Trust Region Policy Optimization for Functional Linear Policies

Anonymous authors
Paper under double-blind review

Abstract

Reinforcement Learning (RL) tasks where the states are given by spatial or temporal measurements often lead to high-dimensional state spaces, making function approximation difficult and unstable. We adapt the classic RL framework to allow the direct use of the inherent functional state, which can be estimated from the discrete measurements. We propose a suitable family of policies based on functional linear models, allowing us to take actions conditionally on functional states. Moreover, we extend Trust Region Policy Optimization (TRPO) to improve such policies and address the challenge of operator inversion in infinite-dimensional spaces using techniques from Functional Data Analysis (FDA). Furthermore, we implement Proximal Policy Optimization (PPO) for these policies. In experiments on three PDE control tasks, functional policies yield more stable training and achieve better performance than multilayer perceptron policies, highlighting the benefits of functional representations in RL.

Introduction

Reinforcement Learning (RL) can be extended beyond a finite-dimensional setting to handle state measurements, which are functions in the infinite-dimensional space of square-integrable functions. Despite this added complexity, the core RL framework remains intact. With appropriately defined policies, standard policy gradient methods are still applicable—but their implementation requires inverting a linear operator in an infinite-dimensional space. We tackle this challenge by leveraging techniques from Functional Data Analysis (FDA).

From playing video games (Lample & Chaplot, 2017) to smart grid control (Arwa & Folly, 2020) and even solving control problems (Recht, 2019; Zhang et al., 2024), the RL paradigm has allowed to tackle an important number of real-world problems. Markov Decision Processes (MDP) are the backbone of most RL methods, which rely on a Markovian hypothesis: future states and future rewards depend on past states and actions only through the last state and action; as a direct consequence, the policy must choose an action using only the current state. To use temporal information, a popular workaround is to augment the state space, for example, by including the last 4 frames of an Atari game (Mnih et al., 2015) or using the 24 hourly measures of electric load and electricity prices to dynamically manage electricity production (Ji et al., 2019). Another type of relevant state space is of spatial character, consisting of measurements of the same quantity across a fine grid; for example, in the *heat invader* control problem, an action must be chosen depending on heat measures in a 64×64 spatial grid (Farahmand et al., 2017).

Spatial or temporal states are challenging due to their high-dimensional nature. In the literature, these problems have mainly been approached using Deep Reinforcement Learning (DRL) by using value based methods with approximation (Mnih et al., 2015; Ji et al., 2019; Farahmand et al., 2017). In contrast, we develop policy based methods, approaching these high-dimensional problems from the FDA perspective, which is a popular approach to analyze data that are curves (Ramsay & Silverman, 2005) dealing with the infinite-dimensionality by using dimension reduction techniques such as *Functional Principal Component Analysis* (FPCA).

Adopting the framework introduced by Hernandez-Lerma (2001), we propose an MDP setting where the state space is a functional space, we define linear functional policies, and then adapt Trust Region Policy Optimization (TRPO) (Schulman et al., 2015a) to improve such policies. We show that the main theoretical results from TRPO generalize well into this setting, but the practical implementation is challenging because a linear-inverse problem must be solved in an infinite-dimensional space. We tackle these issues by exploring different FDA techniques: finite basis projection, FPCA (Wang et al., 2016) and a resolvent approach (Martini et al., 2022). Additionally, based on these FDA techniques, we propose a Proximal Policy Optimization (PPO) update.

Main Contributions: We introduce a family of policies, Functional Linear Policies (FLPs), which take continuous actions depending on functional states. These policies are direct adaptations of functional linear models (Cardot et al., 1999), which are well known in the FDA literature but have never been used in a RL context. Additionally, we prove that the main theoretical result from TRPO still holds for FLPs, and we adapt TRPO to propose practical algorithms to improve these policies. Prior work proposed the TRPO update in a classical RL context, with finite action and state spaces (Schulman et al., 2015a). Our main theoretical result follows the same proof of Schulman et al. (2015a, Theorem 1) and the proposed practical algorithms deal with issues arising from the highly dimensional setting by relying on classical FDA techniques: finite basis projection, FPCA (Wang et al., 2016) and a resolvent approach (Kreyszig, 2007).

1 Background

1.1 RL in Borel Spaces

Let us consider an MDP $(\mathcal{S}, \mathcal{A}, q, r)$ where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, which we suppose are Borel spaces, q is the transition law (probability measure on $\mathcal{S}$, given a state-action pair), and r is the reward function. We note Π the set of policies, which is defined as the set of probability distributions on $\mathcal{A}$, conditionally on a given state of $\mathcal{S}$. Hernandez-Lerma (2001) studied the construction of MDPs in this scenario.

Let us note $[\![1, n]\!]$ the set of integer numbers between 1 and n. Additionally, we note $\mathcal{B}(\mathcal{S})$ and $\mathcal{B}(\mathcal{A})$ the Borel sets of the state and action spaces, respectively.

Let H_t be the set of rollouts up to time $t \in \mathbb{N}$ and Ω the set of (countably) infinitely long rollouts: $H_t = \{(s_0, a_0, s_1, a_1, \dots, s_{t-1}, a_{t-1}, s_t) \mid s_k \in \mathcal{S}, a_k \in \mathcal{A}, k \in [\![1, t]\!]\}$ and $\Omega = \{(s_0, a_0, s_1, a_1, \dots) \mid s_k \in \mathcal{S}, a_k \in \mathcal{A}, k \in \mathbb{N}\}$.

Theorem 1 (Chapter 1, Section 2.5 Hernandez-Lerma (2001))**.** *Let $(\mathcal{S}, \mathcal{A}, q, r)$ be a MDP, let $(\Omega, \mathcal{F})$ be the measurable space where $\mathcal{F}$ is the product σ-algebra on Ω. Then for every policy $\pi \in \Pi$ and for every initial state distribution ρ, there exists a unique probability measure $\mathbb{P}_\rho^\pi$ on $(\Omega, \mathcal{F})$ satisfying:*

1. $\mathbb{P}_\rho^\pi(x_0 \in A) = \rho(A)$,
2. $\mathbb{P}_\rho^\pi(a_t \in B | h_t) = \pi(B | s_t)$,
3. $\mathbb{P}_\rho^\pi(s_{t+1} \in A | h_t, a_t) = q(A | s_t, a_t)$,

for all $A \in \mathcal{B}(\mathcal{S})$, $B \in \mathcal{B}(\mathcal{A})$, $h_t \in H_t$, $t \in \mathbb{N}$.

In the following, for any $\mathcal{F}$-measurable function $h \colon \Omega \to \mathbb{R}$, we note $\mathbb{E}_\rho^\pi(h)$, the expectation of h, which is the Lebesgue integral, with respect to the probability measure $\mathbb{P}_\rho^\pi$, that is:

$$\mathbb{E}_\rho^\pi(h) = \int_\Omega h(\omega) \mathbb{P}_\rho^\pi(d\omega).$$

Let $\gamma \in]0, 1[$ be a discount factor. The function $G = \sum_{t=0}^\infty \gamma^t r(s_t, a_t)$ is measurable and finite as long as the function r is bounded. In the rest of this document, we suppose that the function r is bounded on $\mathcal{S} \times \mathcal{A}$, thus expectation and sum may be interchanged. We can now consider the value, state-value, and

advantage functions, defined respectively as $V^\pi(s) = \mathbb{E}^\pi_\rho(G|s_0 = s)$, $Q^\pi(s,a) = \mathbb{E}^\pi_\rho(G|s_0 = s, a_0 = a)$ and $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$.

1.2 Linear Functional Policies

Separable Hilbert spaces are Borel spaces and thus can be considered under the presented framework. A notable example of such a space is the space of square-integrable real functions, noted $L^2([0,1])$. The space $L^2([0,1])$ is a Hilbert space, endowed with the scalar product:

$$\langle f, g\rangle_{L^2} = \int_{[0,1]} f(s)g(s)ds \ ; \quad f,\ g \in L^2([0,1]).$$

In a supervised context, this enables the creation of a diverse range of function-to-scalar regression models, including functional linear models (Cardot et al., 1999), generalized functional linear models (Müller & Stadtmüller, 2005), and functional generalized additive models (McLean et al., 2014).

