STIEFEL MANIFOLD OPTIMIZATION ACCELERATES REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with high-dimensional observations and overparameterized neural networks is common in practice, yet its learning dynamics remain poorly understood. To develop a theoretically grounded approach to improving learning in this setting, we study learning a feedback controller for the linear-quadratic regulator problem with high-dimensional observations generated from low-dimensional latent states. We use deep linear networks to model overparameterized policies and value functions, with weight matrices constrained to have orthonormal columns, i.e., to lie on the Stiefel manifold. Within this framework, we show that the learning dynamics of the high-dimensional optimization problem match those of the underlying low-dimensional latent-state problem, allowing actor-critic learning to exploit the hidden low-dimensional structure. Motivated by this theory, we improve the performance of proximal policy optimization by incorporating Stiefel-manifold optimization in MuJoCo environments with pixel observations, trained end-to-end from reward and without auxiliary representation-learning losses. We further show that the same optimization principle improves large language model fine-tuning using distillation for the Qwen3-4B model.

1 INTRODUCTION

In reinforcement learning (RL), an agent makes an observation, takes an action, and obtains a reward. The agent learns to maximize cumulative discounted rewards in an environment through repeated interaction (Sutton & Barto, 1998). Learning to control an agent using RL is a natural idea and has been applied to robotic control from high-dimensional sensor input like pixels (Levine et al., 2016; Yarats et al., 2021; Kaufmann et al., 2023) where the agent is equipped with neural network function approximators. An important aspect of learning algorithms is the choice of optimizer, which determines how the parameters of neural networks change over a learning step given the direction of steepest ascent in the objective. RL has also been applied to training large language model (LLM) agents that generate text (Ouyang et al., 2022; Shao et al., 2024). In LLM-based RL, there has been significant progress in theoretical analysis and application of various new optimizers (Gupta et al., 2018; Jordan et al., 2024; Vyas et al., 2024; Liu et al., 2025; Bernstein, 2025a). These methods improve performance by treating the weights of neural networks as matrices rather than as vectors. Such theoretically grounded results for better optimizers are largely missing in robotic control, and most work relies on the Adam optimizer (Kingma & Ba, 2015). Therefore, improving optimization for RL with high-dimensional observations requires theoretically tractable models which are applicable to neural networks.

In supervised learning, deep neural networks are simplified as deep linear networks (Saxe et al., 2014; Arora et al., 2018; Laurent & Brecht, 2018; Arora et al., 2019; Menon, 2023) (DLNs) to develop theoretical understanding. More recently, DLNs have been used to study optimization algorithms (Dragutinović et al., 2026). Similarly, in RL, complex algorithms are analyzed with linear transitions and linear noise, namely the Linear-quadratic regulator (LQR) and Linear–quadratic Gaussian (LQG) environments (Parr et al., 2008; Fazel et al., 2018; Yang et al., 2019; Zhou & Lu, 2023; de Oliveira et al., 2024a;b) due to their tractability. Another factor in pixel-based control is that the high-dimensional observations have a low-dimensional underlying structure (Tenenbaum et al., 2000; Tiwari & Konidaris, 2022). For example, a picture of a humanoid robot taken from a fixed camera can specify its pose. This pose is comprised of very few variables as opposed to the multi-thousand-dimensional pixels of the image (Figure 1). We combine all three ideas and model an

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

RL agent with DLN function approximators that learns in linear environments with low-dimensional states and high-dimensional observations to develop a tractable theoretical model of RL that has broader applicability.

We consider a gradient update rule that restricts the matrices to the Stiefel manifold, meaning the set of all matrices that have mutually orthonormal columns (Wen & Yin, 2013; Bernstein, 2025a). The agent learns by taking the vanilla gradient steps for each parameter matrix in the actor and critic and retracting them back to the Stiefel manifold. By combining our theoretical model with constrained updates of parameter matrices on the Stiefel manifold, we see a surprising structure emerge in the one-step gradient update, wherein the learning dynamics can be described using low-dimensional matrices corresponding to the underlying low-dimensional states. In other words, there is a low-dimensional interpretation of the learning dynamics of high-dimensional neural networks with high-dimensional observations. To the best of our knowledge, this is the first analysis of its kind for reinforcement learning within the actor–critic framework with high-dimensional observations. Our theory, to the best of our knowledge, is also the first to motivate the use of the *manifold Muon* algorithm (Bernstein, 2025b;a), a practical Stiefel manifold optimization algorithm, in RL. Empirically, we validate our theory by demonstrating that learning an LQR on high-dimensional observations on the Stiefel manifold mirrors learning directly on the underlying latent state space. When an agent uses manifold Muon to learn an optimal policy for the pixel-based inverted pendulum task, its true underlying latent states can be recovered from the pixel-based embeddings that it learns. We show that, when paired with an appropriate ISTA encoder (Yu et al., 2023a) as suggested by our theory, manifold Muon successfully learns complex continuous control policies for high-dimensional environments such as Ant and Humanoid. We therefore show optimizer-architecture co-design leads to improved performance. Fine-tuning of LLMs using a student-teacher policy distillation framework also has a similar structure: it has very large state and action spaces, but the token embeddings and intermediate representations in deep networks have a low-dimensional underlying structure to them (Yin & Shen, 2018; Aghajanyan et al., 2021). Finally, we show that a "factored" manifold Muon scales and outperforms Adam for fine-tuning the Qwen3-4B LLM (Yang et al., 2025).

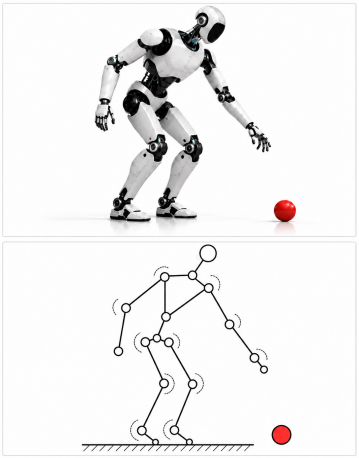


Figure 1: Suppose the agent's goal is to control a humanoid to pick up a ball. Top: a picture of the humanoid robot (upstairs). Bottom: a schematic of its pose denoting the underlying structure (downstairs) [1].

2 Gradient Updates with Retraction to Stiefel Manifold

At their core, deep neural networks are a composition of matrix multiplications interposed by non-linearities. A deep RL agent learns by updating its parameters, the weights of the matrices, by taking gradient ascent steps in the direction of increasing its objective. The Muon optimizer (Jordan et al., 2024), which treats these matrices as operators rather than flattened vectors, has shown improved empirical performance over Adam in supervised learning settings. Another relevant extension of this approach is to restrict the matrices to the *Stiefel manifold*: $\mathrm{St}(d,k)$ (Absil et al., 2008; Wen & Yin, 2013; Bernstein, 2025a), the set of matrices in $\mathbb{R}^{d\times k}$ with orthonormal columns: $\mathrm{St}(d,k) := \{X \in \mathbb{R}^{d\times k} : X^\top X = I_k\}$. In general, a standard Euclidean gradient update does not preserve this constraint, since it discards the matrix structure by treating the parameters as a vector. Therefore, after taking an update in Euclidean space, we retract the matrix back onto the manifold. Let $X \in \mathbb{R}^{d\times d}$ be any square parameter matrix constrained to the Stiefel manifold and let D_X denote its Euclidean gradient update direction. Then, the Stiefel manifold update with learning rate η is:

$$\Omega_X := D_X X^\top - X D_X^\top, \quad X^+ = \left(I_d + \frac{\eta}{2}\Omega_X\right)^{-1}\left(I_d - \frac{\eta}{2}\Omega_X\right)X, \tag{1}$$

where Ω_X is a skew-symmetric matrix representing the tangent direction and the superscript $+$ denotes the updated parameter matrix after one gradient step with retraction (Absil et al., 2008,

[1] Image generated by ChatGPT.

Chapter 4). Therefore, each of the parameter matrices belongs to the Stiefel manifold throughout the training process, given that they are initialized on the manifold (Wen & Yin, 2013; Absil et al., 2008). This line of work has recently been applied to optimization of large language models (Bernstein, 2025b;a). In practice, computing the matrix inversion associated with $(I_d + \frac{\eta}{2}\Omega_X)^{-1}$ scales as $O(d^3)$. When d is large, as is often the case with neural networks, such an inversion becomes impracticable, even on modern hardware. Manifold Muon addresses this limitation by using the following update: $X^+ = \texttt{retract}(X + \Xi_{\text{opt}})$, where Ξ_{opt} is the result of an optimization process (Bernstein, 2025a) and `retract` is the retraction operation to the Stiefel manifold which can be computed efficiently (Jordan et al., 2024; Amsel et al., 2026). This update circumvents the need for the expensive matrix inversion. While our theoretical analysis is for the former retraction, which is theoretically concise, our empirical analysis takes the latter approach. We combine these insights, having a structured matrix and an optimizer that is effective in gradient-based learning, with the underlying low-dimensional structure in RL problems with high-dimensional observations.

3 LQR WITH HIGH-DIMENSIONAL OBSERVATIONS

To develop a theoretically tractable model of control from high-dimensional observations, we introduce continuous-time LQR with linear observations and deep linear networks (DLNs). Continuous-time environments provide a tractable framework for analyzing RL with continuous states and actions (Huang et al., 2025; Feng & Wang, 2025; Tiwari et al., 2026), building on their established use in control theory (Kushner & Dupuis, 2001). The agent observes $o_t \in \mathbb{R}^d$, while the underlying state is $s_t \in \mathbb{R}^{d_s}$, with $d \gg d_s$, and actions belong to $\mathbb{R}^{d_a}$. We refer to the observation-based and latent-state-based formulations as *upstairs* and *downstairs*, respectively. For further background and explanation, see Section D. Let $M = [\mu_1, \ldots, \mu_{d_s}] \in \mathbb{R}^{d \times d_s}$ map states to observations. The observation, exploratory policy, and state dynamics are

$$\begin{aligned} o_t \mathrm{d}t &= M s_t \mathrm{d}t + \epsilon \mathrm{d}b_t, \qquad \pi(o_t)\mathrm{d}t = \phi(o_t; \mathbf{W})\mathrm{d}t + \sigma \mathrm{d}w'_t, \\ \mathrm{d}s_t &= \big(G s_t + H\pi(o_t)\big)\mathrm{d}t + \Sigma \mathrm{d}w_t, \qquad s_0 \text{ fixed}, \end{aligned} \tag{2}$$

Here $G, \Sigma \in \mathbb{R}^{d_s \times d_s}$ and $H \in \mathbb{R}^{d_s \times d_a}$. The mutually independent Wiener processes b_t, w_t, w'_t have dimensions d, d_s, d_a; ϵ and σ control observation and exploration noise. We denote a policy rollout by $(s_t^{\mathbf{W}}, o_t^{\mathbf{W}}, a_t^{\mathbf{W}})$, with $a_t^{\mathbf{W}} = \pi(o_t^{\mathbf{W}})$. For reward $r(s, a) = s^\top Q s + a^\top R a$, with $Q \in \mathbb{R}^{d_s \times d_s}$, $R \in \mathbb{R}^{d_a \times d_a}$, and discount rate $\beta > 0$, the value function and the objective are

$$V(s; \mathbf{W}) := \mathbb{E}\left[\int_0^\infty e^{-\beta t} r(s_t^{\mathbf{W}}, a_t^{\mathbf{W}})\mathrm{d}t \,\middle|\, s_0 = s\right], \tag{3}$$

$$J(\mathbf{W}) := V(s_0; \mathbf{W}). \tag{4}$$

We denote the equivalent downstairs state process by $\tilde{s}_t^{\mathbf{W}}$; background and precise definitions are given in Section A. We also give the simplified downstairs dynamics in Section A.7.

3.1 ACTOR-CRITIC PARAMETERIZATION

We use depth-L DLNs (Saxe et al., 2014; Arora et al., 2018), with $L \geq 2$, and we define the matrix product as $\Pi_{l=1}^L X_l := X_L \cdots X_1$. The policy and value estimate are

$$\phi(o; \mathbf{W}) := \Big(\Pi_{l=1}^L W_l\Big) o, \tag{5}$$

$$\varphi(o; \mathbf{U}) := o^\top \Big(\Pi_{l=1}^L U_l\Big) o + \Big(\Pi_{l=1}^L U'_l\Big) o + c. \tag{6}$$

See Section B for background on why DLNs are a useful theoretical model to study non-linear neural networks. The continuous-time *advantage rate* is $\mathrm{q}(s, a; \mathbf{W}) := \lim_{\Delta t \to 0}[Q_{\Delta t}(s, a; \mathbf{W}) - V(s; \mathbf{W})]/\Delta t$, where $Q_{\Delta t}$ is the return from taking action a for duration Δt and then following the policy (Jia & Zhou, 2023; Cheng et al., 2026). We estimate it using

$$\begin{aligned} \Psi(o, a; \mathbf{Z}) &:= o^\top \Big(\Pi_{l=1}^L Z_l\Big) a + \Big(\Pi_{l=1}^L Z'_l\Big) a, \\ \psi(o, a; \mathbf{Z}) &:= \Psi(o, a; \mathbf{Z}) - \Psi(o, \phi(o; \mathbf{W}); \mathbf{Z}), \end{aligned} \tag{7}$$

so that the estimate vanishes at the deterministic policy action (see Corollary 1). All layer matrices are $d \times d$ except the fixed boundary matrices $W_L \in \mathbb{R}^{d_a \times d}$, $U'_L, Z'_L \in \mathbb{R}^{1 \times d}$, and $Z_1, Z'_1 \in \mathbb{R}^{d \times d_a}$.

The trainable parameter collections are $\mathbf{W} = (W_l)_{l=1}^{L-1}$, $\mathbf{U} = ((U_l)_{l=1}^{L}, (U'_l)_{l=1}^{L-1}, c)$, and $\mathbf{Z} = ((Z_l)_{l=2}^{L}, (Z'_l)_{l=2}^{L-1})$, with $c \in \mathbb{R}$.

Downstairs summary matrices. The dependence of these functions on the underlying state, or their downstairs equivalents, is defined by

$$\widehat{U} := M^\top \Big(\Pi_{l=1}^{L} U_l\Big) M \in \mathbb{R}^{d_s \times d_s}, \quad \widehat{U'} := \Big(\Pi_{l=1}^{L} U'_l\Big) M \in \mathbb{R}^{1 \times d_s},$$
$$\widehat{Z} := M^\top \Big(\Pi_{l=1}^{L} Z_l\Big) \in \mathbb{R}^{d_s \times d_a}, \qquad \widehat{Z'} := \Pi_{l=1}^{L} Z'_l \in \mathbb{R}^{1 \times d_a}, \widehat{W} := \Big(\Pi_{l=1}^{L} W_l\Big) M \in \mathbb{R}^{d_a \times d_s}. \tag{8}$$

Writing $\Theta = (\widehat{U}, \widehat{U'}, \widehat{Z}, \widehat{Z'}, \widehat{W}, c)$, the downstairs functions are

$$\bar{\phi}(s) = \widehat{W} s, \qquad \bar{\varphi}(s) = s^\top \widehat{U} s + \widehat{U'} s + c, \bar{\psi}(s, a) \quad = (s^\top \widehat{Z} + \widehat{Z'})(a - \widehat{W} s), \tag{9}$$

obtained by substituting $o = Ms$, with parameter dependence suppressed. The closed-loop drift matrix is $A_{\mathrm{cl}}(\widehat{W}) := G + H\widehat{W}$.

