first trained on the source task and then adapted to the downstream task. Each task is modelled as a Markov Decision Process (MDP). To make the task-incremental setting explicit (van de Ven et al., 2022), we use an augmented MDP view. For each task $q \in \{\text{source}, \text{down}\}$, let $\mathcal{M}_q = (\mathcal{S}, \mathcal{A}, P_q, r_q, \gamma, \mu_{0,q})$. The policy observes an augmented state $\bar{s} = (s, q) \in \mathcal{S} \times \{\text{source}, \text{down}\}$, and the augmented transition keeps the task identifier fixed: $\bar{P}((s', q') \mid (s, q), a) = P_q(s' \mid s, a)\mathbf{1}\{q' = q\}$. Thus, the task ID defines which task the agent is solving. Reward functions r_{source} and r_{down} define source training and downstream adaptation objectives, whereas the alignment specification Φ is a separate specification function used to constrain forgetting.

A policy π_θ can be evaluated on an MDP $\mathcal{M}$ for a property such as performance, safety or action agreement using a specification function $\Phi(\mathcal{M}, \pi_\theta)$, which maps $\mathcal{M}$ and π_θ into a real scalar. In practice, computing $\Phi(\mathcal{M}, \pi_\theta)$ may not be possible because it may require the full transition dynamics, initial-state distribution, or reward information. Instead, we estimate the source-behaviour property from a behaviour demonstration dataset collected by greedily executing the provided policy π_{θ_0} in the source environment:

$$D = \left\{ \left(s_t^{(i)}, a_t^{(i)} \right) : 1 \le i \le K,\ 0 \le t < T_i \right\}. \quad (1)$$

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the ICML 2026 Workshop "Continual Adaptation at Scale: Towards Sustainable AI". Do not distribute.

Here, $s_t^{(i)} \in \mathbb{R}^n$ is the augmented state and $a_t^{(i)}$ is the policy action at time step t in trajectory i. Since π_{θ_0} and the source environment can be queried, D_{source} need not be prepared in advance and can be enlarged until it covers the desired source behaviour. In deterministic greedy settings, a single policy rollout in the source task may be sufficient. We assume that each demonstrated state has one desired action, avoiding ambiguity regarding which behaviour to preserve. We denote the specification estimate by $\widehat{\Phi}(D;\pi_\theta)$. We refer the reader to Appendix A and Sutton & Barto (2018) for more details on MDPs and RL.

The ideal constrained adaptation problem is

$$\max_\theta \; J_{\text{down}}(\theta) \quad \text{s.t.} \quad \Phi_{\text{acc}}(\mathcal{M}_{\text{source}}, \pi_\theta) = 1, \tag{2}$$

where J_{down} is the downstream return objective and Φ_{acc} is the source-task behaviour-alignment specification. Because the exact source-MDP specification may be unavailable, we solve a certified empirical version:

$$\max_\theta \; J_{\text{down}}(\theta) \quad \text{s.t.} \quad \widehat{\Phi}_{\text{acc}}(D_{\text{source}}, \pi_\theta) = 1. \tag{3}$$

Our method constructs a locally invariant domain (LID) (Elmecker-Plakolm et al., 2025) around a "target" policy, i.e. the policy whose behaviour is to be preserved. This LID computation is done such that every parameter vector inside the LID satisfies the empirical alignment constraint. Downstream updates are then projected into this certified region. We first present the algorithm (§2.1), describe each phase (§2.2–§2.4), and then state the assumptions and theoretical guarantees (§2.5).

2.1. Algorithm Overview

Algorithm 1 gives the pseudocode for CERTALIGN. It proceeds in three phases: (1) calibrate the sound surrogate constraint; (2) compute the maximal locally invariant domain (LID); and (3) adapt to the downstream task with projected gradient descent (PGD). A *NoAdapt* neural policy π_{θ_0}, which was trained on a source task, is assumed to be provided. The source-task behaviour demonstration dataset D_{source} is generated from this policy before certification, and the goal is to preserve that demonstrated behaviour while adapting the policy to a downstream task.

2.2. Phase 1: Sound Surrogate Constraint Calibration

2.2.1. Alignment Specification

Definition 2.1 (Behaviour Alignment Specification)**.** For a state-action pair (s, a), where a is the desired action, we define the *pointwise behaviour-alignment specification* as

$$\phi_{\text{acc}}(s, a; \theta) := 1\left\{ \arg\max_{a' \in \mathcal{A}(s)} \pi_\theta(a' \mid s) = a \right\}. \tag{4}$$