The same construction can be used to define policies over functional state spaces. Indeed, consider the parameters $\beta \in L^2([0,1])$, $c \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, then a Functional Linear Policy (FLP) is the conditional probability distribution with the following density:

$$\pi_\theta(a|s) = (2\pi\sigma^2)^{-1/2} \exp\left(-(c + \langle\beta, s\rangle_{L^2} - a)^2/2\sigma^2\right),$$

with $a \in \mathbb{R}$ and $s \in L^2([0,1])$.

Consider any initial state distribution $\rho(\cdot)$ and define the objective function: $J(\theta) = \mathbb{E}^{\pi_\theta}_\rho(G)$. The goal of policy optimization methods is to find the parameter that maximizes the objective function $J(\cdot)$. The same problem arises for FLPs, with $\theta = (\beta, c, \sigma) \in \Theta = L^2([0,1]) \times \mathbb{R} \times \mathbb{R}^+$. Note that FLPs are differentiable with respect to θ: in the usual sense, for the real parameters c and σ, and as a Fréchet derivative for the functional parameter β (Hsing & Eubank, 2015, Section 3.6). Thus, it is possible to adapt policy gradient methods, such as REINFORCE (Sutton et al., 1999), NPG (Kakade, 2001), TRPO (Schulman et al., 2015a) or PPO (Schulman et al., 2017) in this framework.

1.3 TRPO

Let us consider the TRPO surrogate function, and the state average Kullback-Leibler divergence, defined respectively:

$$L_{\theta_{\text{old}}}(\theta) = \mathop{\mathbb{E}}_{\substack{s\sim\rho^{\theta_{\text{old}}} \\ a\sim\pi_{\theta_{\text{old}}}(\cdot|s)}} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)} Q^{\pi_{\theta_{\text{old}}}}(s,a)\right]$$

$$\bar{D}_{\text{KL}}(\theta_{\text{old}}||\theta) = \mathop{\mathbb{E}}_{s\sim\rho^{\theta}_{\text{old}}} \left(\text{KL}(\pi_{\theta_{\text{old}}}(\cdot|s), \pi_\theta(\cdot|s))\right).$$

TRPO is a policy gradient method designed for direct policy optimization. It achieves this by maximizing a surrogate objective $L_{\theta_{\text{old}}}(\cdot)$, which provides a first-order approximation of the true objective $J(\cdot)$ near θ_{old}. To ensure stable updates, TRPO constrains the state-average Kullback-Leibler (KL) divergence between successive policies, $\bar{D}_{\text{KL}}(\theta_{\text{old}}||\cdot)$. The method is theoretically grounded in a performance improvement bound (Schulman et al., 2015a, Theorem 1). The TRPO update aims to solve the optimization problem (P):

$$\max_\theta L_{\theta_{\text{old}}}(\theta), \text{ such that } \bar{D}^{\rho_{\theta_{\text{old}}}}_{\text{KL}}(\theta_{\text{old}}, \theta) \leq C. \tag{P}$$

Schulman et al. propose to compute the direction of the update by solving a sample-based estimation of the objective and constraints, using a first-order approximation for the objective and a second-order approximation for the constraint. Once the step direction is computed, a line-search is performed to ensure that the surrogate effectively improves while respecting the KL constraint. Concretely, let us note δ the step given: $\theta = \theta_{\text{old}} + \delta$. For the "single path" procedure, δ is computed by solving the problem:

$$\max_\delta \langle g, \delta\rangle, \text{ such that } \langle\delta, \Gamma\delta\rangle \leq C. \tag{P approx}$$

where g is the average gradient of the surrogate, $g = \mathbb{E}_{s\sim\rho_{\theta_{old}}}\left(\nabla_\theta L_{\theta_{old}}(\theta_{old})\right)$ and Γ is the average expected information matrix: $\Gamma = \mathbb{E}_{s\sim\rho_{\theta_{old}}}\left(\nabla^2_\theta \log \pi_{\theta_{old}}(\cdot|s)\right)/2$. For any value of C, there exists $\lambda > 0$ such that Problem (P approx) is equivalent to the unconstrained problem $\text{maximize}_\delta \langle g, \delta\rangle - \lambda/2\langle\delta, \Gamma\delta\rangle$, whose solution δ^*, verifies:

$$g = \lambda\Gamma\delta^*. \tag{1}$$

1.4 Proximal Policy Optimization

TRPO is a policy optimization method of second order, as it relies on a Kullback-Leibler constraint. Inspired by this method, Schulman et al. (2017), proposed Proximal Policy Optimization (PPO), a first-order policy optimization method, similar to the TRPO method. Instead of approaching a maximization problem under constraints, a clipped objective is proposed that enforces that the new policy is close to the current policy, in a certain sense. More specifically, let θ_{old} be the current parameter, and consider the observed probability ratio at instant t: $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$, the authors propose the following objective:

$$\mathbb{E}\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})\right)\hat{A}_t\right],$$

where $\text{clip}(x, a, b) = \min(\max(x, a), b))$ denotes the clipping operator, $\hat{A}_t$ is an estimate of the advantage function at the moment t and $\epsilon_{\text{clip}} > 0$ is a hyperparameter; a common value is $\epsilon_{\text{clip}} = 0.2$. The intuition, given by Schulman et al. (2017), is that for a given timestep t, if the new policy is close to the current one (e.g., $r_t(\theta) \in [1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}}]$), then the clipped objective is the same as the TRPO surrogate; otherwise, the second term $\text{clip}(r_t(\theta), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})$ removes the incentive of taking a step outside or the interval $[1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}}]$, as the final objective is the minimum of those two.

2 TRPO and PPO for FLPs

The main objective of this paper is to adapt TRPO and PPO to improve FLPs. The following result is the main theoretical contribution of this contribution; it allows us to propose practical TRPO methods.

Proposition 1. *Let π_θ, $\pi_{\tilde{\theta}}$ be two FLPs with respective parameters $\theta = (\beta, c, \sigma) \in \Theta$ and $\tilde{\theta} = (\tilde{\beta}, \tilde{c}, \tilde{\sigma}) \in \Theta$, and consider $M > 0$. If the state space is bounded: $\mathcal{S} = \{f \in L^2([0,1]) \mid \|f\|_{L^2} \leq M\}$. Then the function $s \mapsto D_{KL}(\pi_{\tilde{\theta}}(\cdot|s)||\pi_\theta(\cdot|s))$ is continuous, attains a maximum in $\mathcal{S}$ and the function $(s,a) \mapsto |A^{\pi_\theta}(s,a)|$ is bounded. Let $\alpha^2 = \max_{s\in\mathcal{S}} D_{KL}(\pi_\theta(\cdot|s)||\pi_{\tilde{\theta}}(\cdot|s))$ and $\epsilon = \sup_{s,a}|A^{\pi_\theta}(s,a)|$. The following bound holds:*

$$J(\tilde{\theta}) \geq L_\theta(\tilde{\theta}) - 4\epsilon\gamma\alpha^2/(1-\gamma)^2. \tag{2}$$

Proof. We provide a detailed proof in the Annex A. □

As stated in Proposition 1, the improvement bound from (Schulman et al., 2015a, Theorem 1) still holds when the state space is the set of bounded functions $L^2([0,1])$ and the family of policies considered are FLPs. Using this result, just as done by Schulman et al. (2015a), we derive a practical algorithm by replacing the maximum on the KL constraint by the state-average KL, obtaining the same optimization problem stated in Problem (P), except the parameter space is the infinite-dimensional space $\Theta = L^2([0,1]) \times \mathbb{R} \times \mathbb{R}^+$. Using sample-based estimations, the direction is computed using a first order approximation of the surrogate and a second order approximation of the state-average KL constraint. Thus, we obtain optimization Problem (P approx), except that the gradient of the surrogate g is not a vector but an element of Θ and Γ is not a matrix but an operator. The solution to this problem, $\delta^* \in \Theta$, still verifies (1).

Yet, a difficulty arises for the implementation of a practical algorithm: Equation (1) is a linear inverse problem in an infinite-dimensional space, the operator has no global inverse, and its pseudo-inverse may not even be continuous.