3.2 Actor-Critic Learning

We apply continuous-time actor-critic updates to observations (Yarats et al., 2021). For bootstrapping duration $\ell > 0$, the temporal-difference (TD) error is

$$\delta_t := \varphi(o_t^{\mathbf{W}}; \mathbf{U}) - e^{-\beta\ell} \varphi(o_{t+\ell}^{\mathbf{W}}; \mathbf{U}) - \int_t^{t+\ell} e^{-\beta(u-t)} \big[r(s_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \psi(o_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \mathbf{Z})\big] \mathrm{d}u. \tag{10}$$

For a trainable matrix X in the corresponding network, the sample-based update directions are

$$\begin{aligned} \mathcal{G}(X) &:= \int_0^\infty e^{-\beta t} \partial_X \varphi(o_t^{\mathbf{W}}; \mathbf{U}) \delta_t \mathrm{d}t, \\ \mathrm{G}(X) &:= \int_0^\infty e^{-\beta t} \partial_X \psi(o_t^{\mathbf{W}}, a_t^{\mathbf{W}}; \mathbf{Z}) \delta_t \mathrm{d}t, \\ \mathbb{G}(X) &:= \int_0^\infty e^{-\beta t} \partial_X \phi(o_t^{\mathbf{W}}; \mathbf{W})\ \partial_a \psi(o_t^{\mathbf{W}}, a; \mathbf{Z})\big|_{a=\phi(o_t^{\mathbf{W}}; \mathbf{W})}\, \mathrm{d}t. \end{aligned} \tag{11}$$

The policy-gradient theorem and the critic estimate update derivations in Section E are novel and provide a complete and concise learning setup for the LQR environment with high-dimensional observations that we have introduced. Although the theorem is based on the results by Cheng et al. (2026), we are the first to systematically prove it for the exploratory setting in LQR. We initialize all trainable matrices on $\mathrm{St}(d, d)$ and apply the retraction in Equation 1 with learning rate η, writing X^+ for an updated matrix and $\mathrm{grad}_{\mathrm{St}}(X; G) := G - XG^\top X$ for the matrix update operator. Replacing observations and approximators by their downstairs counterparts defines the downstairs TD error $\bar{\delta}_t$. Likewise, $\widehat{\mathcal{G}}$, $\widehat{\mathrm{G}}$, and $\widehat{\mathbb{G}}$ denote the downstairs update directions, differentiated with respect to the summary matrices. Their rollout expectations are $\bar{\mathcal{G}}$, $\bar{\mathrm{G}}$, and $\bar{\mathbb{G}}$; the expected direction for c is $\bar{\mathcal{R}}(\Theta) := (1 - e^{-\beta\ell}) \mathbb{E} \int_0^\infty e^{-\beta t} \bar{\delta}_t \mathrm{d}t$.

4 Theoretical Result: Upstairs Downstairs Match

The learning complexity of an agent making high-dimensional observations in the upstairs formulation with over-parameterized neural networks can grow very quickly as $d \to \infty$ (Bartlett et al., 2019; Mhammedi et al., 2023). Theoretically analyzing learning behavior in this limit has given researchers a better understanding of high-dimensional settings (Wang et al., 2017; Paquette et al., 2021) and especially for deep learning (Ben Arous et al., 2022; Goldt et al., 2019). The learning dynamics of an agent boil down to the dynamics of the objective function, evaluated at parameters $\mathbf{W}$, $J(\mathbf{W})$ (Equation equation 4). Here, the learning dynamics are captured by the equation for $\Delta J(\mathbf{W}) = J(\mathbf{W}^+) - J(\mathbf{W})$, which is the change in J over one gradient step or the moment-to-moment change in parameter time. For a learning rate, η, the agent's learning dynamics are described by the sequence: $J(\mathbf{W}_0), J(\mathbf{W}_\eta), J(\mathbf{W}_{2\eta}), \ldots$, where the parameters are indexed by the number of steps times the learning rate η. We distinguish environment time t from *parameter time* $\tau = k\eta$ after k updates; Θ_τ

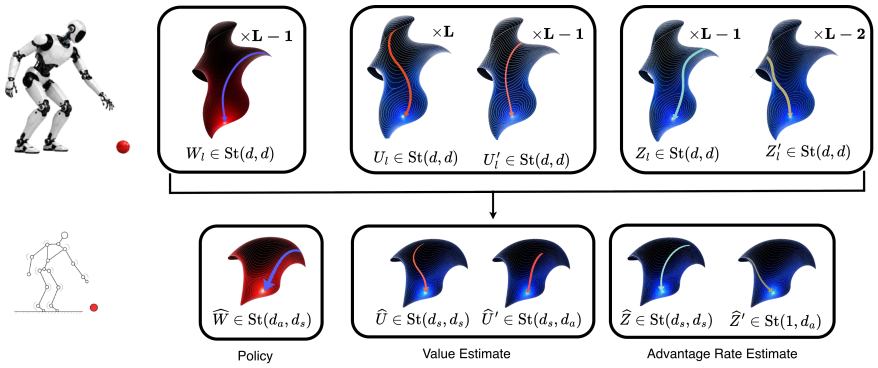


Figure 2: **Upstairs Downstairs Match:** Stiefel optimization learning dynamics of $O(Ld^2)$ parameters can be mapped onto $O(d_s^2 + d_s d_a)$ parameters, proving there is an implicit structure that accelerates learning by making revealing a closed low-dimensional description of learning. The loss and objective over the curved manifolds are denoted using the intensity of color; arrows show the trajectories in parameter time. Manifolds in the downstairs setting are shown to be smaller.

and $J_\tau := J(\mathbf{W}_\tau)$ denote the interpolated summaries and objective. As η approaches 0, the learning dynamics of the agent converge to a process $J_\tau := J(\mathbf{W}_\tau)$ that is continuous in parameter time, with $\tau \in (0, \infty]$. In simpler terms, we study the learning algorithm in parameter time whereas the agent acts in the environment time.

Gradient updates with retraction to the Stiefel manifold (Section 2) for all the trainable parameter matrices lead to a surprising structure: the learning dynamics of an agent in the upstairs formulation can be described using the underlying low-dimensional structure. To derive this mapping between the two formulations, we present the policy, value estimate and advantage rate estimate functions in the downstairs formulation (Equation 8). These are the downstairs analogues of the upstairs matrices parameterizing the agent. Notice that the mapping from state to observations, M, is included in these summary matrices. Together, they summarize the interaction between the high-dimensional parameters and the underlying low-dimensional state and each summary matrix is a function of the upstairs formulation. All the functions in the downstairs formulation, $\bar{\phi}, \bar{\psi}$, and $\bar{\varphi}$ (see Equation equation 13), can be re-written as being parameterized by these matrices. To ensure the well-posedness of the objective, we make the following assumption.

Assumption 1 (Discounted stability). *Let $A_{\mathrm{cl}}(\widehat{W}) := G + H\widehat{W}$, where $\widehat{W}$ is defined in Equation equation 8. We assume that the matrix $A_{\mathrm{cl}}(\widehat{W}) - \frac{\beta}{2} I$ is Hurwitz for all policy parameter matrices on the Stiefel manifold, or equivalently, $\max_{\lambda \in \mathrm{spec}(A_{\mathrm{cl}})} \Re(\lambda) < \frac{\beta}{2}$, where $\Re$ represents the real part and* spec *refers to the spectrum of the matrix and β is the discounting factor.*

Assumption 2 (Orthonormal high-dimensional mapping). *We assume that individual columns of the linear mapping from low-dimensional state to high-dimensional observation are orthonormal, meaning $\mu_i \cdot \mu_{i'} = 0$, for $i \neq i'$ and $||\mu_i||_2 = 1$ for all $i, i' \in \{1, \ldots, d_s\}$.*

The first assumption is standard in LQR because it ensures that the optimal policy matrix exists and can be obtained analytically. The second assumption is key to the following result. Intuitively, it means that each state coordinate affects the observation along a unique direction in observation space, independent of the other coordinates. Thus, changing a single state dimension modifies the observation only along its associated direction μ_i without mixing with the effects of other state dimensions. We also assume that all parameter matrices are initially on the Stiefel manifold, which can be achieved by orthogonal initialization followed by normalization (Hu et al., 2020). Under these assumptions, note that $J(\mathbf{W}) = J(\widehat{W}(\mathbf{W}))$, since the cumulative discounted reward objective simplifies. We also use $\mathrm{grad}_{\mathrm{St}}(X; G)$ for a matrix X on the Stiefel manifold to denote the direction tangent to the manifold given the Euclidean gradient direction G. We state our main theoretical result following the notation in Section 3.1.

Theorem 1 (Informal Upstairs/Downstairs). *Under the assumptions stated above, as $d \to \infty$ and $\eta, \epsilon = O(1/\sqrt{d})$, the upstairs dynamics satisfy $J_\tau = J(\widehat{W}_\tau)$, for $\Theta_\tau = \left\{\widehat{U}_\tau, \widehat{U'}_\tau, \widehat{Z}_\tau, \widehat{Z'}_\tau, \widehat{W}_\tau, c_\tau\right\}$,*

whose parameter time dynamics are determined by the ODE $\dot{\Theta}_\tau = F(\Theta_\tau)$*:*

$$F(\Theta) = \Big(L \operatorname{grad}_{\mathrm{St}}(\widehat{U}; \bar{\mathcal{G}}(\widehat{U})),\ (L-1)\operatorname{grad}_{\mathrm{St}}(\widehat{U'}; \bar{\mathcal{G}}(\widehat{U'})),\ (L-1)\operatorname{grad}_{\mathrm{St}}(\widehat{Z}; \bar{\mathrm{G}}(\widehat{Z})),$$
$$(L-2)\operatorname{grad}_{\mathrm{St}}(\widehat{Z'}; \bar{\mathrm{G}}(\widehat{Z'})),\ (L-1)\operatorname{grad}_{\mathrm{St}}(\widehat{W}; \bar{\mathbb{G}}(\widehat{W})),\ \bar{\mathcal{R}}(\Theta)\Big)^\top,$$

where $\bar{\mathcal{R}}(\Theta) \coloneqq \left(1 - e^{-\beta\ell}\right) \mathbb{E} \int_0^\infty e^{-\beta t} \bar{\delta}_t \, \mathrm{d}t$ *with some initial condition.*

The proof is presented in Section G.2. This result gives a finite-dimensional process for the learning dynamics of the agent in the upstairs formulation in the infinite-dimensional limit. We show that the limiting dynamics of the upstairs agent are the same as a process that remains on the product Stiefel manifold of the downstairs formulation. **Strikingly, this need not be true for vanilla stochastic gradient descent or when Assumption 2 is not true.** While this seems restrictive, it also guides the architecture-optimizer co-design: this suggests that if the agent has an orthogonal mapping from the downstairs formulation then it can use it in conjunction with the optimizer to obtain this upstairs/downstairs match. See Figure 2 for an illustration.

Implicit Acceleration. Theorem 1 identifies an explicit contribution of depth to the learning dynamics in parameter time. In particular, the policy summary matrix evolves according to

$$\frac{\mathrm{d}\widehat{W}_\tau}{\mathrm{d}\tau} = (L-1)\operatorname{grad}_{\mathrm{St}}(\widehat{W}_\tau; \bar{\mathbb{G}}(\widehat{W}_\tau)).$$

The factor $L-1$ arises from the $L-1$ trainable policy matrices, with the output matrix W_L held fixed. As shown in Lemma 8, the Stiefel constraints ensure that the first-order changes induced by these matrices contribute the same downstairs update direction. These contributions add, multiplying the instantaneous rate of change of the policy summary matrix by $L-1$ at a given actor–critic configuration and a fixed learning-rate normalization. For $L > 2$, this amplifies policy movement along the induced update direction, providing a mechanism for implicit acceleration through over-parameterization. The acceleration mechanism is therefore explicit in the limiting update dynamics and does not follow merely from the existence of a low-dimensional description. While Arora et al. (2018) also characterize the dynamics induced by over-parameterization in supervised learning, their end-to-end matrices retain the ambient input dimension d, so that description alone does not guarantee a closed system with finite variables as $d \to \infty$. Our result establishes a finite-dimensional limit for the downstairs actor–critic summary variables, whose dimension depends only on d_s and d_a, for the first time.

5 Empirical Results

We validate our theoretical result in the LQR setting, which exactly matches the theory, albeit with time discretization. We use continuous-time RL as a theoretical framework for studying continuous state and action problems, but in practice we use small time-discretization steps. We show how our model applies to the inverted pendulum from pixels which is LQR-stabilizable, meaning that our theory provides insights even if the mapping from state to observation (M) is a complex non-linear mapping. First, we train an image encoder using the underlying states to approximate $\phi_{\text{enc}}(o) = Ms$, with $M^\top M = I$, and freeze it during policy learning. We also show that if the agent hypothetically has a fixed encoder which maps from low-dimensional states to orthonormal mappings (Assumption 2) the upstairs-downstairs dynamics approximately match for the same learning rate. We also apply the insights from Theorem 1 and the structure of the mapping M to show that we can co-design NNs and optimizers to obtain improved performance in complex control environments. Finally, we show the scalability of our insight by applying our optimizing principle to a 4-billion-parameter LLM (Yang et al., 2025).

5.1 Empirical Validation

We construct an LQR problem which matches our theoretical assumptions with $d = 128$ (observation), $d_s = 8$ (state), and $d_a = 2$ (action) to validate our theory. We evaluate the equivalence between high-dimensional upstairs and low-dimensional downstairs actor-critic learning on a toy LQR system matching Equation 2 with observation noise $\epsilon = 0.05$ and exploration coefficient $\sigma = 0.1$, averaged

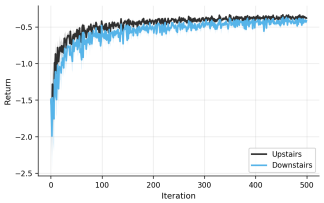


Figure 3: The mean learning curves for an agent learning in the upstairs setting match those of an agent in the downstairs setting.

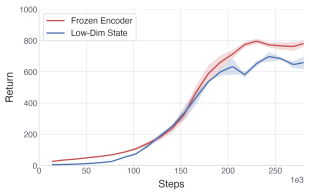


Figure 4: Inverted pendulum return with low-dimensional states (in blue) and with a learned encoder (in red).

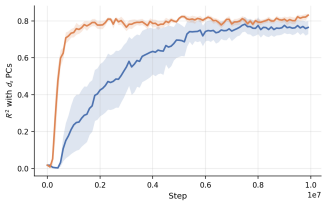


Figure 5: Linear probe R^2, with d_s principal components: manifold muon in red, Adam in blue.

across 16 seeds. The upstairs agent applies depth-3 deep linear actor-critic networks with high-dimensional observations, while the downstairs agent applies summary matrices (as in Equation equation 8) with the low-dimensional state; both are initialized identically and trained with the same martingale loss, learning rate η, and gradient updates (Section 3.2 and Algorithm 1) using gradient descent with a Cayley retraction (Li et al., 2020). We find that the two formulations exhibit nearly identical policy learning dynamics with both converging close to the optimal policy shown in Figure 3, corroborating the result of Theorem 1. We see that upstairs performs slightly better for $\eta = 0.001$; this could be due to the implicit acceleration from over-parameterization (Arora et al., 2018).