Algorithm 1 CERTALIGN: Continual RL Adaptation with Certified Alignment

Input: NoAdapt policy π_{θ_0}, source MDP $\mathcal{M}_{\text{source}}$, downstream MDP $\mathcal{M}_{\text{down}}$, downstream objective (total return) J_{down}, settings for rollouts, RL, MaxLID, and PGD
Output: adapted policy π_{θ_L} aligned with D_{source}
Preliminary step: Collect D_{source} by greedy rollouts of π_{θ_0} in $\mathcal{M}_{\text{source}}$.
Phase 1: Sound surrogate constraint calibration
Choose δ such that $\widetilde{\Phi}_{\text{acc},\tau}(D_{\text{source}};\theta) \geq \delta \Rightarrow \widehat{\Phi}_{\text{acc}}(D_{\text{source}};\theta) = 1$ for all θ.
$\tau_{\max} \leftarrow \sup\{\tau > 0 : \widetilde{\Phi}_{\text{acc},\tau}(D_{\text{source}};\theta_0) \geq \delta\}$.
Phase 2: Maximal LID computation
$\mathcal{C}_{\delta,\tau_{\max}} \leftarrow \{\theta : \widetilde{\Phi}_{\text{acc},\tau_{\max}}(D_{\text{source}};\theta) \geq \delta\}$.
$\Theta^{\text{LID}}_{\text{source}} \leftarrow \text{MaxLID}(\theta_0, \mathcal{C}_{\delta,\tau_{\max}})$.
Phase 3: Downstream adaptation with PGD
Initialise $\theta \leftarrow \theta_0$ and retrieve L from the RL method settings.
for $l = 0$ **to** $L-1$ **do**
  $\theta' \leftarrow \theta + \eta \nabla_\theta J_{\text{down}}(\theta)$
  $\theta \leftarrow \Pi_{\Theta^{\text{LID}}_{\text{source}}}(\theta')$
end for
Return π_{θ_L}, where $\theta_L \leftarrow \theta$.

Thus, $\phi_{\text{acc}}(s, a; \theta) = 1$ if π_θ is aligned in state s, and $\phi_{\text{acc}}(s, a; \theta) = 0$ otherwise. Essentially, it measures alignment "accuracy". We define the corresponding dataset-level specification estimate as the minimum value of $\phi_{\text{acc}}(s, a; \theta)$ for the policy π_θ across behaviour demonstrations in a dataset D:

$$\widehat{\Phi}_{\text{acc}}(D, \theta) = \min_{(s,a)\in D} \phi_{\text{acc}}(s, a; \theta), \tag{5}$$

where $\widehat{\Phi}_{\text{acc}}(D, \theta)$ is a shorthand for $\widehat{\Phi}_{\text{acc}}(D, \pi_\theta)$.

Definition 2.2 (Policy Update with Behaviour Alignment)**.** Let $\mathcal{U}$ be an update mechanism that takes a parameter vector θ and a downstream-task MDP $\mathcal{M}_{\text{down}}$, and returns a parameter update. Define

$$\theta^+ := \theta + \mathcal{U}(\theta, \mathcal{M}_{\text{down}}). \tag{6}$$

We say that $\mathcal{U}$ is behaviour-aligned with respect to source-task behaviour demonstration dataset D_{source} if

$$\widehat{\Phi}_{\text{acc}}(D_{\text{source}}; \theta^+) = 1. \tag{7}$$

2.2.2. Sound Surrogate Constraint

The specification estimate $\widehat{\Phi}_{\text{acc}}(D, \theta)$ is not directly suitable for gradient-based optimisation because it contains both an $\arg\max$ and an indicator. We therefore introduce a differentiable surrogate together with a sound constraint whose satisfaction implies perfect behaviour alignment. Let $z_{\theta,a}(s)$ denote the logit assigned to action a in state s by the policy π_θ. Recall that the softmax function $\text{softmax}_\tau(z_{\theta,a}(s))$, with the temperature $\tau > 0$, is a smooth approximation of the indicator that action a has the largest logit among all

actions in state s. The *dataset-level surrogate* $\widetilde{\Phi}_{\text{acc}}(D,\theta)$ is then defined by taking the minimum across the demonstration dataset:

$$\widetilde{\Phi}_{\text{acc}}(D,\theta) := \min_{(s,a)\in D} \text{softmax}_\tau\left(z_{\theta,a}(s)\right). \tag{8}$$

Although the minimum is not smooth, it is subdifferentiable, and can therefore be used in modern automatic differentiation frameworks.

Theorem 2.3 (Sound Surrogate Constraint)**.** *Assume there are no ties between actions in the logit space. Then,* $\forall \tau > 0$,

$$\widetilde{\Phi}_{\text{acc}}(D,\theta) \geq 0.5 \implies \widehat{\Phi}_{\text{acc}}(D,\theta) = 1. \tag{9}$$

Theorem 2.3 provides a sound constraint on the surrogate, i.e., $\widetilde{\Phi}_{\text{acc}}(D,\theta) > 0.5$, which guarantees that the policy remains aligned to the desired source-task behaviour. Here, "sound" means that satisfying the surrogate constraint has no false positives for the hard action-agreement specification; the converse need not hold. We provide a proof in Section B.

2.3. Phase 2: Maximal LID Computation

Let us define the locally invariant domain (LID) (Elmecker-Plakolm et al., 2025) as follows:

Definition 2.4 (Locally Invariant Domain)**.** Given a policy π_θ and a specification Φ, a parameter subspace $\Theta^{\text{LID}} \in \mathbb{R}^p$ is a *locally κ-invariant domain* for an MDP $\mathcal{M}$ if for all $\theta' \in \Theta^{\text{LID}}$, $\Phi(\mathcal{M}, \pi_{\theta'}) \geq \kappa$.