Luckily, this problem is common in the FDA literature, and there exist common techniques to work around this difficulty. Concretely, we implement and compare four methods: using a finite basis expansion (Section 2.1), FPCA (Section 2.2) or using a resolvent estimator (Section 2.3).

2.1 Naïve approach

A common technique in FDA (Wang et al., 2016) is to consider a finite expansion of the functional terms. Let $(\phi_i(\cdot))_{i\in\mathbb{N}}$ and $(\psi_j)_{j\in\mathbb{N}}$ be functional bases of $L^2([0,1])$. Consider two functions$f, g \in L^2([0,1])$, and suppose they can be written using N_ψ and N_ϕ terms from the functional bases, respectively. Then they are of the following form: $f(\cdot) = \sum_{i=1}^{N_\Psi} a_i \cdot \psi_i(\cdot) = \boldsymbol{a}^T \cdot \boldsymbol{\Psi(\cdot)}$ and $g(\cdot) = \sum_{j=1}^{N_\Phi} b_j \cdot \phi_j(\cdot) = \boldsymbol{b}^T \boldsymbol{\Phi(\cdot)}$, where $\boldsymbol{a} = (a_1, \dots, a_{N_\Psi})^T$ and $\boldsymbol{b} = (b_1, \dots, b_{N_\Phi})$ are vectors of coefficients, and $\boldsymbol{\Psi}(\cdot) = (\psi_1, \dots, \psi_{N_\Psi})$ and $\boldsymbol{\Phi}(\cdot) = (\phi_1, \dots, \phi_{N_\Psi})$ are vectors of functions. Then the functional scalar product between f and g can be written as a matrix product: $\langle f, g\rangle_{L^2} = \boldsymbol{a}^T \boldsymbol{R} \boldsymbol{b}$, where $\boldsymbol{R}$ is the $N_\phi \times N_\psi$ matrix with the inner product of the elements from the two functional bases, $\boldsymbol{R} = (\langle \phi_i, \psi_j\rangle)_{i\in[\![1,N_\Phi]\!], j\in[\![1,N_\Phi]\!]}$.

This can be used to turn TRPO for FLPs into a finite-dimensional problem: instead of computing $\delta(\cdot) \in L^2([0,1])$, if the step can be written in a finite functional basis: $\delta(\cdot) = \boldsymbol{b}^T \boldsymbol{\Phi}(\cdot)$, then it is enough to compute its functional coefficients, $\boldsymbol{b} \in \mathbb{R}^{N_\Phi}$.

2.2 FPCA approach

FPCA provides a functional basis of orthonormal eigenfunctions. Let, $u, v \in [0,1]$, we consider the functional states $s(\cdot)$, which are random curves following a distribution ρ. Let us note $\mu_\rho(u) = \mathbb{E}_{s\sim\rho}(s(u))$ the functional mean and $\Sigma_\rho(u,v) = \mathbb{E}_{s\sim\rho}((s(u) - \mu(u))(s(v) - \mu(v)))$ the functional covariance.

The covariance operator $\Sigma_\rho(f(\cdot)) = \int \Sigma(\cdot, u) f(u) du$ is a positive, symmetric, compact operator. By the Karhun-Loeve theorem, there exists an orthonormal functional basis $(\xi_k(\cdot))_{k\in\mathbb{N}}$, with corresponding eigenvalues, $(\lambda_k)_{k\in\mathbb{N}}$ such that: $\Sigma(u,v) = \sum_{k\in\mathbb{N}} \lambda_k \xi_k(u) \xi_k(v)$ and $\langle \xi_j, \xi_k\rangle_{L^2} = \mathbb{1}_{j=k}$, for all $j, k \in \mathbb{N}$. FPCA is particularly relevant for TRPO: if the functional mean of s, with respect to ρ is null, the quadratic constraint of Problem (P approx) becomes diagonal. Indeed, let $\theta = (\beta(\cdot), c, \sigma) \in \Theta$ and $\delta = (\delta_\beta(\cdot), \delta_c, \delta_\sigma) \in \Theta$, consider a state $s \in \mathcal{S}$, then the explicit form for state-average KL divergence is:

$$\bar{D}_{\mathrm{KL}}(\theta||\theta+\delta) = \log\left(\tfrac{\sigma+\delta_\sigma}{\sigma}\right) + \tfrac{\sigma^2 + \langle \delta_\beta, \Gamma_\Sigma(\delta_\beta)\rangle_{L^2}^2 + \delta_\sigma^2}{2(\sigma+\delta_\sigma)^2} - \tfrac{1}{2}. \tag{3}$$

Projecting $\delta_\beta(\cdot)$ upon the first $N_\xi \in \mathbb{N}$ components allows us to efficiently use a low rank approximation of the Kullback-Leibler divergence. Moreover, in the eigenbasis $(\xi_k)_{k=1}^{N_\xi}$, the second order approximation of the constraint is a diagonal quadratic form, with respect to the coefficients of $\delta_\beta(\cdot)$. Indeed, let us suppose $\delta_\beta(\cdot) = \sum_k^{N_\xi} b_k \xi_k(\cdot)$, then Equation 3 becomes:

$$\bar{D}_{\mathrm{KL}}(\theta||\theta+\delta) = \log\left(\tfrac{\sigma+\delta_\sigma}{\sigma}\right) + \tfrac{\sigma^2 + \sum_k^{N_\xi} \lambda_k b_k^2 + \delta_\sigma^2}{2(\sigma+\delta_\sigma)^2} - \tfrac{1}{2}, \tag{4}$$

which is a diagonal quadratic form in the functional coefficients $(b_k)_{k=1}^{N_\xi}$.

It is important to note that the eigenfunctions depend on the state distribution, ρ and thus the FPCA decomposition should be computed after every episode. In practice, to select the number of eigenfunctions, we keep only the ones with sufficiently large eigenvalues.

2.3 Resolvent approach

An alternative solution to equation 1 can be obtained through resolvents. The solution by projection is equivalent to approximating Γ^{-1} using a linear operator with additional regularity, $\Gamma^\dagger = \sum_{k=1}^{N_\xi} b(\lambda_k)(\xi_k \otimes \xi_k)$, where N_ξ is an increasing sequence of integers tending to infinity, and b is a smooth function converging pointwise to $x \mapsto 1/x$. Indeed, $\Gamma^\dagger \to \Gamma^{-1}$ as $N_\xi \to \infty$. Choosing $b(x) = 1/x$ for a finite N_ξ results in setting $\Gamma^\dagger$ to be a spectral cutoff approximation of Γ^{-1}. However, this choice is not unique. Consider the following family of functions $b_{n,p} : \mathbb{R}_+ \to \mathbb{R}_+$, parameterized by $p \in \mathbb{N}$, such that

$$b_{n,p}(x) = x^p/(x+\alpha_n)^{p+1},$$

where α_n is a strictly positive sequence that tends to 0 as $n \to \infty$.

Based on this, we can define the following class of solutions

$$\delta_p^* = b_p(\Gamma)g, \tag{5}$$

where $b_p(\Gamma) = (\Gamma + \alpha I)^{-(p+1)}\Gamma^p$, with $p \geq 0$, $\alpha > 0$. This resolvent class corresponds to regularized approximations of Γ^{-1} to address the inversion problem (Martini et al., 2022).

2.4 PPO for FLPs

We readily adapt the PPO update using the functional methods of Sections 2.1 and 2.2. We compute the observed probability ratio in terms of the functional scalar product, which can be computed using a finite basis expansion—for a given and known functional basis as explained in Section 2.1, or as a basis of eigenfunctions, as presented in Section 2.2. We cannot adapt the resolvent approach for PPO, as the resolvent approach considers a second method, but PPO is a first-order method.

3 Numerical Experiments

In this section we aim to answer the following questions:

1. Do our proposed methods work in practice?
2. How do they compare among themselves?
3. How do they compare against taking the raw observations and using standard DRL methods?

We do the best we can to answer these carefully by following guidelines presented by Patterson et al. (2024). We compare RL methods rigorously, using powerful statistical tests.