5.2 Inverted Pendulum from Pixels

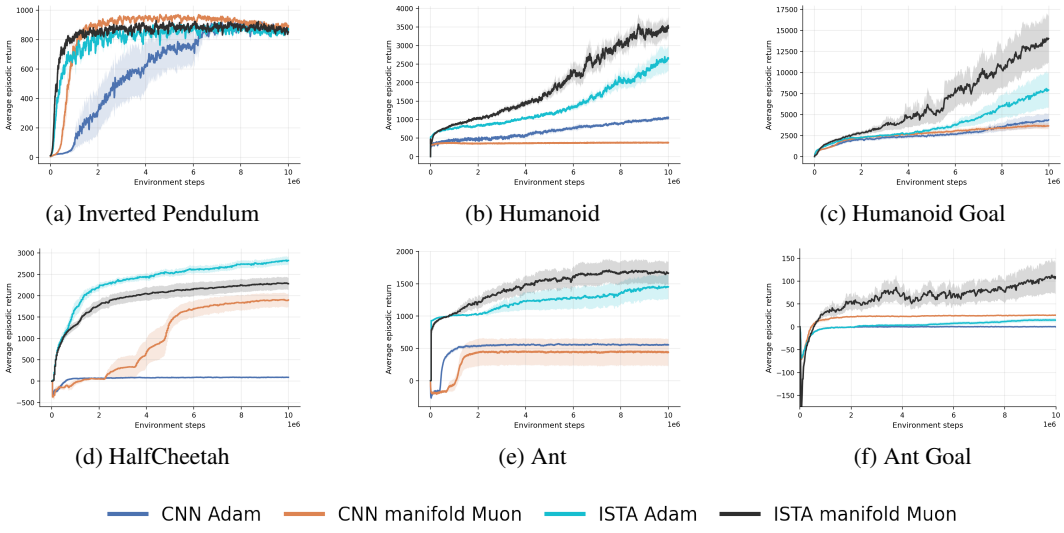
Next, we present a case study of reinforcement learning to stabilize an inverted pendulum, which is LQR-stabilizable, from pixel observations. The mapping $f : \mathbb{R}^{d_s} \to \mathbb{R}^d$, from state to observations, is complex and non-linear. We also choose a practical RL algorithm that is online, episodic and on-policy: PPO (Schulman et al., 2017). We present two studies. First, we learn an encoder from images that maps pixel input to an orthonormal linear mapping from the d_s-dimensional state: $\phi_{\text{enc}}(o) = Ms$. This is the same mapping as in Assumption 2. This encoder is trained using samples collected with knowledge of the low-dimensional underlying state. In Figure 4 we show that given an orthonormal encoder the learning dynamics match those of PPO using low-dimensional states for the same learning rate (5×10^{-4}) consistent with the mechanism identified by Theorem 1. We expand on how the encoder is trained in Section K.3. An RL agent needs to learn such an encoder from pixels while learning a policy without the underlying low-dimensional states. Therefore, we endow the agent with a convolutional neural network (CNN) to learn a dense high-dimensional representation that is input into the actor and critic heads (Figure 9). The CNN uses three convolutional layers followed by a 512-dimensional latent projection, with separate two-layer 256-unit MLP actor and critic heads [2]. As suggested by our theory, we optimize the actor and critic parameter matrices using manifold Muon [3]. In our extended case study, presented in Section K, we find that manifold Muon's orthogonalization biases the CNN encoder towards a low-dimensional representation from which the downstairs state is recoverable by a linear probe (Figure 5). The encoder learns to implicitly linearize the underlying states by learning a mapping from images to a dense representation, thereby giving a structure similar to our observation mapping M. Manifold Muon learns such a representation with fewer samples than Adam, supporting the interpretation that successful upstairs learning benefits from a low-dimensional representation matched linearly to the downstairs state. We also introduce the use of the orthonormal representation learning layer in the next section to make this more effective.

5.3 Complex Control Environments from Pixels

We demonstrate the applicability of our theory by applying a practical optimizer and encoder to complex pixel-based environments that are not LQR-stabilizable and that have relatively complicated state spaces. We design an encoder to learn a locally linear, *sparse*, low-dimensional mapping from

[2] We provide the full implementation details in Section I.

[3] Manifold Muon is approximately 1.35 times as expensive as Adam to compute; however, more efficient implementations are an on-going area of research (Buchanan, 2025).



(a) Inverted Pendulum (b) Humanoid (c) Humanoid Goal

(d) HalfCheetah (e) Ant (f) Ant Goal

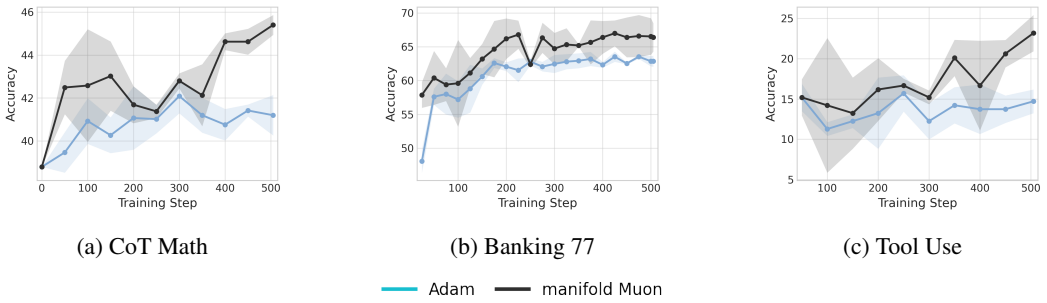
Figure 6: RL control performance across locomotion and goal-reaching tasks. Curves show mean performance with standard-error bands over 6 seeds. Higher is better.

pixels. In a series of works Chan et al. (2022); Yu et al. (2023b;c); Pai et al. (2024), researchers have argued for a better design of neural networks that compress and transform high-dimensional data. They assumed that the data locally has low-dimensional underlying latent structure. They view learning as disentangling representations across different latent dimensions via sparsification. Their approach is similar to what our theory suggests: map each underlying latent state along a unique direction given pixel-based input. The sparsification layer, known as an ISTA layer (Yu et al., 2023b), of width n is defined as Swish $\left(Z - \alpha W^\top (WZ - Z)\right)$, where α is the *sparse rate step size* parameter, Z is the input to the layer, W is the $n \times n$ weight matrix. Our ISTA variant uses a two-layer convolutional encoder followed by a high-dimensional flattening operation, a 512-dimensional linear projection, and a 512-dimensional ISTA sparsification block. The actor and critic use separate heads each consisting of a 256-unit hidden layer followed by a 256-dimensional ISTA block. While, in general, CNN-manifold Muon outperforms CNN-Adam in these complex environments, we find that also endowing the encoder, actor and critic heads with an ISTA layer dramatically improves performance across the board (Figure 6). Manifold Muon combined with the ISTA-CNN is our most performant model on almost all tasks. Moreover, in Section K.3.1 we show that the ISTA layer learns a near-orthogonal representation as required by Assumption 2 for inverted pendulum from pixels. This indicates that manifold Muon is able to exploit the structure of embeddings since ISTA representations are orthogonal for data points across two manifolds Tiwari et al. (2025). For further explanation of this sparsification layer, refer to Appendix I.3, and for additional experiments please refer to Appendix J. Not only does our approach lead to better cumulative discounted rewards, but we also see low-variance returns, which are a result of structure (Bottou et al., 2018; Gupta et al., 2018).

Computation Requirements: We observe that manifold Muon takes 1.35 times as much compute. For the humanoid environment, the vanilla gradient descent obtains 1532 steps per second whereas manifold Muon reaches 1133 steps per second. This pattern holds across all environments simulated on Nvidia H100 GPUs with 128 parallel environments. While our proposed change is sample-efficient, making it more compute-efficient is a subject for future research (Buchanan, 2025). Despite this, we observe that for environments such as ant goal and humanoid goal our method reaches better optima across the compute budget allocated to the Adam methods.

5.4 Large Language Model Distillation Fine-Tuning

LLMs have been formulated as RL agents (Ouyang et al., 2022). Fine-tuning LLMs, meaning further adapting and training them, on datasets for reasoning has been a prominent application of RL algorithms (Agarwal et al., 2024; Shenfeld et al., 2025; 2026; Chen et al., 2025b). The action set is the set of all possible tokens ($\sim$150k) at every step; a natural way to model this is as



(a) CoT Math (b) Banking 77 (c) Tool Use

Figure 7: LLM fine-tuning accuracy across three tasks. Curves show mean accuracy with standard-error bands over 3 seeds. Higher is better.

a probability distribution. Distillation fine-tuning (DFT) uses the negative of the Kullback–Leibler (KL) divergence of token probabilities between a teacher model and a student model as a reward to be maximized. This has a similar structure: the transformer's internal state space is obtained by mapping from possible token sequences to a high-dimensional dense representation. The manifold Muon optimizer only works if the matrices are already on the Stiefel manifold, but that is not true for an open-weight model such as Qwen 3-4B (Yang et al., 2025). To benefit from the implicit low-dimensional structure of Stiefel-manifold optimization, we use the fixed-rank quotient factorization of a matrix $W \in \mathbb{R}^{n\times k}$ (Mishra et al., 2014). We use a rank-r approximation $W \approx X\Gamma Y^\top$, where X and Y contain the leading r left and right singular vectors, respectively, and Γ contains the corresponding positive singular values. For an orthogonal matrix $O \in \mathbb{R}^{r\times r}$, we write the same approximation as $W \approx X'\Gamma' Y'^\top$, with $X' = XO \in \mathrm{St}(n, r)$, $Y' = YO \in \mathrm{St}(k, r)$, and $\Gamma' = O^\top \Gamma O \in \mathbb{R}^{r\times r}$ symmetric positive definite. We update X' and Y' using manifold Muon with ADMM (Buchanan, 2025), and Γ' using Riemannian gradient updates on the symmetric positive definite manifold. We call this *factored manifold Muon*. We see that an actor trained using the factored manifold Muon optimizer outperforms Adam (Figure 7), showing the acceleration from exploiting the low-dimensional underlying structure. Once again, despite requiring more compute, our optimizer remains sample-efficient, and on CoT Math, our optimizer surpasses within 150 steps the accuracy AdamW achieves after 505 steps. Details are in Section L.

6 Discussion

In summary, we have shown that a simplified theoretical model of the problem introduced in this work is a good setting to study complex empirical problems of RL with high-dimensional observations. Our work introduces a tractable theory that can improve applications for robotic control and LLM fine-tuning by prescribing a new optimizer and also providing a new metaphor for deep RL with high-dimensional observations. We present an LQR learning environment with DLNs which is theoretically precise. We show that an agent learning via Stiefel manifold optimization in the upstairs environment behaves as if it were optimizing in the downstairs formulation. Further, we are also the first to show that in the limit $d \to \infty$ the agent's learning behavior can be described using far fewer variables than there are parameters and data-dimensions. This results in implicit acceleration and stability in learning, due to the structure introduced by Stiefel manifold optimization. We show that theoretical results port over to the far more complex empirical setting, with non-linear functions and high-dimensional observations. The main limitation of our analysis is its strict assumptions. However, our empirical results show that even when these assumptions don't hold, the motivated optimizer-architecture co-design is useful in practice. A possible theoretical extension of this work is to policies with linearized neural networks (Jacot et al., 2018; Lee et al., 2019), which are non-linear in input but linear in parameters, which have been applied to the theoretical study of deep RL (Cai et al., 2019b;a; Gaur et al., 2023; Tiwari et al., 2025; 2026). Another application, we foresee, is better representation learning: how do we learn representations that map from the image to representations (Yarats et al., 2021; Dunion et al., 2023; Garcin et al., 2025) that are mutually orthogonal combinations of vectors corresponding to the underlying low-dimensional state? We also expect that the application of our method would improve the performance of far more complex RL algorithms that are used for robotic control from pixels. Most importantly, we need better optimizers for RL for high-dimensional settings, and our work opens up a new avenue.

AI USE STATEMENT

We used large language models (LLMs) as assistive tools in three limited ways. First, we used them to help write and debug parts of the code for our experiments. Second, we used them to improve the grammar, clarity, and readability of the manuscript. Third, we used them to check our mathematical proofs for errors. All research ideas, the methodology, the experimental design, the analysis, and the conclusions are entirely the authors' own. Every piece of LLM-assisted content, including code, text, and proof feedback, was reviewed and verified by the authors. The authors take full responsibility for the content of this paper.

ETHICS STATEMENT

All authors have read and adhere to the ICLR Code of Ethics. This work does not involve human subjects, and no personally identifiable or sensitive information was collected or used. All datasets used are publicly available and were used in accordance with their respective licenses and terms of use. We are not aware of any direct harmful applications of this work. As with most machine learning methods, however, it could be misused or could reflect biases present in the underlying data, and we encourage responsible use and further evaluation before deployment in high-stakes settings. The authors declare no conflicts of interest or sponsorship that could have influenced the results of this work.

REPRODUCIBILITY STATEMENT

We have made every effort to ensure our results are reproducible. The proposed method is described in Sections of the paper, and the full experimental setup, including model architectures, hyperparameters, training procedures, and evaluation protocols. Complete proofs and the assumptions behind our theoretical results appear in Appendix E and F. Anonymized source code, with scripts to reproduce the main experiments, is included in the supplementary materials and will be released publicly upon publication.

REFERENCES

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pp. 7319–7328, 2021.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pp. 242–252. PMLR, 2019.

Noah Amsel, David Persson, Christopher Musco, and Robert M. Gower. The polar express: Optimal matrix sign methods and their application to the muon algorithm. In *The Fourteenth International Conference on Learning Representations*, 2026.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International conference on machine learning*, pp. 244–253. PMLR, 2018.

Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. A convergence analysis of gradient descent for deep linear neural networks. *ArXiv*, abs/1810.02281, 2019.

Karl Johan Åström. Optimal control of markov processes with incomplete state information i. *Journal of mathematical analysis and applications*, 10:174–205, 1965.

Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

Christian Beck, Martin Hutzenthaler, and Arnulf Jentzen. On nonlinear feynman–kac formulas for viscosity solutions of semilinear parabolic partial differential equations. *Stochastics and Dynamics*, 21(08):2150048, 2021.

Richard Bellman. Dynamic programming. *science*, 153(3731):34–37, 1966.

Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. High-dimensional limit theorems for sgd: Effective dynamics and critical scaling. *Advances in Neural Information Processing Systems*, 35: 25349–25362, 2022.

Jeremy Bernstein. Modular manifolds. *Thinking Machines Lab: Connectionism*, 2025a. doi: 10.64434/tml.20250926.

Jeremy Bernstein. The modula docs, 2025b.

Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific, 2012.

V. S. Borkar and S. P. Meyn. The o.d.e. method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000. doi: 10.1137/S0363012997331639.

Vivek Borkar, Shuhang Chen, Adithya Devraj, Ioannis Kontoyiannis, and Sean Meyn. The ode method for asymptotic statistics in stochastic approximation and reinforcement learning. *arXiv preprint arXiv:2110.14427*, 2021.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Sam Buchanan. A faster manifold muon with ADMM, 2025.

Q. Cai, Zhuoran Yang, Jason Lee, and Zhaoran Wang. Neural temporal-difference and q-learning provably converge to global optima. *arXiv: Learning*, 2019a.

Qi Cai, Zhuoran Yang, Jason D. Lee, and Zhaoran Wang. Neural temporal-difference learning converges to global optima. *Advances in Neural Information Processing Systems*, 32, 2019b. ISSN 1049-5258. Publisher Copyright: © 2019 Neural information processing systems foundation. All rights reserved.; 33rd Annual Conference on Neural Information Processing Systems, NeurIPS 2019 ; Conference date: 08-12-2019 Through 14-12-2019.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pp. 38–45, 2020.

Kwan Ho Ryan Chan, Yaodong Yu, Chong You, Haozhi Qi, John Wright, and Yi Ma. Redunet: A white-box deep network from the principle of maximizing rate reduction. *Journal of machine learning research*, 23(114):1–103, 2022.

Alan Chen, Tejas Kotwal, and Govind Menon. Entropic regularization in the deep linear network. *arXiv preprint arXiv:2512.06137*, 2025a.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting. *arXiv preprint arXiv:2510.18874*, 2025b.

Ziheng Cheng, Xin Guo, and Yufei Zhang. Deterministic policy gradient for reinforcement learning with continuous time and state, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems, 2021. 9, 2021.

Arthur Castello B de Oliveira, Milad Siami, and Eduardo D Sontag. Convergence analysis of gradient flow for overparameterized lqr formulations. *arXiv preprint arXiv:2408.15456*, 2024a.

Arthur Castello B. de Oliveira, Milad Siami, and Eduardo D. Sontag. Remarks on the gradient training of linear neural network based feedback for the lqr problem. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pp. 7846–7852, 2024b.

DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020.

Sara Dragutinović, Yedi Zhang, and Rajesh Ranganath. To use or not to use muon: How simplicity bias in optimizers matters. *arXiv preprint arXiv:2603.00742*, 2026.

Simon S. Du, Akshay Krishnamurthy, Nan Jiang, Alekh Agarwal, Miroslav Dudík, and John Langford. Provably efficient RL with rich observations via latent state decoding. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1665–1674, 2019. URL https://proceedings.mlr.press/v97/du19b.html.