We want to compute the LID for the source task $\Theta^{\text{LID}}_{\text{source}}$ such that it provides sufficient plasticity for downstream adaptation. This is the main mechanical step in CERTALIGN: instead of regularising updates toward θ_0, MaxLID searches for a large axis-aligned box around θ_0 in which every policy is certified to remain source-behaviour-compatible. The optimisation expands the lower and upper parameter bounds, uses interval bound propagation to lower-bound the worst-case surrogate over the whole region, and penalises candidate LIDs whose certified lower bound violates the sound threshold. We solve the following primal-dual problem (Elmecker-Plakolm et al., 2025):

$$\min_{\ell,u}\max_{\lambda\geq 0} -\Psi_\alpha + \lambda g + \frac{\mu}{2}[g]_+^2, \tag{10}$$

where

$$\Psi_\alpha(\ell,u) = \frac{1}{p}\sum_{j=1}^{p}\left[\alpha\log(u_j-\ell_j+\varepsilon) + (1-\alpha)(u_j-\ell_j)\right],$$
$$g(\ell,u) = \delta - \underline{\widetilde{\Phi}}_{\text{acc}}(\ell,u,\tau). \tag{11}$$

The objective function $\Psi_\alpha(\ell,u)$ measures the size of the parameter subspace, with α trading off log volume against total interval width and ε preventing $\log 0$. ℓ and u are the lower and upper parameter bounds defining the LID $\{\theta : \ell \leq \theta \leq u\}$. The constraint $g(\ell,u) = \delta - \underline{\widetilde{\Phi}}_{\text{acc}}(\ell,u,\tau)$ requires the certified worst-case surrogate $\underline{\widetilde{\Phi}}_{\text{acc}}(\ell,u,\tau)$, computed with temperature τ, to exceed the threshold δ; we provide details on how $\underline{\widetilde{\Phi}}_{\text{acc}}(\ell,u,\tau)$ is defined in Appendix D. $\lambda \geq 0$ is the Lagrange multiplier, μ is the augmented penalty weight and $[g]_+ = \max(g,0)$ penalises only violated surrogate constraint.

2.4. Phase 3: Downstream Adaptation with PGD

Once $\Theta^{\text{LID}}_{\text{source}}$ is computed, we perform adaptation to the downstream MDP $\mathcal{M}_{\text{down}}$. This can be done via any policy gradient (PG) RL method and any projected gradient descent (PGD) strategy. In our experiments, we employ Proximal Policy Optimisation (PPO) (Schulman et al., 2017) and use element-wise clipping as a projection operator $\Pi_{\Theta^{\text{LID}}_{\text{source}}}$.

2.5. Assumptions and Theoretical Guarantees

We now state the assumptions under which we can provide strong alignment guarantees.

Assumption 2.5 (Discrete finite action space)**.** In $\mathcal{M}_{\text{source}}$ and $\mathcal{M}_{\text{down}}$, the action space $\mathcal{A}$ is discrete and finite.

Assumption 2.6 (Greedy policy deployment)**.** At deployment, the policy selects actions greedily, i.e., $a^\star = \arg\max_{a'\in\mathcal{A}(s)} \pi_\theta(a'|s)$.

Assumption 2.7 (Sufficiency of behaviour demonstration)**.** The dataset D_{source} collected by querying the NoAdapt policy in $\mathcal{M}_{\text{source}}$ is sufficient for the desired source behaviour:

$$\exists D_{\text{source}} :\ \widehat{\Phi}\left(D_{\text{source}},\pi\right) = 1 \Rightarrow \Phi(\mathcal{M}_{\text{source}},\pi) = 1\ \forall\pi. \tag{12}$$

Assumptions 2.5–2.6 make action agreement well-defined under greedy deployment. Assumption 2.7 is a coverage condition; D_{source} is generated by querying π_{θ_0} in the source environment and can be enlarged as needed.

Theorem 2.8 (Continual RL adaptation with certified behaviour alignment)**.** *Let Assumptions 2.5-2.7 hold and let $\Theta^{\text{LID}}_{\text{source}}$ be the certified parameter region returned by Phase 2 of Algorithm 1. Then:*

$$\Phi_{\text{acc}}\left(\mathcal{M}_{\text{source}},\pi_{\theta'}\right) = 1\ \ \forall\,\theta' \in \Theta^{\text{LID}}_{\text{source}}.$$

In particular, since Phase 3 updates the policy by projected gradient descent onto $\Theta^{\text{LID}}_{\text{source}}$, every iteration θ_k satisfies

$$\Phi_{\text{acc}}\left(\mathcal{M}_{\text{source}},\pi_{\theta_k}\right) = 1.$$

We provide the proof of Theorem 2.8 in Appendix B. The theorem gives an a priori retention guarantee: every projected downstream update remains source-behaviour-aligned by construction.

Panel A: `Frozen Lake`

Method	diagonal_10x10		diagonal_20x20		diagonal_30x30		diagonal_60x60	
	$J_{\text{source}} \uparrow$	$J_{\text{down}} \uparrow$	$J_{\text{source}} \uparrow$	$J_{\text{down}} \uparrow$	$J_{\text{source}} \uparrow$	$J_{\text{down}} \uparrow$	$J_{\text{source}} \uparrow$	$J_{\text{down}} \uparrow$
NoAdapt	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00
UncAdapt	$\underline{0.40} \pm 0.49$	$\underline{0.90} \pm 0.30$	0.10 ± 0.30	$\mathbf{1.00} \pm 0.00$	0.00 ± 0.00	$\mathbf{1.00} \pm 0.00$	$\underline{0.20} \pm 0.40$	$\mathbf{0.90} \pm 0.30$
EWC	0.30 ± 0.46	$\mathbf{1.00} \pm 0.00$	$\underline{0.30} \pm 0.46$	$\mathbf{1.00} \pm 0.00$	$\underline{0.20} \pm 0.40$	$\mathbf{1.00} \pm 0.00$	0.10 ± 0.30	$\underline{0.70} \pm 0.46$
CERTALIGN (ours)	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.00$	$\mathbf{1.00} \pm 0.00$	$\underline{0.90} \pm 0.30$	$\mathbf{1.00} \pm 0.00$	0.20 ± 0.40