In theory, our methods should work well when the states are measurements from an underlying inherently functional state, such as the environments developed and implemented in `controlgym` python package (Zhang et al., 2024). Furthermore, in some of these environments, the optimal policy can be computed if the environment is completely known and perfectly observed, which provides a sort of ideal baseline for our RL methods.

3.1 PDE control environments

Let us first present the true continuous controlled PDE problem, which we then formulate as an RL problem with a functional state space.

Just as presented by Zhang et al. (2024), we consider a "one-dimensional PDE control environment with periodic boundary conditions and spatially distributed control inputs". We consider a spatial domain $[0, 1]$, let $T > 0$, and a continuous field $u\colon [0, 1] \times [0, T] \to \mathbb{R}$, where $u(\cdot, t)$ is the state function at an instant t. This continuous field evolves according to the a controlled PDE:

$$\frac{\partial u}{\partial t} - \mathcal{L}\left(\frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial^2 x}, \dots\right) = a(x, t), \tag{6}$$

with $x \in [0, 1]$, $t \in [0, T]$ and $\mathcal{L}$ is a linear differential operator and $a(x, t)$ is the distributed control force applied at location x at instant t.

Unlike what was proposed by Zhang et al. (2024), in this paper, we consider only a single scalar input, applied over the subdomain $[0.3, 0.7]$, that is, we suppose it is given by $a(x, t) = \mathbb{1}_{[0.3,0.7]}(x)a(t)$.

Let us consider a given ideal state function $u_{\text{ideal}}(\cdot) \in L^2([0, 1])$, we wish to find a distributed control, $a(\cdot)$, such that the following quantity is minimized:

$$\int_{[0,T]} \int_{[0,1]} (u(x, t) - u_{\text{ideal}}(x))^2 dx dt + \int_{[0,T]} a(t)^2 dt. \tag{7}$$

This problem can be formulated as an RL problem by discretizing the equation in time, with functional states and actions sampled from an FLP π_θ. For each time step t_i, the state observation is given by $s_i(\cdot) = u(\cdot, t_i) \in L^2([0,T])$, the action is the input applied at the instant t_i, which in turn is sampled from an FLP $a_i \sim \pi_\theta(\cdot|s_i)$ and the reward is defined by $r_i = \int_{[0,1]}(s_i(x) - u_{\text{ideal}}(x))^2 dx - \lambda a_i^2$.

In practice, this controlled PDE is also discretized in space; we consider a fine grid with $N_x \in \mathbb{N}$ points $\{x_k\}_{k=1}^{N_x}$, at the timestep t_i, we compute the states only on points $s_i(x_k) \in \mathbb{R}$. In the package `controlgym`, these are computed using efficient numerical methods. At last, to mimic real-world scenarios, we only observe $0 < N_{\text{obs}} \leq N_x$ noisy versions of some of these points. Concretely, let $\boldsymbol{s}_i = (s_i(x_1), \ldots, s_i(x_{N_x}))$ be the vector of states at the discretization points; we observe a vector of noisy measurements $\boldsymbol{s}_i^{\text{obs}} \in \mathbb{R}^{N_{\text{obs}}}$

$$\boldsymbol{s}_i^{\text{obs}} = \boldsymbol{C}\boldsymbol{s}_i + \boldsymbol{e}_i \ ; \ \boldsymbol{e}_i \sim \mathcal{N}(\boldsymbol{0}_{N_{\text{obs}}}, \boldsymbol{\Sigma}), \tag{8}$$

where $\boldsymbol{C} \in \mathbb{R}^{N_{\text{obs}} \times N_x}$ is a matrix with a single 1 per row and zeros elsewhere, and the covariance matrix $\boldsymbol{\Sigma}$ is user-specified.

Although the state function $s_i(\cdot)$ is not truly observed, there exists an extensive literature on how to estimate it using the noisy discrete observations $\boldsymbol{s}_i^{\text{obs}}$, the main idea being choosing the functional coefficients in such a way that the mean-square error is minimized while penalizing the roughness of the obtained functional object. The reader may consult Ramsay & Silverman (2005) or Kreyszig (2007) for more details.

3.2 Method

We evaluate and compare our methods in all the linear PDE environments from `controlgym`, as these allow comparing performances against the ideal LQR controllers. Namely, we carry out our experiments in the three following environments: "wave", "convection_diffusion_reaction" and "schrodinger". We consider episodes with 100 temporal steps, with a temporal discretization step of 0.1, the spatial domain is discretized in 16 points, and observed only on 12 points. The rest of the parameters are the default-ones proposed in `controlgym`, and are specific for each PDE environment. As an intuitive baseline, we also compare our methods to the zero-controller, whose action is always 0.

We use a discount rate $\gamma = 0.99$ for all experiments, and, as recommended by Huang et al. (2022a), we use generalized advantage estimation (Schulman et al., 2015b), with weight $\lambda_{GAE} = 0.95$ in all experiments. This allows us to estimate the advantage function the same way for all methods.

For our functional methods, we use a Fourier basis, with 10 elements, to parametrize both the $\beta(\cdot)$ and the state functions. We use the implementation from the Python package `scikit-fda` (Ramos-Carreño et al., 2024). The $\beta(\cdot)$ parameter is initialized as the null-function and the constant c is initialized as 0; thus, during the first episode, actions are taken independently of the observed state. Additionally, we implement a critic network given by a fully connected feedforward neural network taking as input the functional coefficients of the state function and having two hidden layers of size 64. Each hidden layer of the critic is followed by a hyperbolic tangent activation function.

To compare against our methods, we consider the PPO controller implemented in the `controlgym` package. This controller uses a classical actor-critic structure, using an actor and a critic with independent networks, both parametrized by a multilayer perceptron, with ReLu activations and two hidden layers of size 64. All linear layers are initialized orthogonally as suggested by Saxe et al. (2014) with $\sqrt{2}$ gain and biases initialized as zero. This network takes as input the discrete values $\boldsymbol{s}_i^{\text{obs}} \in \mathbb{R}^{12}$.

Both the FLPs and the neural-network policy have a standard deviation parameterized as a state-free learnable parameter, which is initialized as $\sqrt{0.05}$, to encourage small early action outputs.

We adapt the PPO implementation from the `cleanRL` package (Huang et al., 2022b), which follows guidelines proposed by Huang et al. (2022a). We rely on automatic differentiation, implemented in `PyTorch` (Paszke et al., 2019). For each method, we run 20 episodes in parallel, and using these episodes, perform one PPO update. We repeat this until we obtain 4000 episodes; this defines one learning trajectory.

To provide a fair comparison between methods, as suggested by Patterson et al. (2024), we carefully tune each method separately in each environment. For a given environment, we test 20 combinations of hyperparameters,

running three seeds per set of hyperparameters, doing an efficient search using the `optuna` package (Akiba et al., 2019). In addition to the functional methods hyperparameters, we tune for the learning rate in PPO and both the critic learning rate and the maximum KL in TRPO. We provide a summary of the range of search of hyperparameters as well as the selected hyperparameter for each environment and method, as well as other parameters (number of gradient descents, batch size, etc.), in Annex B.

At last, we evaluate the performance of each tuned method, in each environment and method, by running 50 learning trajectories with different random seeds. As suggested by Patterson et al. (2024), we use the same random seeds across methods, allowing us to control the initial state across methods. Doing this, we obtained paired measurements, which we used to compare methods using paired statistical tests, which are more powerful than unpaired tests.

All experiments were conducted on a machine with an Intel(R) Xeon(R) Silver 4114 CPU and an NVIDIA RTX A6000 GPU. The complete set of experiments, including tuning and comparisons across all methods and environments, required approximately 330 hours of compute.

3.3 Results

In Figure 1, we present the evolution of the median episodic reward, calculated using the 50 random seeds. For comparison, we show the performances of the LQR and zero controllers. For visualization purposes, values smaller than baseline minus ten are truncated and displayed at the threshold. Additionally, we show the two-sided 0.95-confidence interval for the median, calculated using bootstrap. We observe that updating FLPs using any proposed method, either using PPO or TRPO, improves the performance of the functional policy. Indeed, at the end of training, the median episodic reward of FLPs is significantly better than those of the zero controller. Additionally, we note that functional methods perform similarly for a given algorithm, except when using PPO in the "convection" environment, where FPCA seems to outperform the naive method at the end of training.