Mhairi Dunion, Trevor McInroe, Kevin Luck, Josiah Hanna, and Stefano Albrecht. Conditional mutual information for disentangled representations in reinforcement learning. *Advances in neural information processing Systems*, 36:80111–80129, 2023.

Jianqing Fan, Zhaoran Wang, Yuchen Xie, and Zhuoran Yang. A theoretical analysis of deep q-learning. In *Learning for dynamics and control*, pp. 486–489. PMLR, 2020.

Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International conference on machine learning*, pp. 1467–1476. PMLR, 2018.

Qi Feng and Gu Wang. Continuous policy and value iteration for stochastic control problems and its convergence. *arXiv preprint arXiv:2506.08121*, 2025.

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation, 2021.

Samuel Garcin, Trevor McInroe, Pablo Samuel Castro, Prakash Panangaden, Christopher G Lucas, David Abel, and Stefano V Albrecht. Studying the interplay between the actor and critic representations in reinforcement learning. *arXiv preprint arXiv:2503.06343*, 2025.

Mudit Gaur, Vaneet Aggarwal, and Mridul Agarwal. On the global convergence of fitted q-iteration with two-layer neural network parametrization. In *International Conference on Machine Learning*, pp. 11013–11049. PMLR, 2023.

Sebastian Goldt, Madhu Advani, Andrew M Saxe, Florent Krzakala, and Lenka Zdeborová. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. *Advances in neural information processing systems*, 32, 2019.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pp. 1842–1850. PMLR, 2018.

Wei Hu, Lechao Xiao, and Jeffrey Pennington. Provable benefit of orthogonal initialization in optimizing deep linear networks. *arXiv preprint arXiv:2001.05992*, 2020.

Yilie Huang, Yanwei Jia, and Xun Yu Zhou. Sublinear regret for a class of continuous-time linear-quadratic reinforcement learning problems. *SIAM Journal on Control and Optimization*, 63(5): 3452–3474, 2025.

Arthur Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *NeurIPS*, 2018.

Yanwei Jia and Xun Yu Zhou. Policy evaluation and temporal-difference learning in continuous time and space: A martingale approach. *Econometrics: Econometric & Statistical Methods - Special Topics eJournal*, 2021.

Yanwei Jia and Xun Yu Zhou. Policy gradient and actor-critic learning in continuous time and space: Theory and algorithms. *Journal of Machine Learning Research*, 23(275):1–50, 2022.

Yanwei Jia and Xun Yu Zhou. q-learning in continuous time. *Journal of Machine Learning Research*, 24(161):1–61, 2023.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on learning theory*, pp. 2137–2143. PMLR, 2020.

Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. *URL https://kellerjordan. github. io/posts/muon*, 6(3):4, 2024.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Ioannis Karatzas and Steven Shreve. *Brownian motion and stochastic calculus*, volume 113. springer, 2014.

Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620 (7976):982–987, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Harold J. Kushner and Paul Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*, volume 24 of *Stochastic Modelling and Applied Probability*. Springer, New York, NY, 2 edition, 2001. ISBN 978-1-4613-0007-6. doi: 10.1007/978-1-4613-0007-6.

Harold J Kushner and G George Yin. *Stochastic approximation and recursive algorithms and applications*. Springer, 2003.

Thomas Laurent and James Brecht. Deep linear networks with arbitrary loss: All local minima are global. In *International conference on machine learning*, pp. 2902–2907. PMLR, 2018.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

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

Jun Li, Li Fuxin, and Sinisa Todorovic. Efficient riemannian optimization on the stiefel manifold via the cayley transform. *arXiv preprint arXiv:2002.01113*, 2020.

Kathryn Lindsey and Govind Menon. Regularization implies balancedness in the deep linear network. *arXiv preprint arXiv:2511.01137*, 2025.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Trevor McInroe and Samuel Garcin. Pixelbrax: Learning continuous control from pixels end-to-end on the gpu. *arXiv preprint arXiv:2502.00021*, 2025.

Govind Menon. The geometry of the deep linear network. In *Symposium on Probability and Stochastic Processes*, pp. 1–47. Springer, 2023.

Zakaria Mhammedi, Dylan J Foster, and Alexander Rakhlin. Representation learning with multi-step inverse kinematics: An efficient and optimal approach to rich-observation rl. In *International Conference on Machine Learning*, pp. 24659–24700. PMLR, 2023.

Bamdev Mishra, Gilles Meyer, Silvere Bonnabel, and Rodolphe Sepulchre. Fixed-rank matrix factorizations and riemannian low-rank optimization. *Computational Statistics*, 29(3):591–621, 2014.

Parikh Neal, Chu Eric, Peleato Borja, and Eckstein Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Druv Pai, Sam Buchanan, Ziyang Wu, Yaodong Yu, and Yi Ma. Masked completion via structured diffusion with white-box transformers. In *The Twelfth International Conference on Learning Representations*, 2024.

Courtney Paquette, Kiwon Lee, Fabian Pedregosa, and Elliot Paquette. Sgd in the large: Average-case analysis, asymptotics, and stepsize criticality. In *Conference on Learning Theory*, pp. 3548–3626. PMLR, 2021.

Ronald Parr, Lihong Li, Gavin Taylor, Christopher Painter-Wakefield, and Michael L Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 752–759, 2008.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *CoRR*, abs/1312.6120, 2014.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less. *arXiv preprint arXiv:2509.04259*, 2025.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

David Silver, Guy Lever, Nicolas Manfred Otto Heess, Thomas Degris, Daan Wierstra, and Martin A. Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014.

Daniel W. Stroock and S. R. Srinivasa Varadhan. *Multidimensional Diffusion Processes*. Classics in Mathematics. Springer, Berlin, Heidelberg, 2006.

Weijie Su, Stephen Boyd, and Emmanuel J Candes. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research*, 17 (153):1–43, 2016.

Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

Russ Tedrake. Underactuated robotics: Learning, planning, and control for efficient and agile machines: Course notes for mit 6.832. *Working draft edition*, 3(4):2, 2009.

Joshua B Tenenbaum, Vin de Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Saket Tiwari and George Konidaris. Effects of data geometry in early deep learning. *Advances in Neural Information Processing Systems*, 35:30099–30113, 2022.

Saket Tiwari, Omer Gottesman, and George Konidaris. Geometry of neural reinforcement learning in continuous state and action spaces. In *The Thirteenth International Conference on Learning Representations*, 2025.

Saket Tiwari, Tejas Kotwal, and George Konidaris. From ticks to flows: Dynamics of neural reinforcement learning in continuous environments. In *The Fourteenth International Conference on Learning Representations*, 2026.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Mujin Kwun, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. Soap: Improving and stabilizing shampoo using adam. *arXiv preprint arXiv:2409.11321*, 2024.

Chuang Wang, Jonathan Mattingly, and Yue M Lu. Scaling limit: Exact and tractable analysis of online learning algorithms with applications to regularized regression and pca. *arXiv preprint arXiv:1712.04332*, 2017.

Haoran Wang, Thaleia Zariphopoulou, and Xun Yu Zhou. Reinforcement learning in continuous time and space: A stochastic control approach. *J. Mach. Learn. Res.*, 21:198:1–198:34, 2020.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1):397–434, 2013.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhuoran Yang, Yongxin Chen, Mingyi Hong, and Zhaoran Wang. Provably global convergence of actor-critic: A case for linear quadratic regulator with ergodic cost. *Advances in neural information processing systems*, 32, 2019.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

Zi Yin and Yuanyuan Shen. On the dimensionality of word embedding. *Advances in neural information processing systems*, 31, 2018.

Yaodong Yu, Sam Buchanan, Druv Pai, Tianzhe Chu, Ziyang Wu, Shengbang Tong, Benjamin Haeffele, and Yi Ma. White-box transformers via sparse rate reduction. *Advances in Neural Information Processing Systems*, 36:9422–9457, 2023a.

Yaodong Yu, Sam Buchanan, Druv Pai, Tianzhe Chu, Ziyang Wu, Shengbang Tong, Benjamin David Haeffele, and Yi Ma. White-box transformers via sparse rate reduction. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b.

Yaodong Yu, Tianzhe Chu, Shengbang Tong, Ziyang Wu, Druv Pai, Sam Buchanan, and Yi Ma. Emergence of segmentation with minimalistic white-box transformers. In *Conference on Parsimony and Learning (Proceedings Track)*, 2023c.

Yang Zheng, Luca Furieri, Maryam Kamgarpour, and Na Li. Sample complexity of linear quadratic gaussian (lqg) control for output feedback systems. In *Learning for dynamics and control*, pp. 559–570. PMLR, 2021.

Mo Zhou and Jianfeng Lu. Single timescale actor-critic method to solve the linear quadratic regulator with convergence guarantees. *Journal of Machine Learning Research*, 24(222):1–34, 2023.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, 36: 50117–50143, 2023.

A Background on Continuous Time Stochastic Processes

Continuous time RL requires background on continuous time tochastic processes. For a complete treatment on stochastic processes we refer to the textbooks by Karatzas & Shreve (2014) and Oksendal (2013). Here we succinctly present the main ideas used in our work, including some definitions which are not key but significant to understanding the work. We also refer to the series of work on continuous time RL (Wang et al., 2020; Jia & Zhou, 2021; 2022; 2023) which laid a much more general and theoretically strong foundation of continuous time RL.

A.1 Filtration

To establish the probabilistic framework for continuous-time RL, we briefly recall standard notions from stochastic processes and filtrations. A complete probability space is given by the triple $(\Omega, \mathcal{F}, P)$, where Ω is the sample space, $\mathcal{F}$ is a σ-algebra of events, and P is a probability measure. We consider random variables and stochastic processes defined on this space taking values in $\mathbb{R}^d$.

Let $\eta(t)$, $t \in [0, T)$, be a stochastic process on $(\Omega, \mathcal{F}, P)$. The information revealed up to time t is represented by a family of increasing σ-algebras $\{\mathcal{B}_t\}_{t\in[0,T)}$, called a filtration, satisfying $\mathcal{B}_s \subset \mathcal{B}_t$ for $s < t$. Intuitively, $\mathcal{B}_t$ represents all information available up to time t. Conditional expectation with respect to $\mathcal{B}_t$, denoted $\mathbb{E}[\cdot \mid \mathcal{B}_t]$, represents the best prediction of a random variable given this information. Further background on stochastic processes and filtrations can be found in Oksendal (2013) and Karatzas & Shreve (2014).

All stochastic processes are defined on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ where $(\mathcal{F}_t)_{t\geq 0}$ denotes the usual augmentation of the natural filtration generated by the mutually independent Wiener processes b_t, w_t, and w'_t i.e. $\mathcal{F}_t = \sigma((b_l, w_l, w'_l)_{0\leq l\leq t})^{\text{aug}}$ driving the observation, state, and policy dynamics, respectively.

A.2 Wiener Process

A one-dimensional Wiener process, taking values in the Euclidean space $\mathbb{R}$, is one of the fundamental objects in stochastic process theory (Karatzas & Shreve, 2014; Oksendal, 2013).

Definition 1. *A stochastic process w_t is called a Wiener process if it satisfies the following properties:*

1. $w_0 = 0$,
2. *the sample paths of w_t are almost surely continuous,*
3. *w_t is a square-integrable martingale,*
4. *for any $0 \leq s < t$, the increment $w_t - w_s$ has mean 0 and variance $t - s$.*

An important property of the Wiener process is that it has independent increments: for any sequence of times $0 \leq t_0 < t_1 < \cdots < t_k$, the random variables

$$w_{t_1} - w_{t_0},\ w_{t_2} - w_{t_1},\ \ldots,\ w_{t_k} - w_{t_{k-1}}$$

are independent. Intuitively, this means that the noise added to the system over disjoint time intervals does not depend on the past evolution of the process. Moreover, the increments are *stationary*, meaning the distribution of $w_t - w_s$ depends only on the time difference $t - s$. This makes it ideal for modeling randomness in RL because the current random change, in some sense, does not depend on the past random change in variables.

The multidimensional Wiener process is formed by concatenating independent one-dimensional Wiener processes. Because of its continuous paths and stationary independent increments, the Wiener process provides a natural mathematical model for environmental noise in continuous-time dynamical systems.

A.3 White Noise

In continuous-time reinforcement learning, exploration is commonly modeled by injecting stochastic perturbations into the policy through white noise. Formally, white noise can be interpreted as the generalized time derivative of a Wiener process. If w_t denotes a Wiener process, then the white noise process $\xi(t)$ is defined in a distributional sense as $\xi(t) = \frac{\mathrm{d}w_t}{\mathrm{d}t}$. Although $\xi(t)$ does not exist as an ordinary function, it is well-defined when integrated over time, yielding the increments of the Wiener process. In particular, for any interval $[s, t]$,

$$\int_s^t \xi(\tau)\,\mathrm{d}\tau = w_t - w_s,$$

where the increments $w_t - w_s$ are independent Gaussian random variables with mean 0 and variance $t - s$.

In stochastic control and reinforcement learning, this noise is typically introduced into the dynamics or the policy through stochastic differential equations (SDEs). For example, exploration in the action space may be modeled as

$$a_t = \bar{\phi}(x_t; \mathbf{W}) + \sigma\,\xi(t),$$

where σ controls the exploration magnitude. Equivalently, the resulting system dynamics can be written in Itô form as

$$\mathrm{d}x_t = f(x_t, \bar{\phi}(x_t; \mathbf{W}))\,\mathrm{d}t + \Sigma\,\mathrm{d}w_t,$$

where $\mathrm{d}w_t$ represents Wiener process increments. Because the increments of the Wiener process are independent across time intervals, the injected noise continually perturbs the system without introducing temporal correlations, enabling systematic exploration of the state-action space while preserving tractable stochastic dynamics.

A.4 Square Root of a Matrix

We introduce the notion of square root of a matrix which is used in our work. We use $\tilde{a}_t^{\mathbf{W}}$ to denote the action chosen by the agent $\bar{\phi}(\tilde{s}_t^{\mathbf{W}}; \mathbf{W})$ where $\bar{\phi} : \mathbb{R}^{d_s} \to \mathbb{R}^{d_a}$ is the function mapping from the state to the action and is defined as: $\bar{\phi}(s) = W_L W_{\text{eff}} M s = \Pi_{l=1}^{L-1} W_l M s$. The square root $\widetilde{\Sigma}$ of $\sigma^2 H H^\top + \Sigma\Sigma^\top + \epsilon^2 H \left(\Pi_{l=1}^{L} W_l M\right) \left(\Pi_{l=1}^{L} W_l M\right)^\top H^\top$ (in Section 3.1) is a matrix version of the scalar square root. This square root is well defined under the assumption that $\sigma^2 H H^\top + \Sigma\Sigma^\top$ is positive semidefinite, the principal square root is defined using the eigendecomposition of the positive semidefinite (PSD) matrix:

$$\sigma^2 H H^\top + \Sigma\Sigma^\top + \epsilon^2 H \left(\Pi_{l=1}^{L} W_l M\right) \left(\Pi_{l=1}^{L} W_l M\right)^\top H^\top = E \Lambda E \implies \widetilde{\Sigma} = E \Lambda^{1/2} E^\top,$$

where $E \in \mathbb{R}^{d_s \times d_s}$ is the square matrix whose i-th column is the eigenvector and Λ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues. These matrices are well defined under the assumption of the square matrix being positive semi-definite (PSD).