Panel B: `Lunar Lander`

Method	sluggish				underpowered			
	$J_{\text{source}} \uparrow$	$\text{SR}_{\text{source}} \uparrow$	$J_{\text{down}} \uparrow$	$\text{SR}_{\text{down}} \uparrow$	$J_{\text{source}} \uparrow$	$\text{SR}_{\text{source}} \uparrow$	$J_{\text{down}} \uparrow$	$\text{SR}_{\text{down}} \uparrow$
NoAdapt	215.02 ± 5.26	1.00 ± 0.00	-55.29 ± 35.72	0.00 ± 0.00	214.32 ± 8.52	1.00 ± 0.00	-22.06 ± 140.35	0.30 ± 0.46
UncAdapt	172.36 ± 79.66	$\underline{0.90} \pm 0.30$	$\mathbf{164.09} \pm 51.75$	$\mathbf{1.00} \pm 0.00$	98.91 ± 157.62	0.70 ± 0.46	$\mathbf{181.32} \pm 47.09$	$\mathbf{1.00} \pm 0.00$
EWC	$\mathbf{216.07} \pm 6.51$	$\mathbf{1.00} \pm 0.00$	11.45 ± 140.88	$\underline{0.50} \pm 0.50$	$\underline{183.63} \pm 81.55$	$\underline{0.90} \pm 0.30$	$\underline{135.43} \pm 128.52$	$\underline{0.90} \pm 0.30$
CERTALIGN (ours)	$\underline{215.02} \pm 5.26$	$\mathbf{1.00} \pm 0.00$	$\underline{39.33} \pm 137.75$	$\underline{0.50} \pm 0.50$	$\mathbf{214.32} \pm 8.52$	$\mathbf{1.00} \pm 0.00$	129.23 ± 115.01	$\underline{0.90} \pm 0.30$

Table 1. Source–downstream adaptation results across `Frozen Lake` and `Lunar Lander`. Source-task metrics measure behaviour retention, while downstream-task metrics measure adaptation plasticity. We report total undiscounted return J and, for `Lunar Lander`, success rate SR (mean $\pm$ standard deviation over 10 seeds). In `Frozen Lake`, $J = \text{SR}$. Bold and underlining denote the best and second-best adaptation methods, respectively, within each setting and metric. *NoAdapt* is included as a non-adaptive reference.

3. Experiments

We evaluate **CERTALIGN** against unconstrained adaptation (**UncAdapt**) and Elastic Weight Consolidation (**EWC**) (Kirkpatrick et al., 2017). UncAdapt optimises the downstream task without alignment constraints; EWC regularises changes to source-important parameters. We also report **NoAdapt**, the source-trained policy π_{θ_0} without downstream updates. We use `Frozen Lake` and `Lunar Lander` as environments. Those have discrete actions, shared state–action spaces across tasks, and task IDs appended to observations. Source–downstream shifts alter transition dynamics: `Frozen Lake` changes hole placement near the start, while `Lunar Lander` uses `underpowered` and `sluggish` vehicle-dynamics shifts. Pipeline details and visualisations are in Appendix E.

Results. Table 1 shows the retention–plasticity trade-off across `Frozen Lake` and `Lunar Lander`. Among the adaptation methods, CERTALIGN is the only one that preserves source-task behaviour both provably, by construction of the certified LID, and empirically, maintaining perfect source-task success in all pipelines. In contrast, UncAdapt achieves strong downstream performance, but often suffers substantial source-task degradation. This is the expected failure mode of unconstrained adaptation: downstream gradients can move the policy into regions of parameter space that solve the new task while changing the source-task action choices. EWC improves retention relative to UncAdapt in some settings, but provides no behaviour-alignment guarantee and still forgets in several cases. EWC uses a quadratic penalty that discourages changes in parameters estimated to be important for the source task. However, EWC does not certify preservation of the source-task action decisions, so it cannot rule out small parameter changes that alter the behaviour.

NoAdapt retains the source behaviour by not updating the policy, but its poor downstream performance shows its lack of robustness against the downstream transition-dynamics shifts. CERTALIGN achieves competitive downstream performance in several settings, matching the best downstream return in `diagonal 10x10`, `diagonal 20x20`, and `diagonal 30x30`, and obtaining non-trivial adaptation in `Lunar Lander`. However, its downstream plasticity is weaker in harder settings, especially in `diagonal 60x60` and `sluggish`. In `diagonal 60x60`, the larger state space makes the certified parameter region more restrictive relative to the downstream adaptation required. In the `sluggish` shift, the vehicle responds less strongly to the same control inputs, making safe landing more challenging. These results indicate that the certified LID can be too conservative. Improving the computation of LIDs that retain the source-task behaviour while allowing meaningful downstream updates is therefore an important direction for future work. We discuss one potential avenue in Appendix F.