In contrast, the performance of the Neural Network (NN) policy is not significantly better than the performance of the zero controller in any environment. Interestingly, for environments "convection" and "wave", using TRPO to improve the NN policy *sometimes* yields better performances than the zero controller, as the point estimate is above that baseline, but sometimes it is way lower than the zero controller, as the lower confidence interval is lower than the zero controller. With PPO in environments "convection" and "schrodinger", the NN policy improves stably but never above the zero controller, and in environment "convection" performances increase until they attain those of the zero controller. To sum up, updating the NN policy with PPO yields ineffective policies, and updating it with TRPO yields unstable performances. In Figure 2 we present the evolution of the median difference of episodic rewards of FLPs and the NN policy. Additionally, we show the one-sided 0.95-confidence interval for the median, calculated using bootstrap. For visualization purposes, we clip the median difference between policies and its lower bound at 50. We observe that using PPO with any functional method yields performances that are significantly better than using the NN policy, all through training. Similarly, when using TRPO, in the environments "schrodinger" and "wave", we observe that all the functional methods outperform the NN policy, and in the environment "convection", the resolvent approach does not seem significantly better than the NN policy.

At last, in Figure 3, we compare the performances of methods at the end of training. For each method and random seed, we compute the average reward over the last 20 episodes, then clip the obtained average at -500. This yields one scalar measurement of performance. We show the probability density of this quantity for each method and algorithm in each environment, estimated with the scalar measurements obtained from the 50 random seeds. Note that the x-axis is in the $\log 10$ scale. For reference, we add vertical lines with the performances of the LQR and zero controllers. As we observed in Figures 1 and 2, we can see that trained FLPs performances are better than the zero controller and usually are close to the ones of the ideal LQR controller. For NN policies, we observe that most of the density of the reward after training lies below the performance of the zero controller when using PPO and the density is bimodal when using TRPO—some trained NN policies will perform better than the zero-controller but some have bad performances.

To further evaluate the final performance of models, in each environment and for each algorithm, we compare them using paired non-parametric tests. We first evaluate if we observe a significant difference among

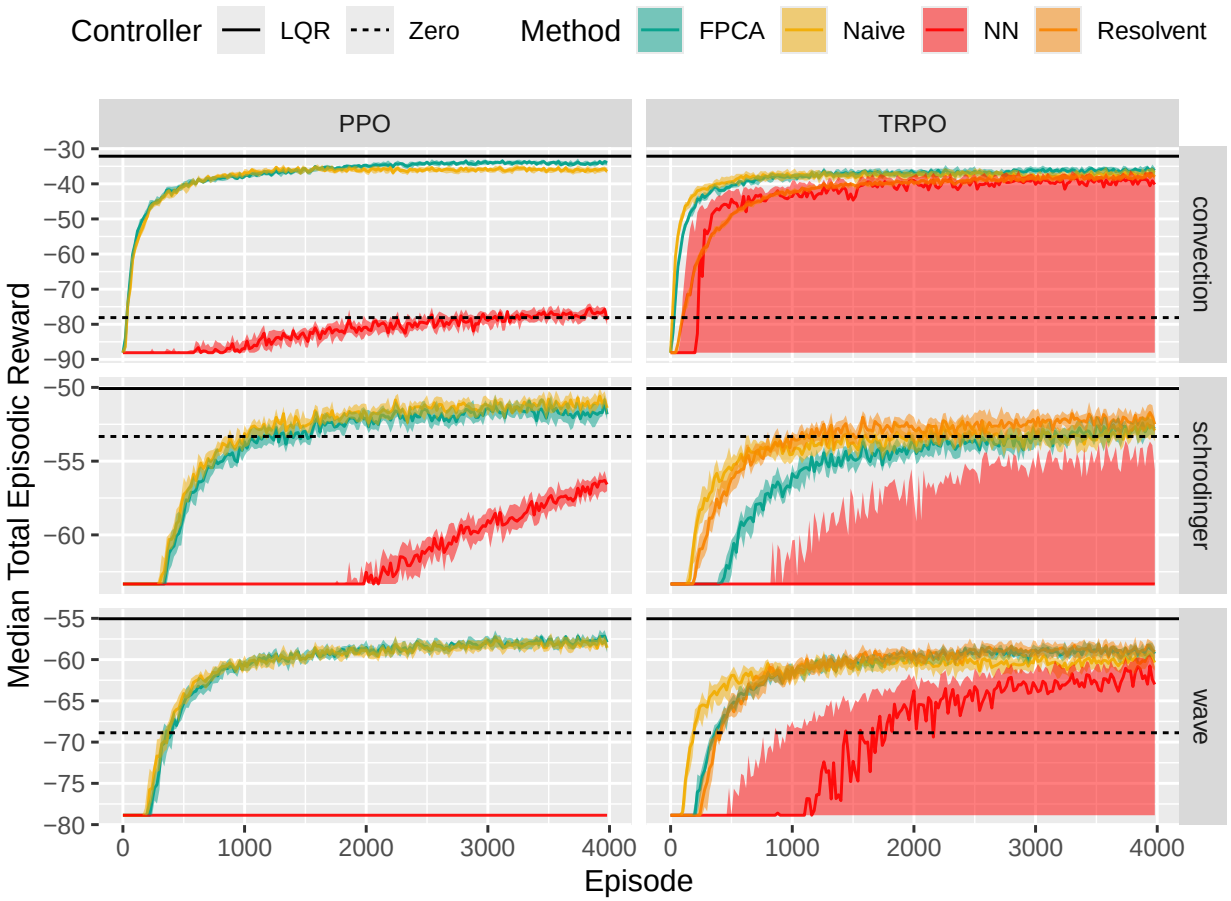

Figure 1: Median Episodic Reward vs. Episode for Different Algorithms and Methods.

methods, using the Friedman test (Friedman, 1937), and if there is one, we perform post-hoc comparisons using Wilcoxon's test (Mann & Whitney, 1947), correcting the p-values for the multiple testing using the Benjamini-Hochberg procedure (Benjamini & Hochberg, 1995). We observe that functional methods significantly outperform the NN policy in all environments when using PPO. Likewise, the naive and FPCA functional methods significantly outperform the NN policy in all environments, and so does the resolvent method in all environments except in the "convection" environment; in this last environment, the resolvent method is outperformed by all the other methods. There is no significant difference between the FPCA and naive functional methods. We provide all the pairwise comparisons in Annex C.

To answer our initial questions:

1. Our proposed methods do seem to work for PDE control; as policies are updated, we do observe an increase in the episodic rewards. After training, all our methods perform significantly better than the zero-controller.

2. All of our functional methods perform similarly, but using the PPO algorithm yields similar or better performances than TRPO, which is coherent with the literature (Schulman et al., 2017).

3. We observed that using FLPs instead of a MLP parametrized policy yielded significantly better performances in these environments.