A.5 Martingale Process

Definition 2. *A d-dimensional stochastic process $\eta(t)$ is called a martingale with respect to a filtration $\{\mathcal{B}_t\}$ and probability measure P if for all $s \leq t$:*

1. *$\eta(t)$ is $\mathcal{B}_t$-measurable,*
2. *$\mathbb{E}[|\eta(t)|] < \infty$,*
3. *$\mathbb{E}[\eta(t) \mid \mathcal{B}_s] = \eta(s)$.*

A.6 Frobenius Inner Product

The Frobenius inner product provides a natural way to define an inner product between matrices by treating them as vectors in Euclidean space. For two matrices $A, B \in \mathbb{R}^{m \times n}$, the Frobenius inner product is defined as

$$A \circ B := \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} B_{ij} = \operatorname{tr}(A^\top B).$$

A.7 Simplified Downstairs Environment Dynamics

There are three independent sources of noise in the upstairs formulation: transition noise w_t, exploration noise w'_t, and observation noise b_t. Any theoretical analysis with three independent sources of noise can prove to be cumbersome. Formulating and analyzing the underlying state dynamics requires further simplification. To do so we reformulate the SDE (Equation equation 2), the solution to the following SDE has the same pathwise distribution to the process with all three sources of noise:

$$\mathrm{d}\tilde{s}_t^{\mathbf{W}} = (G + H\bar{\phi}(\tilde{s}_t^{\mathbf{W}}; \mathbf{W}))\mathrm{d}t + \Sigma(\mathbf{W})\mathrm{d}w_t, \tag{12}$$

where $\Sigma(\mathbf{W})$ is the principal square root of the matrix:

$$\sigma^2 H H^\top + \Sigma\Sigma^\top + \epsilon^2 H \left(\Pi_{l=1}^{L} W_l M\right) \left(\Pi_{l=1}^{L} W_l M\right)^\top H^\top,$$

which is well defined because of the matrix being PSD, $\bar{\phi}$ is the downstairs policy, and $M = [\mu_i]_{i=1,\ldots d_s}$, as described in the previous section. Here, the covariance term Σ is the covariance in the reformulated dynamics where the noise from the observation (b_t) has been subsumed while matching the pathwise distribution in addition to the noise from exploration (w'_t). Note that we abuse notation by reusing the expression for $\tilde{s}_t^{\mathbf{W}}$ for these new dynamics that have both exploration and observation noise. This stochastic process is defined on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ that satisfies the usual conditions, where the filtration $\mathcal{F}_t$ is the natural filtration generated by $\mathcal{F}_t^w$. This once again follows from Lemma 3.1 by Tiwari et al. (2026). We also denote by ξ_t the white noise from the observation. Note that the SDE introduced by us is non-degenerate for deterministic environment as opposed to the relaxed control formulation used by Cheng et al. (2026) (see section 7.1 by Tiwari et al. (2026) for explanation).

B Deep Linear Networks and Theory of Deep Neural Networks

We use deep linear networks (DLNs) as a tractable model of the composition of operators and over-parameterization in deep neural networks (DNNs). A DLN retains the layered matrix parameterization of a neural network while replacing its activation functions by the identity (Saxe et al., 2014; Arora et al., 2018). This allows us to study how the geometry and updates of individual parameter matrices affect learning.

Linear functions with nonlinear learning dynamics. For the policy in Equation equation 5, the input-output map is linear, but it depends on the trainable matrices through their product. It is therefore nonlinear in the collection of parameters: the derivative with respect to one layer depends on the matrices before and after it. Updating these factors separately generally induces a different update of the product from directly optimizing a single matrix. This distinction allows DLNs to exhibit nontrivial learning dynamics despite their linear input-output map. For example, Saxe et al.

(2014) analyze plateaus and transitions in learning, while Arora et al. (2018) show that depth can induce preconditioning and accelerate optimization in particular settings. DLNs also permit analyses of optimization landscapes and convergence (Laurent & Brecht, 2018; Arora et al., 2019).

Over-parameterization and matrix geometry. In our actor, the $L-1$ trainable hidden matrices each have shape $d \times d$, while the fixed output matrix has shape $d_a \times d$. The parameterization therefore uses $(L-1)d^2$ trainable matrix entries to represent an input-output map of shape $d_a \times d$. Different factorizations can represent the same map; changes of basis in hidden layers provide a simple example of this redundancy. The geometry associated with these factorizations has been studied through invariant manifolds and induced dynamics of the product (Menon, 2023). Recent work further studies entropic regularization, balancedness, and optimizer-dependent learning dynamics in DLNs (Chen et al., 2025a; Lindsey & Menon, 2025; Dragutinović et al., 2026). These analyses motivate retaining the factorization when studying an optimizer that treats parameters as matrices.

Connection to nonlinear networks. The connection can also be seen through the input-output Jacobian. For a bias-free network with elementwise activation Φ,

$$F(o) = W_L \Phi\big(W_{L-1} \cdots \Phi(W_1 o)\big), \qquad J_o F = W_L D_{L-1}(o) W_{L-1} \cdots D_1(o) W_1,$$

where $D_l(o)$ is the diagonal matrix of activation derivatives evaluated at the preactivations of layer l, wherever these derivatives exist. For a DLN, $D_l(o) = I_d$, so the Jacobian is a product of the parameter matrices alone. For a ReLU network, the activation matrices are constant within a region with a fixed activation pattern, giving a local product of weight matrices and fixed gates. Thus, DLNs isolate the matrix-composition component of a nonlinear network's local behavior. The gates can change across inputs and during training, so the DLN analysis does not establish identical global learning dynamics for a nonlinear network.

Why constrain the trainable matrices? We take the square trainable matrices to lie on $\mathrm{St}(d, d)$, the orthogonal group, and keep the rectangular boundary matrices fixed. Freezing an output layer is also used in analyses of over-parameterized neural networks and neural RL (Allen-Zhu et al., 2019; Cai et al., 2019b). In our model, products of the square trainable matrices remain orthogonal, preventing those products from expanding or contracting Euclidean norms. This structure simplifies the induced updates of the actor and critic products and allows us to examine their interaction with the observation map M. The fixed boundary matrices still affect the input-output map, and activation derivatives can affect conditioning in a nonlinear network. The constraints also restrict the representable functions; the analysis concerns this particular policy and critic class.

For the critic, the matrix products parameterize the coefficients of quadratic and bilinear functions in Equation equation 6 and Equation equation 7. Thus, using DLN parameterizations does not require the value estimate itself to be linear in observations. Together, the linear environment, observation map, and matrix factorizations isolate how an optimizer can exploit latent structure during actor-critic learning. The nonlinear experiments evaluate how useful the resulting optimization principle remains beyond this tractable model.

C Notation Summary

We summarize all the symbols and formulae used in our work by three categories: **1)** stochastic processes (Table 1), **2)** LQR problem definition (Tables 2 and 3), and **3)** RL agent functions and matrices (Table 4). All the function approximators defined in Section 3.1 are in the upstairs formulation. Therefore, the policy ϕ, the value estimate φ, and the advantage rate estimate ψ also have their downstairs analogs:

$$\begin{aligned} \bar{\phi}(s; \mathbf{W}) :=& \Pi_{l=1}^{L} W_l M s, \quad \bar{\varphi}(s; \mathbf{U}) := x M^\top \Pi_{l=1}^{L} U_l M s + \Pi_{l=1}^{L} U_l' M s + c, \\ \text{and } \bar{\psi}(s, a; \mathbf{Z}) :=& \bar{\Psi}(s, a; \mathbf{Z}) - \bar{\Psi}(s, \bar{\phi}(s; \mathbf{W}); \mathbf{Z}), \end{aligned} \tag{13}$$

where $\bar{\Psi}(s, a; \mathbf{Z}) := x^\top M^\top \Pi_{i=1}^{L} Z_l a + \Pi_{i=1}^{L} Z_i' a$, for $s \in \mathbb{R}^{d_s}$ and $a \in \mathbb{R}^{d_a}$. These functions are the "factored out" versions of the state-dependent approximators given the observation-dependent functions; therefore, we have the M in these functions. Our work provides a connection between these two formulations and shows how learning in the upstairs formulation, for the LQR problem

Table 1: Notation: probability spaces, filtrations, and stochastic processes.

Symbol	Formula	Note
Ω	–	Sample space.
$\mathcal{F}$	–	σ-algebra on Ω.
$\mathbb{P}$	–	Probability measure defined on $(\Omega, \mathcal{F})$.
$\mathcal{F}_{t\geq 0}$	$\sigma((b_\ell, w_\ell, w'_\ell)_{0\leq \ell\leq t})^{\text{aug}}$	Natural filtration generated by the driving Wiener processes with usual augmentation.
$\mathbb{E}[\cdot \mid \mathcal{F}_t]$	–	Conditional expectation with respect to the signm algebra $\mathcal{F}_t$.
w_t	–	d_s-dimensional Wiener process representing transition noise.
w'_t	–	d_a-dimensional Wiener process representing exploration noise.
b_t	–	d-dimensional Wiener process representing observation noise.
ξ_t	–	White noise, defined formally as the time derivative of a Wiener process.
$A \circ B$	$\text{tr}(A^\top B)$	Frobenius inner product used to define matrix inner products and gradients.

stated above, maps to learning in the downstairs formulation with optimizers that retain structure of the matrices. We use these function approximations, as downstairs formulation, in the proofs that follow. The summarized policy matrix of matrices that are trainable is defined as $W_{\text{eff}} = \Pi_{l=1}^{L-1} W_l = W_L W_{L-1} \dots W_1$ for succintness. The summarized matrices for the value estimate, that are trainable and square, resulting from the multiplication are defined as: $U_{\text{eff}} = \Pi_{l=1}^{L} U_l = U_L U_{L-1} \dots U_1$ and similarly for the linear component of the value estimate function approximator we have $U'_{\text{eff}} = \Pi_{l=1}^{L-1} U'_l = U'_{L-1} \dots U'_1$. Similar to above, we denote by $Z_{\text{eff}} := \Pi_{l=2}^{L-1} Z_l$, $Z'_{\text{eff}} := \Pi_{l=2}^{L-1} Z'_l$, the effective matrices for the advantage rate function.

D Upstairs and Downstairs Formulation of the Environment

A model of control from pixels. An important aspect of control from high-dimensional observations is that the underlying environment can have a much lower-dimensional state. For example, a humanoid simulator evolves joint positions and velocities, while an agent controlling the humanoid from pixels receives an image rendered from the simulation (Todorov et al., 2012). The dimension of this image can be much larger than the number of variables that determine the dynamics. Our linear observation model provides a tractable version of this setting: the state $s_t \in \mathbb{R}^{d_s}$ determines the environment dynamics, while the agent receives an observation in $\mathbb{R}^d$, with $d \gg d_s$. The columns μ_i of M describe how individual latent state coordinates contribute to the observation.

Upstairs and downstairs. The downstairs formulation describes the controlled Markov process in the latent state space. In discrete time, this corresponds to the usual Markov decision process (MDP) formulation (Bellman, 1966). The upstairs formulation describes the information available to the agent, which acts through a policy defined on observations. When an observation does not determine the underlying state, this is a partially observable setting (Åström, 1965; Kaelbling et al., 1998). In particular, the observation process need not itself be Markov, and our observation-based policy is a memoryless policy within this partially observable problem. The two formulations describe the same physical environment; they differ in the variables available to the controller and used to parameterize its functions. Access to the downstairs state is used for analysis and is not assumed for the upstairs learning algorithm.

Continuous-time formulation. Continuous-time environments provide a tractable framework for analyzing RL with continuous states and actions (Huang et al., 2025; Feng & Wang, 2025; Tiwari et al., 2026), building on their established use in control theory (Kushner & Dupuis, 2001). The state dynamics are described by a stochastic differential equation, and the value and advantage rate functions can be characterized using its infinitesimal generator. This makes it possible to

Table 2: Notation: LQR environment, dimensions, dynamics, and noise sources.

Symbol	Formula	Note
d	–	Observation dimension (high-dimensional pixels).
d_s	–	Latent state dimension.
d_a	–	Action dimension.
s_t	–	Latent state of the environment at time t.
o_t	$o_t\,\mathrm{d}t = \sum_{i=1}^{d_s} \mu_i s_t^i\,\mathrm{d}t + \epsilon\,\mathrm{d}b_t$	Observation generated from the latent state through a linear mapping with additive noise.
μ_i	–	Observation feature vector corresponding to latent coordinate i.
M	$[\mu_1, \ldots, \mu_{d_s}]$	Linear observation map from latent state space to observation space.
ϵ	–	Magnitude of observation noise.
G	–	State transition matrix governing deterministic dynamics.
H	–	Control matrix determining the effect of actions on the state.
Σ	–	Diffusion matrix describing stochastic state perturbations.
$\mathrm{d}s_t^\pi$	$(Gs_t + H\pi(o_t))\mathrm{d}t + \Sigma\,\mathrm{d}w_t$	Continuous-time state dynamics under policy π.
$\pi(o_t)$	$\phi(o_t)\mathrm{d}t + \sigma\,\mathrm{d}w_t'$	Exploratory policy obtained by adding Gaussian noise to the deterministic policy.
σ	–	Magnitude of exploration noise in action space.
β	–	Discount rate used in the infinite-horizon objective.
$r(s,a)$	$s^\top Q s + a^\top R a$	Quadratic reward function defining the LQR objective.
$J(\mathbf{W})$	$\mathbb{E}\left[\int_0^\infty e^{-\beta t} r(\tilde{s}_t^{\mathbf{W}}, \tilde{a}_t^{\mathbf{W}})\,\mathrm{d}t\right]$	Infinite-horizon discounted return optimized by the policy.
δ_t	–	The bootstrapped TD-error in the upstairs formulation at time t
$\bar{\delta}_t$	–	The bootstrapped TD-error in the downstairs formulation at time t
ℓ	–	The bootstrapping duration: the time window for we which we use the rewards before truncating it with the value estimate

connect sample-based actor-critic updates to the underlying stochastic dynamics. Our observation and exploration noise are written in differential form; their interpretation and the equivalent downstairs state diffusion are given in Sections A.3 and A.7. Environment time t describes the evolution of a rollout, while parameter time τ describes learning across updates.

Relation to block MDPs. The upstairs/downstairs distinction is closely related to the block MDP formulation of RL with rich observations (Du et al., 2019). In a block MDP, observations generated by different latent states belong to disjoint blocks of the observation space. Thus, an observation identifies its generating state, although the decoding map is unknown to the learner. Our noise-free observation model has this decodability property: under Assumption 2,

$$o = Ms, \qquad M^\top M = I_{d_s}, \qquad s = M^\top o.$$

Consequently, distinct latent states have distinct noise-free observations. At fixed decision times, this gives a continuous-state analogue of the block MDP construction; the usual finite-latent-state guarantees do not directly apply to our setting. For nonzero observation noise, the finite-step Gaussian observation distributions overlap, so the noisy model does not satisfy the exact block condition. We therefore use the block MDP connection to describe the decodable latent structure of the noise-free model, while retaining partial observability in the noisy formulation.

What the learning comparison measures. The summary matrices in Equation equation 8 express how the upstairs actor and critic depend on latent states through M. Defining these summaries

Table 3: Additional LQR parameters.

Symbol	Formula	Note
$\tilde{s}_t^{\mathbf{W}}$	–	Latent state trajectory under the policy parameterized by $\mathbf{W}$ after rewriting the dynamics with a single effective noise source.
$\tilde{a}_t^{\mathbf{W}}$	–	Action induced by the policy parameterized by $\mathbf{W}$ along the rollout.
$\bar{\phi}(s;\mathbf{W})$	$\Pi_{l=1}^{L} W_l M s$	Effective downstairs policy obtained by composing the upstairs policy with the observation map.
$A_{\mathrm{cl}}(\mathbf{W})$	$G + H\bar{\phi}(\cdot;\mathbf{W})$	Closed-loop drift matrix under the effective downstairs policy.
C_{tr}	$\Sigma\Sigma^\top$	State transition noise covariance induced by the base diffusion matrix Σ.
C_{expl}	$\sigma^2 H H^\top$	State-space covariance induced by exploration noise in the action.
$C_{\mathrm{obs}}(\mathbf{W})$	$\epsilon^2 H \left(\Pi_{l=1}^{L} W_l M\right)\left(\Pi_{l=1}^{L} W_l M\right)^\top H^\top$	State-space covariance induced by observation noise after passing through the policy and control matrix.
$C_{\mathrm{eff}}(\mathbf{W})$	$C_{\mathrm{tr}} + C_{\mathrm{expl}} + C_{\mathrm{obs}}(\mathbf{W})$	Total effective covariance after combining transition noise, exploration noise, and observation noise.
$\Sigma(\mathbf{W})$	$C_{\mathrm{eff}}(\mathbf{W})^{1/2}$	Effective diffusion matrix in the observation-free downstairs SDE. This is the principal square root of the total effective covariance matrix.

does not require the agent to explicitly learn a decoder. Our theoretical question is whether their updates can be described using only these low-dimensional quantities, despite learning through high-dimensional observations and many trainable matrices. This compares the induced learning dynamics of the two formulations, beyond the existence of a low-dimensional representation of the environment.