4. Conclusion

We proposed CERTALIGN, a framework for continual RL with certified alignment. Our method constrains downstream adaptation to a locally invariant domain in parameter space, ensuring that the adapted policy does not forget the source-task behaviour. Empirical results show that CERTALIGN is the only adaptation method that preserves source-task behaviour both provably and empirically, while still allowing non-trivial downstream adaptation. Our method is focused on discrete action spaces and behaviour retention using a generated source-behaviour dataset. Future work may extend the framework to continuous actions, richer behavioural specifications, and LID computation techniques that provide more downstream plasticity.

References

Anisimov, M., Belardinelli, F., and Wicker, M. Safeadapt: Provably safe policy updates in deep reinforcement learning, 2026. URL https://arxiv.org/abs/2604.09452.

Elmecker-Plakolm, L., Fasterling, P., Sosnin, P., Tsay, C., and Wicker, M. Provably safe model updates. *arXiv preprint arXiv:2512.01899*, 2025. Submitted to IEEE SaTML 2026.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Khetarpal, K., Riemer, M., Rish, I., and Precup, D. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwińska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences (PNAS)*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Mirman, M., Gehr, T., and Vechev, M. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning (ICML)*, pp. 3578–3586. PMLR, 2018.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. Experience replay for continual learning. *arXiv preprint arXiv:1811.11682*, 2019.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

van de Ven, G. M., Tuytelaars, T., and Tolias, A. S. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022. doi: 10.1038/s42256-022-00568-3. URL https://doi.org/10.1038/s42256-022-00568-3.

A. Preliminaries

Markov Decision Process (MDP). We model tasks as a Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu_0)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition dynamics $P(s'|s, a)$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, discount factor $\gamma \in [0, 1)$, and initial state distribution μ_0. Since not all actions may be executable in every state, we also define the set of state-specific actions as $\mathcal{A}(s)$. A *policy* π, parametrised by θ, is a function $\pi_\theta : \mathcal{S} \to \mathcal{A}$ that maps states into action distributions and is used to define a decision-making strategy in $\mathcal{M}$.

Reinforcement Learning (RL). Provided that a task is modelled as an MDP, the RL objective is to find a policy maximising the *expected (discounted) return* $J(\theta)$ defined as:

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t\, r(s_t, a_t)\right], \tag{13}$$

where the expectation is over trajectories $\tau = (s_0, a_0, s_1, a_1, \ldots)$ with $s_0 \sim \mu_0$, $a_t \sim \pi_\theta(\cdot|s_t)$, and $s_{t+1} \sim P(\cdot|s_t, a_t)$.

B. Theorem Proofs

Following Anisimov et al. (2026), we state a theorem for the settings in which one wants to ensure that a single action is chosen in a state s.

Theorem B.1. *Denote the set of desired (allowed) actions in state s as $\mathcal{A}^{\text{allow}}(s)$. We denote the cardinality of this set as $|\mathcal{A}^{\text{allow}}(s)|$. Also, suppose the neural policy $\pi(a|s)$ maps a state vector into a vector of action logits $z_\theta(s)$. Then:*

$$\sum_{a\in\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)) > \frac{|\mathcal{A}^{\text{allow}}(s)|}{1 + |\mathcal{A}^{\text{allow}}(s)|} \implies \arg\max_{a\in\mathcal{A}} \pi(a|s) \in \mathcal{A}^{\text{allow}}(s). \tag{14}$$

Proof. Assume the following inequality holds in the state s:

$$\sum_{a\in\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)) > \frac{|\mathcal{A}^{\text{allow}}(s)|}{1 + |\mathcal{A}^{\text{allow}}(s)|}.$$

From that, we can get:

$$\frac{\sum_{a\in\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s))}{|\mathcal{A}^{\text{allow}}(s)|} > \frac{1}{1 + |\mathcal{A}^{\text{allow}}(s)|}.$$

This implies that the average softmax of allowed actions exceeds $\frac{1}{1+|\mathcal{A}^{\text{allow}}(s)|}$. Then, the maximum softmax among allowed actions must exceed $\frac{1}{1+|\mathcal{A}^{\text{allow}}(s)|}$ as well:

$$\max_{a\in\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)) > \frac{1}{1 + |\mathcal{A}^{\text{allow}}(s)|}. \tag{15}$$

Since the allowed-action softmax mass exceeds $\frac{|\mathcal{A}^{\text{allow}}(s)|}{1+|\mathcal{A}^{\text{allow}}(s)|}$, the prohibited-action softmax mass cannot exceed $\frac{1}{1+|\mathcal{A}^{\text{allow}}(s)|}$:

$$\sum_{a\notin\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)) < \frac{1}{1 + |\mathcal{A}^{\text{allow}}(s)|}$$

The upper bound for the sum of non-negative values is also the upper bound for the maximum term in the sum:

$$\max_{a\notin\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)) < \frac{1}{1 + |\mathcal{A}^{\text{allow}}(s)|} \tag{16}$$

From Eqs. 15 and 16, we get:

$$\max_{a\in\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)) > \max_{a\notin\mathcal{A}^{\text{allow}}(s)} \text{softmax}_\tau(z_{\theta,a}(s)). \tag{17}$$

Since softmax is a strictly monotonically increasing function,

$$\max_{a\in\mathcal{A}^{\text{allow}}(s)} z_{\theta,a}(s) > \max_{a\notin\mathcal{A}^{\text{allow}}(s)} z_{\theta,a}(s),$$

which can be equivalently represented as

$$\arg\max_{a\in\mathcal{A}} \pi(a|s) \in \mathcal{A}^{\text{allow}}(s). \tag{18}$$

□

Proof of Theorem 2.3.