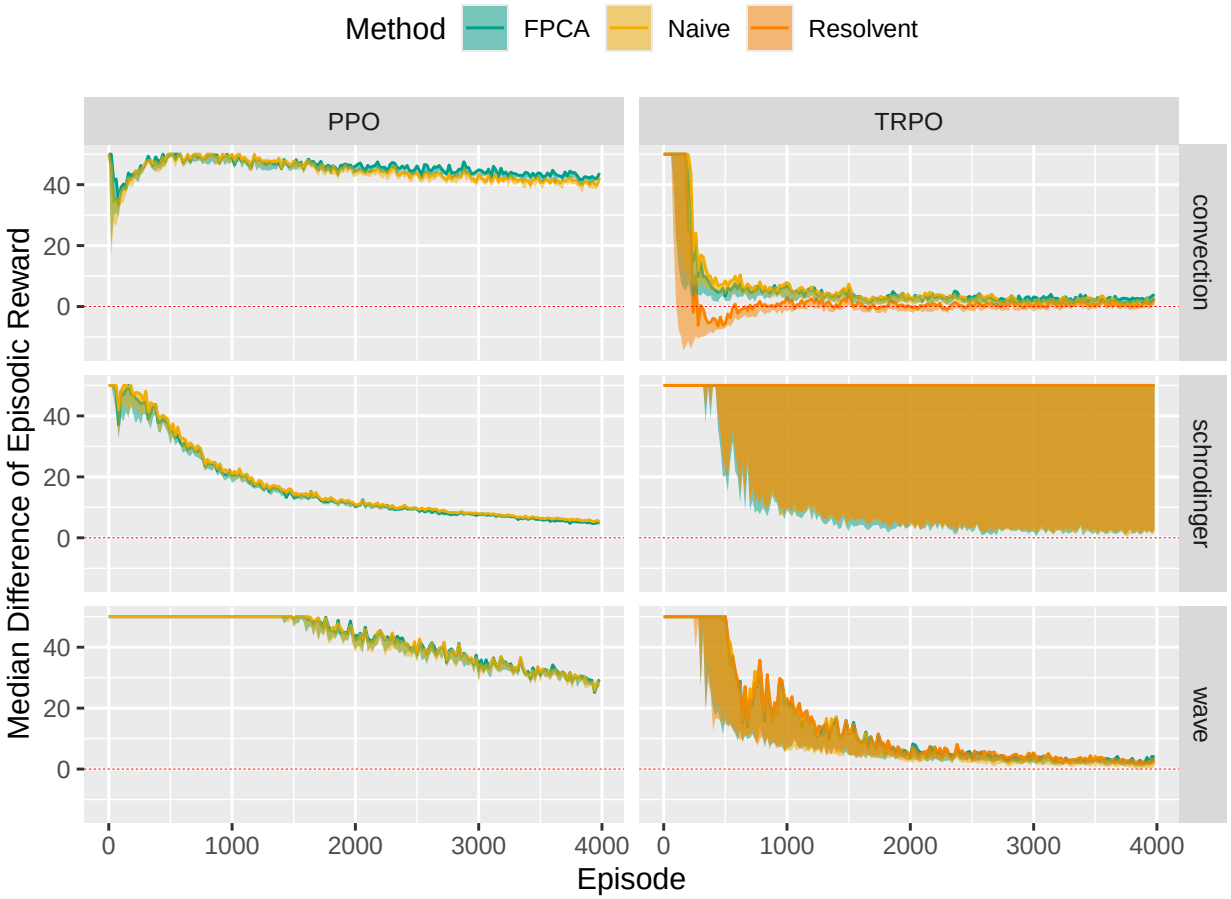

Figure 2: Difference Between the Median Episodic Reward of FLPs and Neural-Network Policy vs. Episode.

4 Limitations

While our theoretical framework and algorithmic developments are general, the numerical experiments in this paper were carried out only on three environments. Indeed, we focused on canonical PDE control environments as a first testbed, which already present significant challenges while offering explicit, interpretable baselines. In this practical setting, we proposed a class of policies taking actions depending on a functional state and proved that TRPO succeeds in improving such policies, whereas standard NN-parameterized policies did not yield competitive performance in this setting. Extending to broader application domains is an exciting future direction.

In our experiments we used an MLP-parameterized policy, as these are the classic choice in the RL literature. While such architecture does not encode spatial structure, it provides a standard baseline for comparison. Exploring architectures better suited to functional states may further improve performance, but this lies outside the scope of our focus on FLPs.

In this paper, we considered linear functional methods; these may not be as expressive as deep RL methods; nevertheless, an immediate extension allowing for more expressive policies is using functional additive models (McLean et al., 2014). We considered the scenario of a univariate functional state; when the state is a multivariate functional space, using the signature representation (Kidger et al., 2019; Cugliari et al., 2025) may prove more effective than our functional approach.

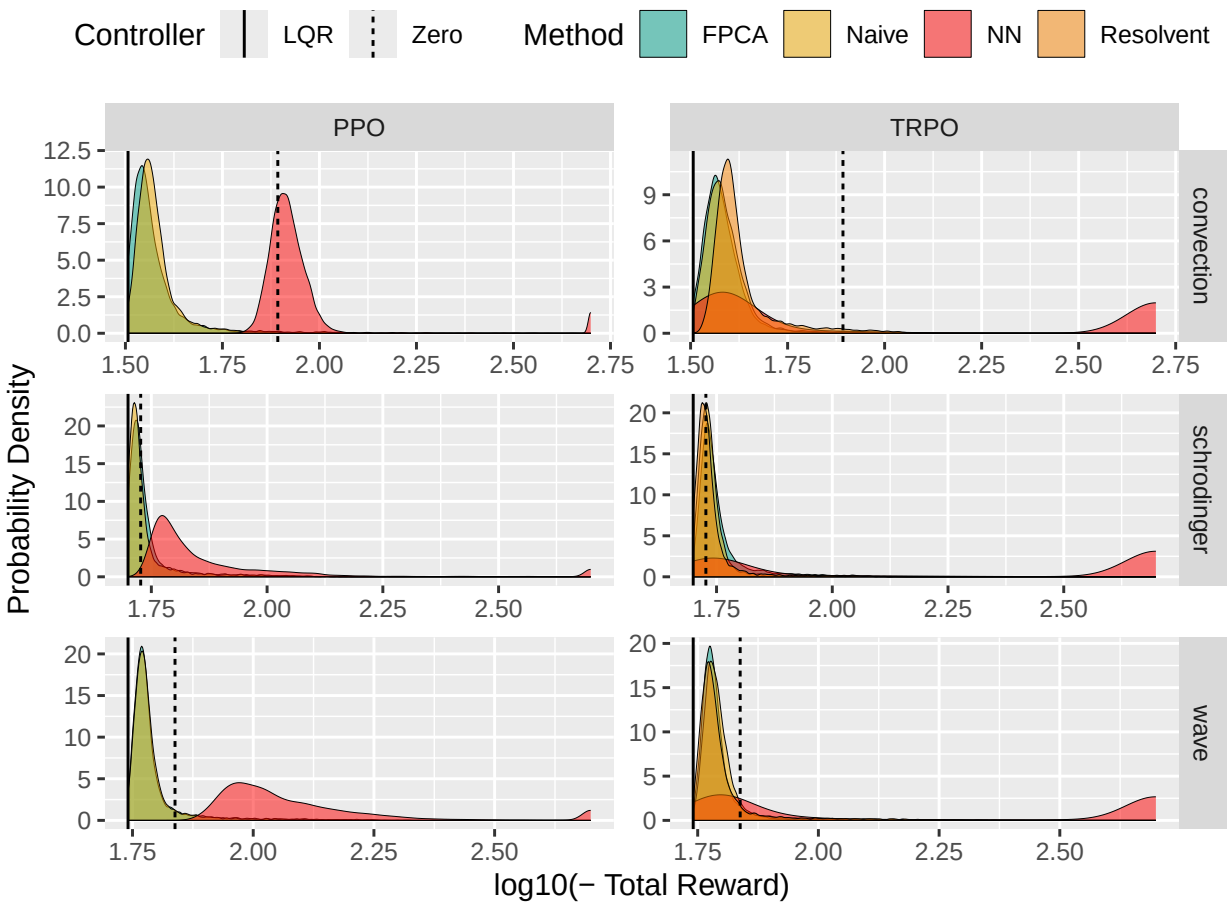

Figure 3: Probability Densities of Final Episodic Reward for Different Algorithms and Methods.

Conclusion and further work

This work introduces a family of policies capable of handling functional state spaces and extends TRPO to provide a practical update algorithm for such policies. These methods offer a promising approach to addressing high-dimensional state representations, particularly in spatial and temporal settings. While this study marks an initial step, it opens several avenues for further development and real-world applications.

Deep reinforcement learning has proven efficient in numerous complex tasks, but RL methods do not necessarily need to use deep learning, and for some applications simpler models may be preferred. For instance, using simpler models may be particularly relevant for embedded or wearable devices, where compute, memory, and battery are limited. Anonymous et al. (Year)[1] illustrate this point with a concrete example: using a functional regression model provides competitive performance while consuming less energy; later, Anonymous et al. (Year)[1] demonstrated that using an RL approach seemed effective for the personalized control of a wearable device. This present work extends RL to account for functional states, filling a gap in the literature and enabling a wide range of applications, such as the personalized control of wearable devices using functional inputs.