E Proofs for Continuous Time Actor-Critic

In context of the LQR problem, we state all definitions and learning algorithms in the downstairs formulation, meaning all functions are state dependent and we will show how it applies to the upstairs formulation, where functions depend on the observations, in the following section. To introduce the learning algorithm, we first define the *infinitesimal generator* associated with the process governed by the SDE defined in Equation equation 12 under a policy ϕ with parameters $\mathbf{W}$ for a function $f : [0, T] \times \mathbb{R}^{d_s} \to \mathbb{R}$ as:

$$\mathcal{L}^{\mathbf{W}} f(t,x) := (Gx + H\bar{\phi}(x;\mathbf{W})) \circ \frac{\partial f(t,x)}{\partial x} + \frac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top \circ \frac{\partial^2 f(t,x)}{\partial^2 x} + \frac{\partial f(t,x)}{\partial t}, \quad (14)$$

where $\circ$ is the Frobenius inner product (see Section A.6). For more details and the formalism of the infinitesimal generator, we refer the reader to Chapter 5 in the textbook by Karatzas & Shreve (2014). This is also a specific case of the more general action-based generator:

$$\mathcal{L}^{a} f(t,x) := (Gx + Ha) \circ \frac{\partial f(t,x)}{\partial x} + \frac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top \circ \frac{\partial^2 f(t,x)}{\partial^2 x} + \frac{\partial f(t,x)}{\partial t}. \quad (15)$$

In plain words, the infinitesimal generator describes, for some function of the state, how is its expected value going to change over the next infinitesimal small instant of time? This operator is applied to characterize equations whose solutions are the value function and the advantage rate function. These equations help derive algorithms for learning the value estimate and the advantage rate estimate using a martingale process, in a sample based manner, which give the deterministic policy gradietn analog in continuous time, and finally leading up to the actor-critic learning updates (Section 3.2).

Table 4: Notation: function approximators and effective matrices used in the actor–critic model.

Symbol	Formula	Note
$\phi(o;\mathbf{W})$	$\prod_{l=1}^{L} W_l o$	Deep linear network representing the deterministic policy mapping observations to actions.
$\mathbf{W}$	$(W_{L-1},\dots,W_1)$	Trainable parameters of the policy network.
W_l	–	Policy matrix corresponding to layer l of the deep linear network.
W_{eff}	$\prod_{l=1}^{L-1} W_l$	Effective matrix obtained by multiplying the trainable policy layers.
$\widehat{W}$	$W_L W_{\text{eff}} M$	Effective policy mapping from latent state to action in the downstairs formulation.
$V(x;\mathbf{W})$	$\mathbb{E}\left[\int_0^\infty e^{-\beta t} r(\tilde{s}_t^{\mathbf{W}}, \tilde{a}_t^{\mathbf{W}})\,\mathrm{d}t\right]$	True value function associated with the policy.
$\varphi(x;\mathbf{U})$	$x^\top(\prod_{l=1}^{L} U_l)x + (\prod_{l=1}^{L} U_l')x + c$	Quadratic neural network used to estimate the value function.
U_{eff}	$\prod_{l=1}^{L} U_l$	Effective quadratic matrix in the value estimate.
U'_{eff}	$\prod_{l=1}^{L-1} U_l'$	Effective linear component in the value estimate.
$\mathrm{q}(x,a)$	$\lim_{\Delta t\to 0} \frac{Q_{\Delta t}(x,a)-V(x)}{\Delta t}$	Continuous-time advantage rate function.
$\Psi(o,a)$	$a^\top(\prod_{l=1}^{L} Z_l)o + (\prod_{l=1}^{L} Z_l')a$	Neural network approximation of the advantage rate function.
$\widehat{U}$	$M^\top U_{\text{eff}} M$	Low-dimensional quadratic value matrix.
$\widehat{U'}$	$U_L' U'_{\text{eff}} M$	Low-dimensional linear value component.
$\widehat{Z}$	$M^\top Z_{\text{eff}} Z_1$	Low-dimensional matrix coupling state and action in the advantage estimate.
$\widehat{Z'}$	$Z_L' Z'_{\text{eff}} Z_1'$	Effective linear-in-action component of the advantage rate function.
$\mathcal{L}^{\mathbf{W}}$	Equation equation 14	Infinitesimal generator describing the expected infinitesimal change of functions of the state.

Characterizing the Value Function: Given the definition for the value function (Equation equation 3 for the infinite horizon problem, we present the following lemma that characterizes the value function in a time-invariant manner for the LQR.

Lemma 1. *Assume that for some policy parameters* $\mathbf{W}$ *there exists a function* $V \in C^2(\mathbb{R}^{d_s})$ *satisfying the following equation:*

$$\mathcal{L}^{\mathbf{W}}\widehat{V}(x;\mathbf{W}) + r(x,\bar{\phi}(x;\mathbf{W})) - \beta\widehat{V}(x;\mathbf{W}) = 0, \tag{16}$$

then we have $V = \widehat{V}$ *for* $x \in \mathbb{R}^{d_s}$.

The proof is presented in Section E.1.

Characterizing the Advantage Rate Function: To estimate the semi-gradient based update direction for the policy, the agent has to simulataneously maintain estimates of the value function, and the advantage rate function q (Equation equation 26). We present a characterization of the q-function as a solution to a PDE similar to the value function.

Lemma 2. *The* q*-function satisfies the following PDE:*

$$\mathrm{q}(x,a;\mathbf{W}) = \mathcal{L}^a V(x;\mathbf{W}) - \beta V(x;\mathbf{W}) + r(x,a). \tag{17}$$

The proof of this Lemma can be found in Section E.2. A simple corollary of this lemma characterizes the advantage rate function at $a = \bar{\phi}(x;\mathbf{W})$ to be zero.

Corollary 1. *For* $a = \bar{\phi}(x;\mathbf{W})$, $\mathbb{E}[\mathrm{q}(x,a;\mathbf{W})] = 0$.

This means that local in time advantage of taking the same action as the policy is zero, which can also be observed from evaluating the limit in Equation 26. Equipped with the definition and key characteristics, we can see that the parameterization of the advantage rate function as above (Equation equation 7) is consistent.

Martingale Characterization: The agent's goal is to estimate the value function V and the advantage rate function q, which are used to estimate the direction of ascent of the policy parameters. One way to learn these functions together is to take the optimization approach by Robbins & Monro (1951). We show that a Martingale process can be used to estimate semi-gradients. A martingale is a stochastic process whose conditional expected future value, given all information up to the present, equals its current value (See Section A.5). We describe the nuances of how the reward is designed in our exploratory setting. In practice, while the exploratory dynamics are as described in Equation equation 12 the reward the agent receives from the environment also depends on the noisy action the agent takes (see equation equation 4). Therefore, the action for which the agent receives a reward is governed by the following equation.

$$a_t^{\mathbf{W}}\mathrm{d}t \;=\; \bar{\phi}(\tilde{s}_t^{\pi};\mathbf{W})\mathrm{d}t + \bar{\Sigma}(\mathbf{W})\mathrm{d}w_t', \tag{18}$$

where $\bar{\Sigma}(\mathbf{W}) \in \mathbb{R}^{d_a\times d_a}$ is the principal square root of $\epsilon^2\widehat{W}(\mathbf{W})\widehat{W}(\mathbf{W})^\top + \sigma^2 I$, where $\widehat{W}(\mathbf{W}) := W_{\text{eff}}\, M \in \mathbb{R}^{d_a\times d_s}$ is the summarized policy matrix in the downstairs formulation. This process $a_t^{\mathbf{W}}$ can be re-interpreted in the equivalent form:

$$a_t \;=\; \bar{\phi}(\tilde{s}_t^{\pi};\mathbf{W}) \;+\; \bar{\Sigma}(\mathbf{W})\,\zeta_t, \tag{19}$$

where ζ_t is the white noise, similar to but independent of ξ_t, see Section A.3. We make an assumption about the noisy action above before introducing a key lemma.

Assumption 3. *We assume that $a_t^{\mathbf{W}}$ has full support in $\mathbb{R}^{d_a}$ for all $t \in (0,\infty)$.*

Under the above parameterization for noisy actions, the assumption reduces to $\bar{\Sigma}$ being a positive semidefinite matrix and, therefore, $\bar{\Sigma}(\mathbf{W})\zeta_t$ being a nondegenerate Gaussian in $\mathbb{R}^{d_a}$. Under this assumption, we can characterize a Martingale process which will be used to learn value estimate and the advantage rate functions. Note that the following lemma is similar to Theorem 3.2 by Cheng et al. (2026) but for the exploratory dynamics of Equation equation 12 and the above actions.

Lemma 3. *Let $\mathbf{W}$ be the policy parameter, $\hat{V} \in C^{1,2}(\mathbb{R}^{d_s})$, and $\hat{\mathrm{q}} \in C(\mathbb{R}^{d_s}\times\mathbb{R}^{d_a})$ satisfy the following conditions:*

1. *$\hat{\mathrm{q}}(x,\bar{\phi}(a;\mathbf{W})) = 0$,*
2. *The process: $e^{-\beta(l-t)}\hat{V}(\tilde{s}_l^{\mathbf{W}}) + \int_t^l e^{-\beta(u-t)}\left(r(\tilde{s}_u^{\mathbf{W}},a_u^{\mathbf{W}}) - \hat{\mathrm{q}}(\tilde{s}_u^{\mathbf{W}},a_u^{\mathbf{W}};\mathbf{W})\right)\mathrm{d}u$, for $l \in (t,\infty)$, where $\tilde{s}_t^{\mathbf{W}}$ is governed by the SDE in Equation equation 12 for the actions that satisfy Assumption 3 and are governed by the SDE in Equation 18 is martingale.*

Further, all stochastic processes are defined on a filtered probability space $(\Omega,\mathcal{F},(\mathcal{F}_t)_{t\geq 0},\mathbb{P})$ where $(\mathcal{F}_t)_{t\geq 0}$ denotes the usual augmentation of the natural filtration generated by the mutually independent Wiener processes w_t, and w_t' i.e. $\mathcal{F}_t = \sigma((w_s,w_s')_{0\leq s\leq t})^{\text{aug}}$. Then $\hat{V}(x) = V(x;\mathbf{W})$ and $\hat{\mathrm{q}}(x,a) = \mathrm{q}(x,a;\mathbf{W})$ for all $x\in\mathbb{R}^{d_s}$ and $a\in\mathbb{R}^{d_a}$.

We refer the reader to Section E.3 for the proof. The martingale characterization can be used to derive a temporal difference error (TD error) to simultaneously learn the parameters of the value and advantage rate estimates. Let δ_t be the analog to the TD error in continuous time: $\varphi(\tilde{s}_t^{\mathbf{W}};\mathbf{U}) - \int_t^{t+\ell} e^{-\beta(u-t)}\left[r(\tilde{s}_u^{\mathbf{W}},a_u^{\mathbf{W}}) - \psi(\tilde{s}_u^{\mathbf{W}},a_u^{\mathbf{W}})\right]\mathrm{d}u - e^{-\beta\ell}\varphi(\tilde{s}_{t+\ell}^{\mathbf{W}};\mathbf{U})$. This implies that the loss $L(\mathbf{U},\mathbf{Z}) = \mathbb{E}\left[\int_0^\infty (e^{-\beta t}\delta_t^2)dt\right]$ has to be zero and we can use this fact to define the Gradient updates in Section 3.2. This is our key theoretical contribution in continuous time RL: deriving the appropriate formulation for learning the value and advantage rate estimate in the high-dimensional LQR setting with exploration.

These lemmas lead trivially to the deep deterministic policy gradient theorem for our setting (following proof similar to the ones provided by Cheng et al. (2026) which builds on discrete time work by Silver et al. (2014)).

Theorem 1. *For all $x\in\mathbb{R}^{d_s}$ and $\mathbf{W}$ we have:*

$$\partial_{\mathbf{W}}V(x;\mathbf{W}) = \mathbb{E}\left[\int_0^\infty e^{-\beta l}\partial_{\mathbf{W}}\bar{\phi}(\tilde{s}_l^{\mathbf{W}};\mathbf{W})\partial_a \mathrm{q}(\tilde{s}_l^{\mathbf{W}},a;\mathbf{W})|_{a=\bar{\phi}(\tilde{s}_l^{\mathbf{W}};\mathbf{W})}\mathrm{d}l\right].$$

In the following subsections we provide proofs for all the theorem that were introduced in Section 3.2 and the lemmas introduced above.

E.1 Proof of Lemma 1

Lemma 1. *Assume that for some policy parameters* $\mathbf{W}$ *there exists a function* $V \in C^2(\mathbb{R}^{d_s})$ *satisfying the following equation:*

$$\mathcal{L}^{\mathbf{W}}\widehat{V}(x;\mathbf{W}) + r(x,\bar{\phi}(x;\mathbf{W})) - \beta\widehat{V}(x;\mathbf{W}) = 0, \tag{20}$$

then we have $V = \widehat{V}$ *for* $x \in \mathbb{R}^{d_s}$.

Proof. First, we define the value function in this setting. Under the policy parameterized by $\mathbf{W}$ the value function at time t and state x is defined as:

$$V(x;\mathbf{W}) = \mathbb{E}\Big[\int_t^\infty e^{-\beta(l-t)}\, r\big(\tilde{s}_l^{\mathbf{W}}, \tilde{a}_l^{\mathbf{W}}\big)\,\mathrm{d}l \Big|\ \tilde{s}_t^{\mathbf{W}} = x\Big],$$
$$\tilde{a}_l^{\mathbf{W}} = \bar{\phi}\big(\tilde{s}_l^{\mathbf{W}};\mathbf{W}\big).$$

Under $\mathbf{W}$, the state follows

$$\mathrm{d}\tilde{s}_t = (G\tilde{s}_t + H\widehat{W}(\mathbf{W}))\,\mathrm{d}t + \Sigma(\mathbf{W})\,\mathrm{d}w_t.$$

Expanding the expression in $[t, t+h]$:

$$V(x) = \mathbb{E}\left[\int_t^{t+h} e^{-\beta(s-t)} r(X_s,\pi(X_s))\,\mathrm{d}s + e^{-\beta h}\, V(X_{t+h}) \;\Big|\; X_t = x\right].$$

Short-time expansions:

$$\int_t^{t+h} e^{-\beta(s-t)} r(X_s,\pi(X_s))\,\mathrm{d}s = h\, r(x,\pi(x)) + o(h),$$
$$e^{-\beta h} = 1 - \beta h + o(h).$$

Itô expansion for $V(X_{t+h})$:

$$\begin{aligned}\mathbb{E}\big[V(X_{t+h}) \,\big|\, X_t = x\big] =& V(t,x)\\ &+ h\Big(\partial_t V(x) + \big(G + H\bar{\phi}(x;\mathbf{W})\big)\circ\nabla_x V(x)\\ &+ \tfrac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top \circ \nabla_x^2 V(x)\Big) + o(h),\end{aligned} \tag{21}$$

Multiply by $e^{-\beta h}$:

$$\begin{aligned}e^{-\beta h}\,\mathbb{E}[V(t+h, X_{t+h}) \,|\, X_t = x] =& \big(1 - \beta h + o(h)\big)\Big(V(x)\\ &+ h\big(\big(Gx + H\bar{\phi}(x;\mathbf{W})\big)\circ\nabla_x V(x)\\ &+ \tfrac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top\circ\nabla_x^2 V(x)\big) + o(h)\Big)\\ =& V(x)\\ &+ h\Big(\big(Gx + H\bar{\phi}(x;\mathbf{W})\big)\circ\nabla_x V(x)\\ &+ \tfrac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top\circ\nabla_x^2 V(x)) - \beta V(x)\Big) + o(h).\end{aligned}$$

Plug back into Equation equation 21 and collect terms:

$$\begin{aligned}V(x) =& h\, r(x,\pi(x))\\ &+ \Big[V(x) + h\big(\big(Gx + H\bar{\phi}(x;\mathbf{W})\big)\circ\nabla_x V(x) + \tfrac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top\circ\nabla_x^2 V(x) - \beta V(x)\big) + o(h)\Big].\end{aligned}$$

Cancel $V(x)$ divide by h, let $h \to 0$:

$$0 = r(x,\pi(x)) + \big(Gx + H\bar{\phi}(x;\mathbf{W})\big)\circ\nabla_x V(x) + \tfrac{1}{2}\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top\circ\nabla_x^2 V(x)\big) - \beta V(x).$$

Equivalently, in generator form we have:

$$(\mathcal{L}^{\mathbf{W}}V)(x) + r(x,\bar{\phi}(x)) - \beta V(x) = 0.$$

Since both V and $\widehat{V}$ satisfy the same equation which has a unique solution (Beck et al., 2021) we have that $V = \widehat{V}$, which proves the Lemma.