Proof. When we aim to replicate one action per state in the dataset D, hence $|\mathcal{A}^{\text{allow}}(s)| = 1$ and $\frac{|\mathcal{A}^{\text{allow}}(s)|}{1+|\mathcal{A}^{\text{allow}}(s)|} = \frac{1}{2}$. Using Theorem B.1, we get the following implication:

$$\forall s:\ \text{softmax}_\tau(z_{\theta,a}(s)) > 0.5 \implies \arg\max_{a\in\mathcal{A}} \pi(a|s) \in \mathcal{A}^{\text{allow}}(s).$$

Suppose the policy $\pi(a|s)$ is parametrised with θ and let us denote it as $\pi_\theta(a|s)$. Then:

$$\forall s:\ \arg\max_{a\in\mathcal{A}} \pi_\theta(a|s) \in \mathcal{A}^{\text{allow}}(s) \implies \phi_{\text{acc}}(s,a;\theta) = 1.$$

Since this holds for any s, it also holds $\forall s \in D$:

$$\forall s \in D:\ \text{softmax}_\tau(z_{\theta,a}(s)) > 0.5 \implies \phi_{\text{acc}}(s,a;\theta) = 1.$$

Now, recall the following definitions:

- $\widetilde{\Phi}_{\text{acc}}(D,\theta) := \min_{(s,a)\in D} \text{softmax}_\tau\left(z_{\theta,a}(s)\right)$
- $\widehat{\Phi}_{\text{acc}}(D,\theta) := \min_{(s,a)\in D} \phi_{\text{acc}}(s,a;\theta)$.

Suppose $\widetilde{\Phi}_{\text{acc}}(D,\theta) > 0.5$. Then, $\forall (s,a) \in D:\ \phi_{\text{acc}}(s,a;\theta) = 1$. Finally, we get:

$$\widetilde{\Phi}_{\text{acc}}(D,\theta) > 0.5 \implies \widehat{\Phi}_{\text{acc}}(D,\theta) = 1.$$

□

Proof of Theorem 2.8.

Proof. Let D_{source} denote the behaviour demonstration collected in the source MDP $\mathcal{M}_{\text{source}}$. By construction, Phase 2 of Algorithm 1 returns a certified local invariant domain $\Theta^{\text{LID}}_{\text{source}}$ with the following property: $\forall\theta' \in \Theta^{\text{LID}}_{\text{source}}$ the certified lower bound used by MaxLID is sufficient to guarantee agreement with the demonstrated action on every state–action pair (s,a) in D_{source} (under Assumption 2.6). Equivalently, for every $(s,a) \in D_{\text{source}}$,

$$\arg\max_{a'\in\mathcal{A}} z_{\theta'}(a' \mid s) = a.$$

This implication follows from the soundness of the certified surrogate used in Phase 2: satisfying the surrogate constraint over $\Theta^{\text{LID}}_{\text{source}}$ is sufficient to guarantee the required action-agreement constraint on the source demonstration. Therefore, every policy parameter $\theta' \in \Theta^{\text{LID}}_{\text{source}}$ perfectly preserves the demonstrated source behaviour on D_{source}, i.e.

$$\widehat{\Phi}_{\text{acc}}\left(D_{\text{source}}, \pi_{\theta'}\right) = 1 \qquad \forall\theta' \in \Theta^{\text{LID}}_{\text{source}},$$

By Assumption 2.7, perfect preservation of the behaviour demonstration is sufficient for perfect behaviour alignment in the source MDP. Hence,

$$\Phi_{\text{acc}}\left(\mathcal{M}_{\text{source}}, \pi_{\theta'}\right) = 1 \qquad \forall \theta' \in \Theta^{\text{LID}}_{\text{source}},$$

which proves the first claim.

It remains to show that the guarantee is maintained during downstream adaptation. In Phase 3, each policy update is followed by projection onto $\Theta^{\text{LID}}_{\text{source}}$. Thus, for every iteration k,

$$\theta_k \in \Theta^{\text{LID}}_{\text{source}}.$$

Applying the result above with $\theta' = \theta_k$ gives

$$\Phi_{\text{acc}}\left(\mathcal{M}_{\text{source}}, \pi_{\theta_k}\right) = 1 \qquad \forall \theta_k.$$

Therefore, all downstream PGD updates preserve the certified source behaviour. □

C. MaxLID Solver Details

Algorithm 2 expands an interval box around θ_0 while maintaining the certified surrogate constraint. The key certification step is the IBP lower bound $\underline{\widetilde{\Phi}_{\text{acc}}}(\ell, u, \tau)$: if this lower bound exceeds δ, every parameter vector in the box satisfies the sound surrogate constraint.

Algorithm 2 MaxLID: Primal-dual interval expansion

Input: centre θ_0, threshold δ, temperature τ, dataset D_{source}, step sizes, penalty μ
Initialise $\ell \leftarrow \theta_0$, $u \leftarrow \theta_0$, $\lambda \leftarrow 0$.
for optimisation step $m = 1, \ldots, M$ **do**
  Compute IBP logit bounds over $\{\theta : \ell \le \theta \le u\}$ for all $(s, a) \in D_{\text{source}}$.
  Evaluate $g(\ell, u) = \delta - \underline{\widetilde{\Phi}_{\text{acc}}}(\ell, u, \tau)$.
  Update (ℓ, u) by descending $-\Psi_\alpha(\ell, u) + \lambda g + \frac{\mu}{2}[g]_+^2$.
  Project bounds so that $\ell \le \theta_0 \le u$.
  Update the dual variable $\lambda \leftarrow [\lambda + \mu g]_+$.
end for
Return $\Theta^{\text{LID}}_{\text{source}} = \{\theta : \ell \le \theta \le u\}$ if $\underline{\widetilde{\Phi}_{\text{acc}}}(\ell, u, \tau) \ge \delta$.