A key advantage of using FDA representations is their ability to handle missing values and irregular sampling—common challenges in real-world problems. In contrast, traditional methods such as state stacking struggle with scalability and flexibility.

[1]Removed for anonymous submission

Further generalizations include exploring function-to-scalar policies for discrete or ordinal actions or even function-to-function policies, enabling functional actions, which could be used to approach control problems where the control itself is a function, from an RL standpoint. Notably, the framework of Hernandez-Lerma (2001) accommodates any state and action spaces that are Borel spaces, suggesting broader applicability.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Erick O Arwa and Komla A Folly. Reinforcement learning techniques for optimal power control in grid-connected microgrids: A comprehensive review. *Ieee Access*, 8:208992–209007, 2020.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

Hervé Cardot, Frédéric Ferraty, and Pascal Sarda. Functional linear model. *Statistics & Probability Letters*, 45(1):11–22, 1999. ISSN 0167-7152.

Jairo Cugliari, Emilie Devijver, Anouar Meynaoui, and Raphaël Mignot. Some recent developments on functional data analysis. *ESAIM: Proceedings and Surveys*, 2025.

Amir-massoud Farahmand, Saleh Nabi, and Daniel N. Nikovski. Deep reinforcement learning for partial differential equation control. In *2017 American Control Conference (ACC)*, pp. 3120–3127, 2017. doi: 10.23919/ACC.2017.7963427.

Milton Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701, 1937.

O. Hernandez-Lerma. *Adaptive Markov Control Processes*. Springer-Verlag, Berlin, Heidelberg, 2001. ISBN 0387969667.

Tailen Hsing and Randall Eubank. *Theoretical foundations of functional data analysis, with an introduction to linear operators*, volume 997. John Wiley & Sons, 2015.

Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. The 37 implementation details of proximal policy optimization. *The ICLR Blog Track 2023*, 2022a.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022b. URL `http://jmlr.org/papers/v23/21-1342.html`.

Ying Ji, Jianhui Wang, Jiacan Xu, Xiaoke Fang, and Huaguang Zhang. Real-time energy management of a microgrid using deep reinforcement learning. *Energies*, 12(12), 2019. ISSN 1996-1073. doi: 10.3390/en12122291. URL `https://www.mdpi.com/1996-1073/12/12/2291`.

Sham M Kakade. A natural policy gradient. In T. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

Sham M. Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002.

Patrick Kidger, Patric Bonnier, Imanol Perez Arribas, Cristopher Salvi, and Terry Lyons. Deep signature transforms. *Advances in neural information processing systems*, 32, 2019.

Erwin Kreyszig. *Introductory Functional Analysis with Applications*. Wiley classics library. Wiley India Pvt. Limited, 2007. ISBN 9788126511914.

Guillaume Lample and Devendra Singh Chaplot. Playing fps games with deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pp. 50–60, 1947.

Eduardo Martini, Junoh Jung, André V. G. Cavalieri, Peter Jordan, and Aaron Towne. Resolvent-based tools for optimal estimation and control via the wiener–hopf formalism. *Journal of Fluid Mechanics*, 937, 2022.

Mathew W. McLean, Giles Hooker, Ana-Maria Staicu, Fabian Scheipl, and David Ruppert. Functional generalized additive models. *Journal of Computational and Graphical Statistics*, 23(1):249–269, 2014.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Hans-Georg Müller and Ulrich Stadtmüller. Generalized functional linear models. *The Annals of Statistics*, 33(2):774 – 805, 2005.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Andrew Patterson, Samuel Neumann, Martha White, and Adam White. Empirical design in reinforcement learning. *Journal of Machine Learning Research*, 25(318):1–63, 2024.

Carlos Ramos-Carreño, José L. Torrecilla, Miguel Carbajo Berrocal, Pablo Marcos Manchón, and Alberto Suárez. scikit-fda: A Python Package for Functional Data Analysis. *Journal of Statistical Software*, 109(2): 1–37, May 2024. doi: 10.18637/jss.v109.i02. URL `https://www.jstatsoft.org/article/view/v109i02`.

J. Ramsay and B.W. Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer, 2005. ISBN 9780387400808.

Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):253–279, 2019.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. URL `https://arxiv.org/abs/1312.6120`.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015a. PMLR.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Jane-Ling Wang, Jeng-Min Chiou, and Hans-Georg Müller. Functional data analysis. *Annual Review of Statistics and Its Application*, 3(Volume 3, 2016):257–295, 2016. ISSN 2326-831X.

Xiangyuan Zhang, Weichao Mao, Saviz Mowlavi, Mouhacine Benosman, and Tamer Başar. Controlgym: Large-scale control environments for benchmarking reinforcement learning algorithms. In Alessandro Abate, Mark Cannon, Kostas Margellos, and Antonis Papachristodoulou (eds.), *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*, volume 242 of *Proceedings of Machine Learning Research*, pp. 181–196. PMLR, 15–17 Jul 2024. URL `https://proceedings.mlr.press/v242/zhang24b.html`.

A Appendix: proof of Proposition 1

To prove Proposition 1, we rely on Lemma 6.1 from Kakade & Langford (2002) and Lemma 3 of Schulman et al. (2015a), their original proof still holds in the context of this article. For completeness, we remind them in Lemma 1 and Lemma 2 respectively. The main idea is that the objective function $J(\cdot)$ can be approximated, locally in the current parameter $\tilde{\theta}$ by a surrogate $L_{\tilde{\theta}}(\cdot)$, which can then be improved; the main result of TRPO (Schulman et al., 2015a), bounds the difference between these two.

Lemma 1. *Given two policies parametrized by θ and $\tilde{\theta}$:*

$$J(\tilde{\theta}) = J(\theta) + \mathbb{E}_{\tilde{\theta}}\left(\sum_{t=0}^{\infty} \gamma^t A_\theta(s_t, a_t)\right).$$

Definition 1. *A couple $(\pi, \tilde{\pi})$ of policies are α-coupled if their joint distribution verifies $\mathbb{P}_{a\sim\pi(\cdot|s),\tilde{a}\sim\tilde{\pi}(\cdot|s)}(a \neq \tilde{a}|s) \leq \alpha$, for all $s \in \mathcal{S}$*

Lemma 2. *Let $(\pi, \tilde{\pi})$ be α-coupled, then:*

$$|\mathbb{E}_{s_t\sim\tilde{\pi}}(\bar{A}(s_t)) - \mathbb{E}_{s_t\sim\pi}(\bar{A}(s_t))| \leq 4\alpha(1-(1-\alpha)^t)\sup_{s,a}|A_\pi(s,a)|. \tag{9}$$

Proof of Proposition 1. Let $\theta, \tilde{\theta} \in \Theta$ and $s \in \mathcal{S}$, using the explicit form of the Kullback-Leibler between two random normal distributions we obtain:

$$D_{\mathrm{KL}}(\pi_\theta(\cdot|s)||\pi_{\tilde{\theta}}(\cdot|s)) = \log\left(\frac{\tilde{\sigma}}{\sigma}\right) + \frac{\sigma^2}{2\tilde{\sigma}^2} + \frac{|\langle \beta - \tilde{\beta}, s\rangle_{L^2}|^2}{2\tilde{\sigma}^2} - \frac{1}{2}. \tag{10}$$

Thus, $s \mapsto D_{\mathrm{KL}}(\pi_\theta(\cdot|s)||\pi_{\tilde{\theta}}(\cdot|s))$ is a continuous function, and because $\mathcal{S}$ is bounded, the function $s \mapsto D_{\mathrm{KL}}(\pi_\theta(\cdot|s)||\pi_{\tilde{\theta}}(\cdot|s))$ attains a maximum in $\mathcal{S}$. The function $(s,a) \mapsto |A^{\pi_\theta}(s,a)|$ is bounded because the reward function is bounded. Thus, $\alpha^2 = \max_{s\in\mathcal{S}} D_{\mathrm{KL}}(\pi_{\tilde{\theta}}(\cdot|s)||\pi_\theta(\cdot|s))$ and $\epsilon = \sup_{s,a}|A^{\pi_\theta}(s,a)|$ are finite and well-defined.