□

E.2 Proof of Lemma 2

Lemma 2. *The* q*-function satisfies the following PDE:*

$$\mathrm{q}(x,a;\mathbf{W}) = \mathcal{L}^a V(x;\mathbf{W}) - \beta V(x;\mathbf{W}) + r(x,a). \tag{22}$$

Proof (following Jia & Zhou (2023), Proposition 3). Fix $(x,a) \in \mathbb{R}^{d_s} \times \mathbb{R}^{d_a}$, and consider the short-time dynamics

$$\mathrm{d}s_l^a = (Gs_l^a + Ha)\,\mathrm{d}l + \Sigma(\mathbf{W})\,\mathrm{d}w_l, \qquad s_0^a = x.$$

Let $V(\cdot;\mathbf{W})$ denote the value function associated with policy π parameterized by $\mathbf{W}$ under discount rate β. Applying Itô's lemma to $e^{-\beta l}V(s_l^a;\mathbf{W})$ on $[0,\Delta t]$, we obtain

$$e^{-\beta\Delta t}V(s_{\Delta t}^a;\mathbf{W}) - V(x;\mathbf{W}) = \int_0^{\Delta t} e^{-\beta l}\big(\mathcal{L}^a V - \beta V\big)(s_l^a;\mathbf{W})\,\mathrm{d}l + M_{\Delta t},$$

where

$$M_{\Delta t} := \int_0^{\Delta t} e^{-\beta l}\nabla_x V(s_l^a;\mathbf{W})^\top \Sigma(\mathbf{W})\,\mathrm{d}w_l$$

is an Itô stochastic integral and hence a martingale satisfying $\mathbb{E}[M_{\Delta t}] = 0$ and $\mathcal{L}^a$ is as defined in equation 15. Taking expectations, the martingale term vanishes and we have

$$\mathbb{E}\big[e^{-\beta\Delta t}V(s_{\Delta t}^a;\mathbf{W})\big] - V(x;\mathbf{W}) = \mathbb{E}\left[\int_0^{\Delta t} e^{-\beta l}\big(\mathcal{L}^a V - \beta V\big)(s_l^a;\mathbf{W})\mathrm{d}l \Big| s_0 = x\right]. \tag{23}$$

By definition (Equation equation 25),

$$Q_{\Delta t}(x,a;\mathbf{W}) = \mathbb{E}\Big[\int_0^{\Delta t} e^{-\beta l} r(s_l^a,a)\,\mathrm{d}l + e^{-\beta\Delta t}V(s_{\Delta t}^a;\mathbf{W})\Big| s_0 = x\Big].$$

Subtracting $V(x;\mathbf{W})$ and substituting from equation 23, we obtain

$$Q_{\Delta t}(x,a;\mathbf{W}) - V(x;\mathbf{W}) = \mathbb{E}\left[\int_0^{\Delta t} e^{-\beta l}\big(r(s_l^a,a) + \mathcal{L}^a V(s_l^a;\mathbf{W}) - \beta V(s_l^a;\mathbf{W})\big)\,\mathrm{d}l \Big| s_0 = x\right].$$

For small Δt, the integrand is approximately constant at $l = 0$, so

$$Q_{\Delta t}(x,a;\mathbf{W}) - V(x;\mathbf{W}) = \big(r(x,a) + \mathcal{L}^a V(x;\mathbf{W}) - \beta V(x;\mathbf{W})\big)\,\Delta t + o(\Delta t).$$

Dividing both sides by Δt and letting $\Delta t \to 0$ yields

$$\lim_{\Delta t\to 0} \frac{Q_{\Delta t}(x,a;\mathbf{W}) - V(x;\mathbf{W})}{\Delta t} = r(x,a) + \mathcal{L}^a V(x;\mathbf{W}) - \beta V(x;\mathbf{W}).$$

By definition of the advantage rate function in equation 26, this gives

$$\mathrm{q}(x,a;\mathbf{W}) = \mathcal{L}^a V(x;\mathbf{W}) - \beta V(x;\mathbf{W}) + r(x,a).$$

□

E.3 Proof of Lemma 3

Lemma 3. *Let* $\mathbf{W}$ *be the policy parameter,* $\hat{V} \in C^{1,2}(\mathbb{R}^{d_s})$*, and* $\hat{\mathrm{q}} \in C(\mathbb{R}^{d_s} \times \mathbb{R}^{d_a})$ *satisfy the following conditions:*

1. $\hat{\mathrm{q}}(x, \bar{\phi}(a;\mathbf{W})) = 0$*,*
2. *The process:* $e^{-\beta(l-t)}\hat{V}(\tilde{s}_l^{\mathbf{W}}) + \int_t^l e^{-\beta(u-t)}\left(r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \hat{\mathrm{q}}(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}};\mathbf{W})\right)\mathrm{d}u$*, for* $l \in (t,\infty)$*, where* $\tilde{s}_t^{\mathbf{W}}$ *is governed by the SDE in Equation equation 12. for the actions satisfy Assumption 3 and are governed by the SDE in Equation 18 is martingale.*

Further, all stochastic processes are defined on a filtered probability space $(\Omega,\mathcal{F},(\mathcal{F}_t)_{t\geq 0},\mathbb{P})$ *where* $(\mathcal{F}_t)_{t\geq 0}$ *denotes the usual augmentation of the natural filtration generated by the mutually independent Wiener processes* w_t*, and* w_t' *i.e.* $\mathcal{F}_t = \sigma((w_s,w_s')_{0\leq s\leq t})^{\mathrm{aug}}$*. Then* $\hat{V}(x) = V(x;\mathbf{W})$ *and* $\hat{\mathrm{q}}(x,a) = \mathrm{q}(x,a;\mathbf{W})$ *for all* $x \in \mathbb{R}^{d_s}$ *and* $a \in \mathbb{R}^{d_a}$*.*

Proof. First apply Itô 's formula to the function $e^{-\beta(s-t)}(\tilde{s}_s^{\mathbf{W}})$:

$$e^{-\beta(l-t)}\hat{V}(\tilde{s}_l^{\mathbf{W}}) - \hat{V}(\tilde{s}_t^{\mathbf{W}}) = \int_t^l e^{-\beta(u-t)}\left(\mathcal{L}^{\mathbf{W}}\hat{V}(\tilde{s}_u^{\mathbf{W}}) - \beta\hat{V}(\tilde{s}_u^{\mathbf{W}})\right)\mathrm{d}u + \int_t^l e^{-\beta(u-t)}\nabla_x\hat{V}(x)|_{x=\tilde{s}_l^{\mathbf{W}}}\Sigma(\mathbf{W})\mathrm{d}w_u,$$

where the infinitesimal generator is as defined in Equation equation 14. Since the second integral expression above, with dw_u is a martingale therefore from the martingale condition in the statement of the lemma we have:

$$\int_t^l \left(\mathcal{L}^{\mathbf{W}}\hat{V}(\tilde{s}_u^{\mathbf{W}}) - \beta\hat{V}(\tilde{s}_u^{\mathbf{W}}) + r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \hat{\mathrm{q}}(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \mathbf{W})\right)\mathrm{d}u, \text{ for } l \in [t, \infty)$$

is an $\mathcal{F}_t$ martingale, which has continuous paths and finite variation. Let $f(u, x, a) = \mathcal{L}^{\mathbf{W}}\hat{V}(x) - \beta\hat{V}(x) + r(x, a) - \hat{\mathrm{q}}(x, a; \mathbf{W})$ finite variation we can state the following:

$$\int_t^l e^{-\beta(u-t)} f(u, \tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}})\mathrm{d}u = 0.$$

Since the integral is absolutely continuous, taking the Lebesgue derivative of this equation gives us: $f(u, \tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) = 0$. Now to show that this holds for general $x \in \mathbb{R}^{d_s}$ and $a \in \mathbb{R}^{d_a}$ we use the idea of full support of the action noise. For any Borel set $B \subset \mathbb{R}^{d_s}$ we have:

$$\begin{aligned} &\mathbb{E}\left[1_{\tilde{s}_u^{\mathbf{W}} \in B} f(u, \tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}})\right] = 0 \\ \implies &\mathbb{E}\left[1_{\tilde{s}_u^{\mathbf{W}} \in B}\mathbb{E}\left[f(u, \tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}})|\tilde{s}_u^{\mathbf{W}}\right]\right] = 0, \end{aligned}$$

and therefore, for $\tilde{s}_u^{\mathbf{W}}$ almost everywhere:

$$\mathbb{E}\left[f(u, \tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}})|\tilde{s}_u^{\mathbf{W}} = x\right] = 0.$$

Since the action $a^{\mathbf{W}}$ (Equation equation 18) is a non-degenerate Gaussian on $\mathbb{R}^{d_a}$ with full support this implies that $f(u, x, a) = 0$ for all $a \in \mathbb{R}^{d_a}$ and $\tilde{s}_u^{\mathbf{W}}$ being x a.e. Similarly, since $\tilde{s}_u^{\mathbf{W}}$ has a nondegenerate diffusion coefficient $\Sigma(\mathbf{W})$ and therefore it has full support on $\mathbb{R}^{d_s}$ and further from continuity we that $f(u, x, a) = 0$ for $u \in [t, \infty)$ a.e. Finally, from continuity in u we have that $f(u, x, a) = 0$ holds everywhere. Now since $f(u, x, a) = 0$ and we also have that $\hat{\mathrm{q}}(x, \bar{\phi}(a)) = 0$ we have:

$$\mathcal{L}^{\mathbf{W}}\hat{V}(x) + r(x, \bar{\phi}(x; \mathbf{W})) - \beta\hat{V}(x; \mathbf{W}) = 0,$$

and therefore since $\hat{V} \in C^{1,2}(\mathbb{R}^{d_s})$ and it satisfies the same PDE as equation equation 16 we have $V = \hat{V}$ by Feynman-Kac formula (Karatzas & Shreve, 2014). This leads to the assertion that $\hat{\mathrm{q}}$ satisfies the $\mathcal{L}^{\mathbf{W}}V(x; \mathbf{W}) + r(x, a) - \hat{\mathrm{q}}(x, a) = 0$ and therefore $\hat{\mathrm{q}} = \mathrm{q}$ by similar argument, which proves the statement of the Lemma. □

E.4 Continuous-Time Deterministic Policy Gradient

We specialize the continuous-time deterministic policy-gradient formula in Proposition 3.1 of Cheng et al. (2026) to discounted infinite-horizon LQR. Write $\bar{\phi}(s; \mathbf{W}) = \bar{\phi}(s; \mathbf{W}) = \widehat{W}s$ for the deterministic downstairs policy. Here the value function evaluates this policy under the downstairs diffusion:

$$V(s; \mathbf{W}) = \mathbb{E}\left[\int_0^\infty e^{-\beta t} r\big(\tilde{s}_t^{\mathbf{W}}, \bar{\phi}(\tilde{s}_t^{\mathbf{W}}; \mathbf{W})\big)\mathrm{d}t \,\middle|\, \tilde{s}_0^{\mathbf{W}} = s\right]. \tag{24}$$

The reward is evaluated at the deterministic policy action; the effect of white-noise perturbations on the state is included in the diffusion described in Section A.7, following the continuous-time exploration formulation of Tiwari et al. (2026).

For completeness, taking action a for duration Δt and then following this policy gives

$$Q_{\Delta t}(s, a; \mathbf{W}) := \mathbb{E}\left[\int_0^{\Delta t} e^{-\beta u} r(s_u^a, a)\mathrm{d}u + e^{-\beta\Delta t} V(s_{\Delta t}^a; \mathbf{W}) \,\middle|\, s_0^a = s\right], \tag{25}$$

where s_u^a has drift $Gs_u^a + Ha$ and the same fixed effective diffusion covariance as the evaluated policy. The advantage rate is

$$\mathrm{q}(s, a; \mathbf{W}) := \lim_{\Delta t \to 0} \frac{Q_{\Delta t}(s, a; \mathbf{W}) - V(s; \mathbf{W})}{\Delta t}. \tag{26}$$

Theorem 2 (Continuous-time deterministic policy gradient)**.** *Assume discounted stability holds locally in the policy parameters, the value function is twice continuously differentiable in state and differentiable in the policy parameters, and the discounted integrability and vanishing boundary conditions needed for differentiation and the infinite-horizon limit hold. Suppose the effective diffusion covariance is fixed under the policy variations considered. Then, for every initial state $s \in \mathbb{R}^{d_s}$,*

$$\partial_{\mathbf{W}} V(s; \mathbf{W}) = \mathbb{E}\left[\int_0^\infty e^{-\beta t} \left[\partial_{\mathbf{W}} \bar{\phi}(\tilde{s}_t^{\mathbf{W}}; \mathbf{W})\right]^\top \partial_a \mathrm{q}(\tilde{s}_t^{\mathbf{W}}, a; \mathbf{W})\big|_{a=\bar{\phi}(\tilde{s}_t^{\mathbf{W}}; \mathbf{W})} \, \mathrm{d}t \,\middle|\, \tilde{s}_0^{\mathbf{W}} = s\right].$$

For constrained policy matrices, the identity is interpreted in local coordinates on the admissible parameter manifold.

The identity is the discounted infinite-horizon counterpart of their finite-horizon result and the continuous-time analogue of deterministic policy gradients in discrete time (Silver et al., 2014). Theorem 3.2 of Cheng et al. (2026) provides the associated martingale characterization of the value and advantage rate functions; our preceding characterizations adapt this approach to the stated downstairs dynamics. Replacing the advantage rate by its learned estimate gives the actor update direction used in Section 3.2.

The covariance condition is relevant when applying the result to our observation model. With fixed W_L and orthogonal trainable policy layers, combining the independent noises gives

$$\Sigma(\mathbf{W})\Sigma(\mathbf{W})^\top = \Sigma\Sigma^\top + \sigma^2 HH^\top + \epsilon^2 HW_L W_L^\top H^\top,$$

which is constant along admissible changes of the trainable matrices. If the effective covariance varies independently with policy parameters, its derivative must also be included in the policy-gradient calculation.

F Proof of Low-Dimensional Underlying Dynamics for One-Step Update

Learning algorithms are modeled as processes in *parameter time*, which corresponds to the updates made to the parameters of neural networks. The learning dynamics, in parameter time, shed light on the behavior of parameterized functions (Jacot et al., 2018; Ben Arous et al., 2022) by deriving moment to moment changes in the parameter over one gradient step. Most effective theoretical models of learning are inspired by optimization theory, which models iterative learning algorithms as continuous time dynamical systems (Kushner & Yin, 2003; Su et al., 2016). While there is a strain of work that analyzes the learning dynamics in online RL (Borkar & Meyn, 2000; Cai et al., 2019b; Fan et al., 2020; Jin et al., 2020), they often study various algorithms for environments with discrete states or actions, with some exceptions (Tiwari et al., 2026). We model the learning dynamics for RL where both states and actions are continuous, and the observations are high-dimensional. To derive the one step gradient update for the objective $J(\mathbf{W})$, we split it into lemmas for teh individual effective matrices, and the TD-error. We map each one of them to the underlying dynamics showing that they individually have low-dimensional structure and together the entire learning dynamics can be described using far fewer variables than there are parameters.