D. Interval Bound Propagation

Suppose we have access to a parameter subspace Θ computed for a policy that is to be executed in the MDP $\mathcal{M}$. Define the state space from demonstrations as $\mathcal{S}_D = \{s \in \mathcal{S} : s \in D\}$. Using IBP (Gowal et al., 2018; Mirman et al., 2018), we can compute logit bounds for each action $a \in \mathcal{A}$ in the state s inside Θ:

$$l_z(s, a) \le z_{a,\theta}(s) \le u_z(s, a).$$

In a vector form, this can be written as:

$$l_z(s) \le \mathbf{z}_\theta(s) \le u_z(s),$$

where $\mathbf{z}_\theta(s)$ contains logits for all actions in the state s. For each state s, let us define the worst-case logit vector $\mathbf{z}^{\text{worst}}_\theta(s)$ as follows:

$$\mathbf{z}^{\text{worst}}_\theta(s) = \mathbf{e}^{\text{allow}} * l_z(s) + (1 - \mathbf{e}^{\text{allow}}) * u_z(s),$$

where $\mathbf{e}^{\text{allow}}$ is the vector with 1s in the indices corresponding to allowed actions and 0s otherwise. In words, $\mathbf{z}^{\text{worst}}_\theta(s)$ contains the logit lower bounds for allowed actions and logit upper bounds for prohibited actions. We denote a worst-case logit of the action a in the state s in the policy network π_θ as $z^{\text{worst}}_{\theta,a}(s)$. Then, we can define the worst-case surrogate over the parameter subspace θ as follows:

$$\underline{\widetilde{\Phi}_{\text{acc}}}(\ell, u, \tau) := \min_{(s,a) \in D} \text{softmax}_\tau\left(z^{\text{worst}}_{\theta,a}(s)\right). \tag{19}$$

While $\mathrm{softmax}_\tau\left(z^{\mathrm{worst}}_{\theta,a}(s)\right)$ measures the worst-case softmax mass on allowed actions in the state s, $\underline{\widetilde{\Phi}}_{\mathrm{acc}}(\ell, u, \tau)$ aggregates the softmax values using the minimum operator and therefore provides a sound method for representing the worst-case dataset-level surrogate value.

E. Environments

Frozen Lake. The source task is an $n \times n$ deterministic diagonal corridor from start to goal. The downstream task preserves grid size and goal location, but changes the hole placement near the start: nearby ice cells become holes, while nearby holes become ice cells. Thus, an action that is safe in $\mathcal{M}_{\mathrm{source}}$ may move the agent into a terminal failure state in $\mathcal{M}_{\mathrm{down}}$.

Lunar Lander. The source task is the deterministic default vehicle on fixed terrain. In the `underpowered` downstream shift, engine thrust is reduced and vehicle mass is increased, so actions produce weaker acceleration and slower trajectory correction. In the `sluggish` shift, reduced thrust, heavier body and legs, weaker leg springs and linear/angular damping make the same actions yield less responsive attitude control and different landing dynamics. Figure 2 shows that NoAdapt completes the source task but fails in the `sluggish` downstream setting.

Figure 1. Frozen Lake diagonal_10x10.

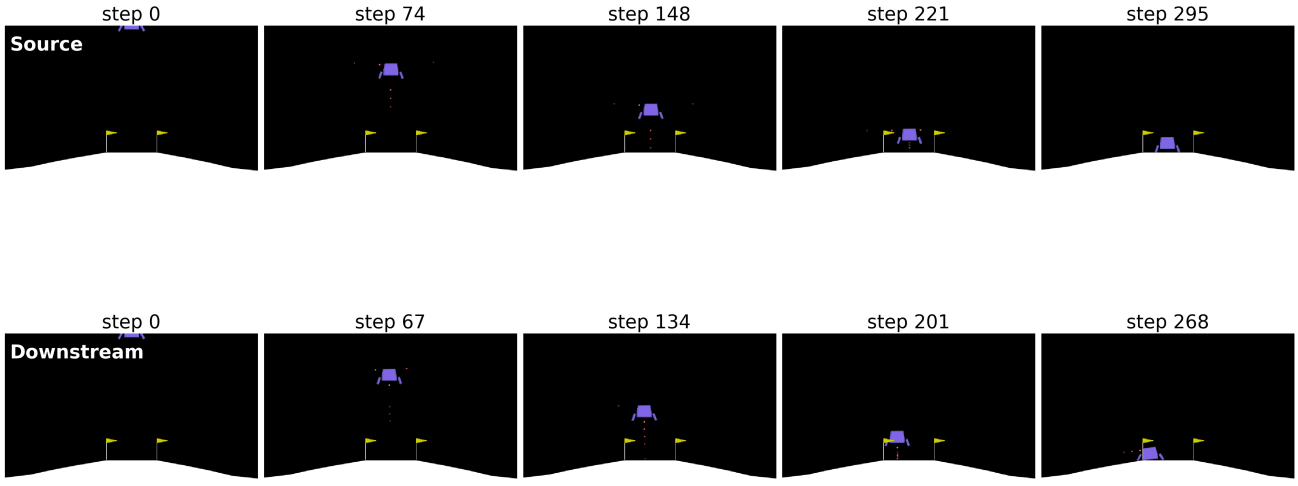

Figure 2. NoAdapt actor in Lunar Lander (`sluggish`). The actor is able to land the spacecraft safely in the source task. In the downstream task, the vehicle becomes more sluggish causing a landing failure for the NoAdapt actor.