The rest of the proof follows the same lines as the one provided in Annex of Schulman et al. (2015a).

Consider $\bar{A}(s) = \mathbb{E}_{a\sim\pi_{\tilde{\theta}}}(A_{\pi_\theta}(s,a))$, by Lemma 1, we can write:

$$J(\tilde{\theta}) = J(\theta) + \mathbb{E}_{\tilde{\theta}}\left(\sum_{t=0}^{\infty}\gamma^t \bar{A}(s_t)\right) \;\; ; \quad L_\theta(\tilde{\theta}) = J(\theta) + \mathbb{E}_\theta\left(\sum_{t=0}^{\infty}\gamma^t \bar{A}(s_t)\right).$$

Let D_{TV} be the total-variation between probability distributions. By Pinsker's inequality, for any state $s \in \mathcal{S}$, we have $D_{\mathrm{TV}}(\pi_\theta(\cdot|s), \pi_{\tilde{\theta}}(\cdot|s))^2 \leq D_{\mathrm{KL}}(\pi_\theta(\cdot|s)||\pi_{\tilde{\theta}}(\cdot|s))$, thus $D_{\mathrm{TV}}(\pi_\theta(\cdot|s), \pi_\theta(\cdot|s)) \leq \alpha$. By (Levin & Peres, 2017, Proposition 4.7), there exists a α-coupling of the policies $(\pi_\theta, \pi_{\tilde{\theta}})$. By Lemma 2, we obtain:

$$|J(\tilde{\theta}) - L_\theta(\tilde{\theta})| = \sum_{t=0}^{\infty}\gamma^t |\mathbb{E}_{s_t\sim\tilde{\pi}}(\bar{A}(s_t)) - \mathbb{E}_{s_t\sim\pi}(\bar{A}(s_t))| \tag{11}$$

$$\leq \sum_{t=0}^{\infty}\gamma^t 4\alpha(1-(1-\alpha)^t)\epsilon \tag{12}$$

$$= \frac{4\epsilon\alpha^2\gamma\epsilon}{(1-\gamma)(1-\gamma(1-\alpha))} \leq \frac{4\gamma\epsilon\alpha^2}{(1-\gamma)^2} \tag{13}$$

□

B Appendix: details of numerical experiments of Section 3

Note that not all of our methods have the same number of hyperparameters; then, to provide a fair comparison, as suggested by Patterson et al. (2024), each method is tuned using only 20 trials in total for the hyperparameter search.

Independently of the method, for the critic training in TRPO and for the PPO update, we used 10 gradient steps, splitting the batch in 4 mini batches.

We show the critic learning rates,γ_{critic}, and maximum Kullback Leibler divergences, C_{KL}, for the TRPO update in Table 1. The range of search for γ_{critic} is [5.0e-05, 5.0e-03] and for C_{KL} is [1.0e-04, 1.0e-01]. As suggested by Patterson et al. (2024), the search was done using the log scale.

Method	wave		schrodinger		convection	
	γ_{critic}	C_{KL}	γ_{critic}	C_{KL}	γ_{critic}	C_{KL}
FPCA	3.1e-04	5.0e-03	3.0e-03	1.8e-02	7.6e-04	9.0e-03
NN	3.1e-03	3.4e-02	4.5e-03	9.9e-02	5.0e-03	5.4e-02
Naive	4.2e-03	3.1e-02	1.0e-03	2.9e-02	4.3e-03	2.7e-02
Resolvent	5.0e-03	1.7e-02	9.3e-04	1.2e-02	2.8e-04	7.1e-04

Table 1: TRPO common hyperparameters among methods. Selected critic learning rates, γ_{critic} and maximum Kullback Leibler divergences, C_{KL}, for each method in each environment

Concerning the PPO algorithm, we used an entropy coefficient of 0.01, a weight for the critic loss function of 0.5 and clipped the gradient, so its norm is at max 0.5. We only tuned the learning rate, γ_{PPO}. We present the selected hyperparameters in different environments in Table 2. The search for γ_{PPO} was done in the interval [1.0e-05, 1.0e-01], using a log scale.

Method	wave	schrodinger	convection
FPCA	5.6e-03	2.6e-03	5.0e-03
NN	7.6e-04	9.9e-04	1.7e-04
Naive	1.7e-02	3.3e-03	2.4e-02

Table 2: PPO common hyperparameters among methods. Selected learning rate γ_{PPO}, for each method in each environment

At last, the FPCA and resolvent methods have additional tunable hyperparameters. When using FPCA, we determine the number of eigenbasis used, by dropping those with those with small eigenvalues; concretely, we select as many components as necessary to explain a certain proportion of the observed variance. We tune for this percentage % Var. When using the resolvent method, we need to tune for the degree of penalization p and a scalar α. We present the selected hyperparameters in Table 3. The range of search for the α hyperparameter was the interval [1.0e-03,1.0e02], and the degree p was searched in the set $\{1, 2, 3\}$, and the percentage of explained variance in TRPO was searched in the set $\{90\%, 95\%, 99\%, 99.9\%\}$.

C Appendix: pairwise comparison of policies at the end of training

As a complement to Figures 1, 2, and 3, in this annex, we provide a thorough comparison of the model's final performance in different environments. These comparisons do not replace the figures; rather, they validate that we are indeed observing significant differences among methods instead of just noise.

To provide a rigorous statistical comparison among methods, we use non-parametric statistical tests: the final reward is not normally distributed and is bimodal for the NN policy when using TRPO. Concretely,

Environment	FPCA % Var		Resolvent	
	PPO	TRPO	α	p
wave	99.9%	99%	2.5e-01	1
schrodinger	99%	95%	1.3e-2	1
convection	90%	99.9%	4.9e-2	2

Table 3: Additional hyperparameters selected for the FPCA method, using either PPO or TRPO and hyperparameters for the resolvent method, in each environment

for a given environment and algorithm (PPO or TRPO), we compare all the different methods, first using a Friedman test (Friedman, 1937), to see if there is any significant difference among methods and. And if there are, we proceed to make pairwise comparisons among all the methods using Wilcoxon's signed-rank test (Mann & Whitney, 1947). For completeness, we also provide the p-values obtained without supposing that the measurements are paired. For instance, we considered three methods when using the PPO algorithm, so three pairwise comparisons are done on the same data, and we considered five methods when using the TRPO algorithm, so ten comparisons are done on the same data. Because we are doing multiple comparisons on the same data, we correct the obtained p-values using the Benjamini-Hochberg procedure (Benjamini & Hochberg, 1995). We present the obtained results in Table 4.

		convection		schrodinger		wave	
Method A	Method B	Paired	Unpaired	Paired	Unpaired	Paired	Unpaired
PPO							
Naive	FPCA	0.025	0.145	0.003	0.078	0.423	0.855
NN	FPCA	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001
NN	Naive	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001
TRPO							
Naive	FPCA	0.334	0.407	< 0.001	< 0.001	1.000	1.000
NN	FPCA	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001
NN	Naive	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001
Resolvent	FPCA	< 0.001	< 0.001	< 0.001	< 0.001	1.000	1.000
Resolvent	Naive	< 0.001	< 0.001	0.297	0.211	1.000	1.000
Resolvent	NN	0.014	0.407	< 0.001	< 0.001	< 0.001	< 0.001

Table 4: Pairwise comparison of final performances, using Wilcoxon's signed-rank test, for different methods, in three environments, using Benjamini-Hochberg correction for multiple testing.

Using the results from Table 4 and Figure 3, when using PPO, we observe that both functional methods outperform the NN policy in all environments. Likewise, the naive and FPCA functional methods significantly outperform the NN policy in all environments, and so does the resolvent method in the Schrödinger and wave environments. The resolvent method is significantly different from the NN policy in the convection environment when considering paired measurements but not when measurements are supposed to be unpaired. When comparing the FPCA and naive functional methods, we observe similar performances in the wave and convection environments and different performances in the Schrödinger environment.