F. Future Work (CERTALIGN+)

One of the limiting assumptions of the LID computation method in Elmecker-Plakolm et al. (2025) is that the LID is convex. In principle, the behaviour-invariant parameter space can be non-convex. Therefore, to make the method more complete, we propose CERTALIGN+ (C+) – a method that allows building a non-convex LID by taking a union of N convex LIDs:

$$\Theta^{\mathrm{C+}}_{\mathrm{source}} = \bigcup_{i=1}^{N} \Theta^{\mathrm{LID},(\mathrm{i})}_{\mathrm{source}}. \tag{20}$$

Figure 3 illustrates how the non-convex certified alignment region may look in the 3D parameter space. We call this non-convex set a Rashomon set, and we call the actor updated with PGD onto this set a Rashomon actor.

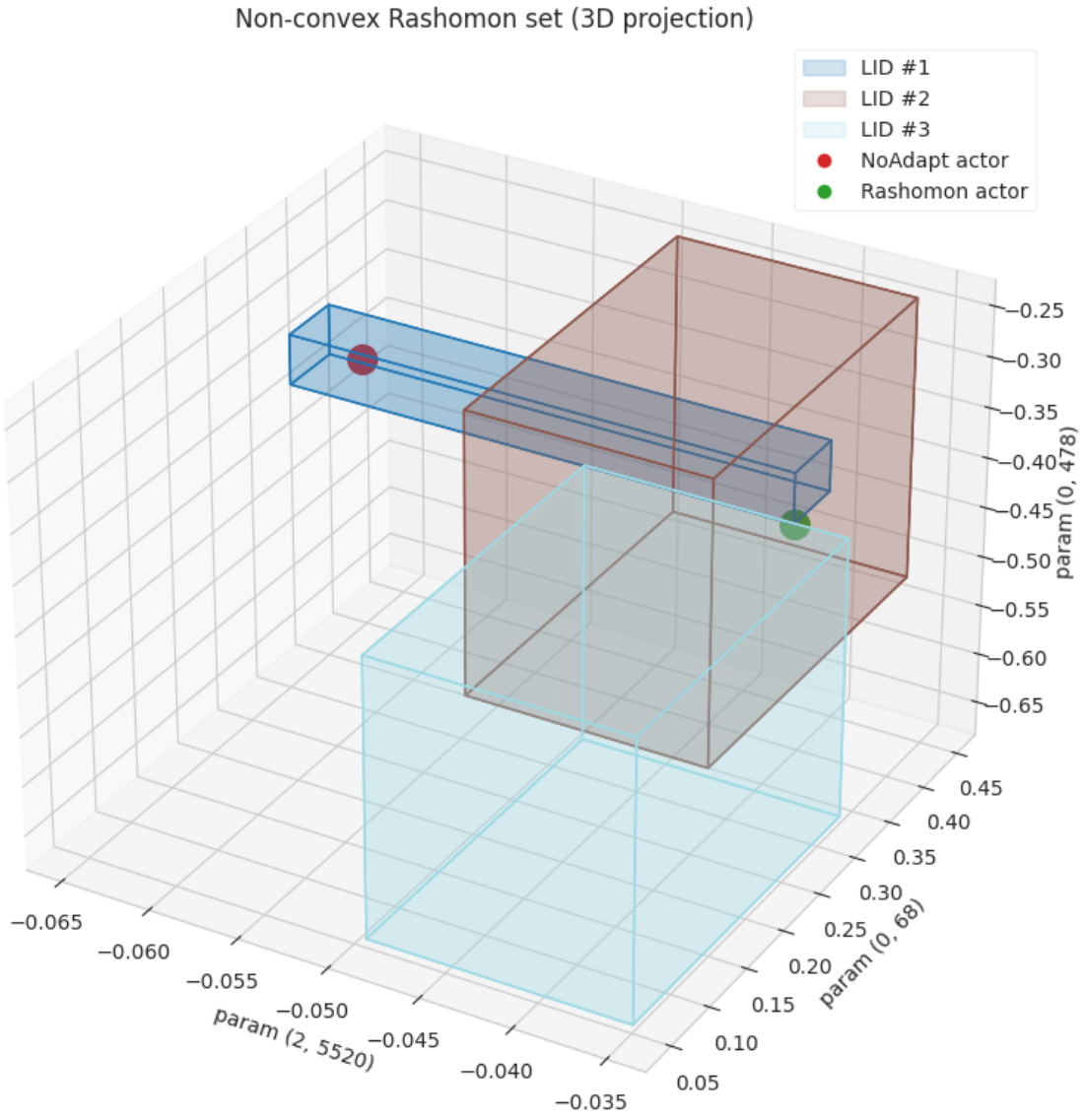

Figure 3. Non-convex certified alignment region: 3D projection.