F.1 Effective Dynamics of Quadratic Value Estimate Matrix

The matrices themselves are large, in dimension $d \times d$, but instead of deriving and analyzing the learning dynamics of large matrices, we can derive and analyze the effective dynamics of the underlying matrices, which capture summary statistics. For the value estimate, φ, consider the following matrix:

$$\widehat{U} := M^\top U_{\text{eff}} M \in \mathbb{R}^{d_s \times d_s},$$

which is the state-dependent quadratic component of the value function as opposed to the observation dependent quadratic component of the value function which is a $d \times d$ matrix. We state the following result for gradient update (one-step dynamic) under the update rule described in Section 2.

Lemma 4. *For learning rate η, the effective matrix $\widehat{U}$ follows the one-step dynamic:*

$$\Delta\widehat{U} = \eta L\Big(\widehat{\mathcal{G}}(\widehat{U}) - \widehat{U}\,\widehat{\mathcal{G}}(\widehat{U})^\top\,\widehat{U}\Big) + O(\eta^2) + O(\eta\epsilon),$$

$$\textit{with}\quad \widehat{\mathcal{G}}(\widehat{U}) = \int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{s}_t^{\mathbf{W}}(\tilde{s}_t^{\mathbf{W}})^\top \mathrm{d}t.$$

Proof. To prove this we first present the expression for $\partial_{U_l}\varphi(\cdot)$:

$$\partial_{U_l}\varphi(o;\mathbf{U}) = \left(\prod_{i=l+1}^{L} U_i\right)^\top o\,o^\top \left(\prod_{i=1}^{l-1} U_i\right)^\top. \tag{27}$$

for $l \in \{1, 2, \ldots, L\}$. For the matrix at $l = 1$ we assume that the products $\Pi_{i=1}^{0}$ is an identity matrices I_d. Now consider the change in U_l:

$$\begin{aligned}\Delta U_l =& U_l^+ - U_l \\ =& \eta\,\left(I - \tfrac{\eta}{2}A_l\right)^{-1} A_l\,U_l,\end{aligned}$$

where $A_l = \mathcal{G}(U_l)U_l^\top - U_l\mathcal{G}(U_l)^\top$. This can be expanded using the Neumann series:

$$\begin{aligned}\Delta U_l =& \eta A_l U_l + O(\eta^2) \\ =& \eta\Big(\mathcal{G}(U_l) - U_l(\mathcal{G}(U_l))^\top U_l\Big) \;+\; O(\eta^2).\end{aligned}$$

Using this we can derive the change in U_{eff} over one gradient step:

$$\begin{aligned}\Delta U_{\text{eff}} \;&=\; \sum_{l=1}^{L}\prod_{i=l+1}^{L} U_i\,\Delta U_l \prod_{i=1}^{l-1} U_i + O(\eta^2) \\ &= \eta\sum_{l=1}^{L}\Big[\prod_{i=l+1}^{L} U_i\,\mathcal{G}(U_l)\prod_{i=1}^{l-1} U_i - \prod_{i=l+1}^{L} U_i\,U_l\,\mathcal{G}(U_l)^\top U_l\prod_{i=1}^{l-1} U_i\Big] + O(\eta^2).\end{aligned}$$

To simplify this expression consider the second summand which is the Stiefel corrective term for any l and notice that the $\mathcal{G}$, gradient expression, already contains $\prod_{i=1}^{l-1} U_i$ and $\prod_{i=l+1}^{L} U_i$ which leads to the following simplification:

$$\begin{aligned}\prod_{i=l+1}^{L} U_i\,U_l\,\mathcal{G}(U_l)^\top U_l\prod_{i=1}^{l-1} U_i &= \int_0^\infty e^{-\beta t}\,\delta_t\,U_{\text{eff}}\,o_t o_t^\top\,U_{\text{eff}}\,\mathrm{d}t \\ &= U_{\text{eff}}\left(\int_0^\infty e^{-\beta t}\,\delta_t\,o_t o_t^\top\,\mathrm{d}t\right)U_{\text{eff}} \\ &= U_{\text{eff}}\mathcal{G}(U_{\text{eff}})U_{\text{eff}}.\end{aligned}$$

Here the expression for $\mathcal{G}(U_l)$ is obtained by substituting the expression for the partial (Equation equation 27) into the gradient ascent expression from Equation equation 11. Further the first expression in the summation can be simplified as shown by Arora et al. (2018) (also see Theorem 2 by Menon (2023)):

$$\begin{aligned}\sum_{l=1}^{L}\Big[\prod_{i=l+1}^{L} U_i\,\mathcal{G}(U_l)\prod_{i=1}^{l-1} U_i\Big] &= \sum_{l=1}^{L}(U_{\text{eff}}U_{\text{eff}}^\top)^{\frac{L-l}{L}}\mathcal{G}(U_{\text{eff}})(U_{\text{eff}}^\top U_{\text{eff}})^{\frac{l-l}{L}} \\ &= \sum_{l=1}^{L}(I_d)^{\frac{L-l}{L}}\mathcal{G}(U_{\text{eff}})(I_d)^{\frac{l-l}{L}} \\ &= L\mathcal{G}(U_{\text{eff}}),\end{aligned}$$

where the second equality is from the fact that U_{eff} is on the Stiefel manifold and therefore $U_{\text{eff}}^\top U_{\text{eff}} = U_{\text{eff}} U_{\text{eff}}^\top = I$. Therefore, we can now summarize the change in the effective matrix, by combining both of the updates derived above, as follows:

$$\Delta U_{\text{eff}} \ = \eta L \left(\mathcal{G}(U_{\text{eff}}) - U_{\text{eff}} \mathcal{G}(U_{\text{eff}})^\top U_{\text{eff}} \right) + O(\eta^2).$$

Now we derive the change in the low-dimensional matrix which summarizes these larger matrices $\widehat{U}$:

$$\begin{aligned} \Delta \widehat{U} =& M^\top \Delta U_{\text{eff}} M \\ =& \eta L M^\top \left(\mathcal{G}(U_{\text{eff}}) - U_{\text{eff}} \mathcal{G}(U_{\text{eff}}) U_{\text{eff}} \right) M. \end{aligned} \tag{28}$$

To further present this in terms of the underlying low-dimensional matrix, consider the following:

$$\begin{aligned} o_t o_t^\top =& \left(M \tilde{s}_t^{\mathbf{W}} + \epsilon \xi_t \right) \left(M \tilde{s}_t^{\mathbf{W}} + \epsilon \xi_t \right)^\top \\ =& M \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top M^\top \ + \ \epsilon \, M \tilde{s}_t^{\mathbf{W}} \xi_t^\top \ + \ \epsilon \, \xi_t (\tilde{s}_t^{\mathbf{W}})^\top M^\top \ + \ \epsilon^2 \xi_t \xi_t^\top. \end{aligned}$$

Substituting these into the expression for the change U_{eff} above and splitting it into three parts, we obtain:

$$\begin{aligned} \mathcal{G}(U_{\text{eff}}) &= \mathcal{G}_{\text{state}} + \mathcal{G}_{\text{cross}} + \mathcal{G}_{\text{noise}}, \\ \mathcal{G}_{\text{state}} &:= \int_0^\infty e^{-\beta t} \, \delta_t \, M \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top M^\top \, \mathrm{d}t, \\ \mathcal{G}_{\text{cross}} &:= \epsilon \int_0^\infty e^{-\beta t} \, \delta_t \left(M \tilde{s}_t^{\mathbf{W}} \xi_t^\top + \xi_t (\tilde{s}_t^{\mathbf{W}})^\top M^\top \right) \mathrm{d}t, \\ \mathcal{G}_{\text{noise}} &:= \epsilon^2 \int_0^\infty e^{-\beta t} \, \delta_t \, \xi_t \xi_t^\top \, \mathrm{d}t. \end{aligned}$$

Here, the first part is the one corresponding to the underlying state dynamics projected "upwards" with the transform: $M(\cdot)M^\top$, the second is the term that is a cross between the state being projected upwards and the noise term, and the third expression is the quadratic noise term emerging from the observation noise. Now we substitute these into the change in the lower-dimensional matrix (Equation equation 28) we obtain:

$$\begin{aligned} M^\top \mathcal{G}(U_{\text{eff}}) M =& M^\top \left(\mathcal{G}_{\text{state}} + \mathcal{G}_{\text{cross}} + \mathcal{G}_{\text{noise}} \right)^\top M \\ =& \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top M \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top M^\top M \, \mathrm{d}t \\ &+ \epsilon \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top \left(M \tilde{s}_t^{\mathbf{W}} \xi_t^\top + \xi_t (\tilde{s}_t^{\mathbf{W}})^\top M^\top \right) M \, \mathrm{d}t \\ &+ \epsilon^2 \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top \xi_t \xi_t^\top \, M \mathrm{d}t \end{aligned}$$

Similarly, consider the second term in Equation equation 28.

$$\begin{aligned} M^\top U_{\text{eff}} \mathcal{G}(U_{\text{eff}})^\top U_{\text{eff}} M =& M^\top U_{\text{eff}} \left(\mathcal{G}_{\text{state}} + \mathcal{G}_{\text{cross}} + \mathcal{G}_{\text{noise}} \right)^\top U_{\text{eff}} M \\ =& \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top U_{\text{eff}} M \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top M^\top U_{\text{eff}} M \, \mathrm{d}t \\ &+ \epsilon \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top U_{\text{eff}} \left(M \tilde{s}_t^{\mathbf{W}} \xi_t^\top + \xi_t (\tilde{s}_t^{\mathbf{W}})^\top M^\top \right) U_{\text{eff}} M \, \mathrm{d}t \\ &+ \epsilon^2 \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top U_{\text{eff}} \xi_t \xi_t^\top \, U_{\text{eff}} M \mathrm{d}t \\ =& \int_0^\infty e^{-\beta t} \, \delta_t \widehat{U} \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top \widehat{U} \, \mathrm{d}t \\ &+ \epsilon \int_0^\infty e^{-\beta t} \, \delta_t \, \left(\widehat{U} \tilde{s}_t^{\mathbf{W}} \xi_t^\top U_{\text{eff}} M + M^\top U_{\text{eff}} \xi_t (\tilde{s}_t^{\mathbf{W}})^\top \widehat{U} \right) \mathrm{d}t \\ &+ \epsilon^2 \int_0^\infty e^{-\beta t} \, \delta_t \, M^\top U_{\text{eff}} \xi_t \xi_t^\top \, U_{\text{eff}} M \mathrm{d}t \end{aligned}$$

Therefore, for mutually orthogonal projection matrix M, and therefore $M^\top M = I_{d_s}$, substituting these terms into the expression for change in $\widehat{U}$ (Equation equation 28):

$$\Delta\widehat{U} = \eta L\Big(\widehat{\mathcal{G}}(\widehat{U}) - \widehat{U}\,\widehat{\mathcal{G}}(\widehat{U})^\top\,\widehat{U}\Big) + O(\eta^2) + O(\eta\epsilon),$$

$$\text{with}\quad \widehat{\mathcal{G}}(\widehat{U}) = \int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{s}_t^{\mathbf{W}}(\tilde{s}_t^{\mathbf{W}})^\top \mathrm{d}t.$$

Meaning that the learning dynamics are sufficiently described by the downstairs dynamics. □

F.2 Effective Dynamics of Linear Value Estimate Matrix

The matrix corresponding to the linear component of the parameterization of the value estimate, U_l', in Equation equation 6 is also a set of large $d \times d$ matrix. Similar to previous section, instead of tracking the high-dimensional matrices we can study the learning dynamics of the LQR problem with observations in the low-dimensional underlying matrices. For the linear component this can be described using the following matrix:

$$\widehat{U'} := U_L' U_{\text{eff}}' M \in \mathbb{R}^{1\times d_s}.$$

This is a $1 \times d_s$ matrix as opposed to larger matrices. We present a Lemma similar to the previous subsection on the dynamics of this lower dimensional matrix.

Lemma 5. *For learning rate η, the effective matrix $\widehat{U'}$ follows the one-step dynamic:*

$$\Delta\widehat{U'} = \eta(L-1)\Big(\widehat{\mathcal{G}}(\widehat{U'}) - \widehat{U'}\left(\widehat{\mathcal{G}}(\widehat{U'})\right)^\top\widehat{U'}\Big) + O(\eta^2) + O(\eta\epsilon),$$

$$\textit{with } \widehat{\mathcal{G}}(\widehat{U'}) := \int_0^\infty e^{-\beta t}\,\delta_t\left(\tilde{s}_t^{\mathbf{W}}\right)^\top \mathrm{d}t \in \mathbb{R}^{1\times d_s}.$$

Proof. To study the gradient updates that map to this lower-dimensional matrices we seek to simplify the equations. To do so first we note the gradients of the φ with respect to the parameters U_l' is:

$$\partial_{U_l'}\varphi(o;\mathbf{U}) = \left(\prod_{i=1}^{l-1} U_i'\right)^\top o^\top \left(\prod_{i=l+1}^{L} U_i'\right)^\top,$$

for $l \in \{1, 2, \ldots, L-1\}$, since the last linear layer, which is rectangular, is assumed frozen at initialization. Now consider the change in U_l':

$$\Delta U_l' = (U_l')^+ - U_l' \tag{29}$$

$$= \eta\left(I - \tfrac{\eta}{2}A_l\right)^{-1} A_l\,U_l', \tag{30}$$

where A_l is defined with respect to U_l' as follows: $A_l = \mathcal{G}(U_l')\,(U_l')^\top - U_l'\mathcal{G}(U_l')^\top$. Substituting this formula for A_l into the expression for change in U_l' over a gradient step is as follows:

$$\Delta U_l' = \eta A_l U_l' + O(\eta^2)$$

$$= \eta\left(\mathcal{G}(U_l')\,(U_l')^\top - U_l'\mathcal{G}(U_l')^\top\right) + O(\eta^2).$$

As in the previous section, we can derive the change in U_{eff}' over one gradient step as follows:

$$\Delta U_{\text{eff}}' = \sum_{l=1}^{L-1}\prod_{i=l+1}^{L} U_i'\,\Delta U_l'\prod_{i=1}^{l-1} U_i' + O(\eta^2)$$

$$= \eta\sum_{l=1}^{L-1}\Big[\prod_{i=l+1}^{L} U_i'\,\mathcal{G}(U_l')\prod_{i=1}^{l-1} U_i' - \prod_{i=l+1}^{L} U_i'\,U_l'\,\mathcal{G}(U_l')^\top U_l'\prod_{i=1}^{l-1} U_i'\Big] + O(\eta^2).$$

This summation can be split into two, same as in the previous section, we first consider the second expression which is the Stiefel correction.

$$\begin{aligned}\prod_{i=l+1}^{L} U_i' U_l' \mathcal{G}(U_l')^\top U_l' \prod_{i=1}^{l-1} U_i' &= \int_0^\infty e^{-\beta t}\, \delta_t\, U'_{\text{eff}}\, (U_L')^\top o_t^\top\, U'_{\text{eff}}\, \mathrm{d}t \\ &= U'_{\text{eff}} \left(\int_0^\infty e^{-\beta t}\, \delta_t (U_L')^\top\, o_t^\top\, \mathrm{d}t \right) U'_{\text{eff}} \\ &= U'_{\text{eff}} \mathcal{G}(U'_{\text{eff}})^\top U'_{\text{eff}}.\end{aligned}$$

Here the expression for $\mathcal{G}(U')$ is obtained by substituting the expression for the partial expression (Equation equation 30) into the gradient ascent expression of Equation equation 11. the second expression can be simplified as done by Arora et al. (2018):

$$\begin{aligned}\sum_{l=1}^{L-1} \Big[\prod_{i=l+1}^{L} U_i' \mathcal{G}(U_l') \prod_{i=l+1}^{L} U_i' \Big] &= \sum_{l=1}^{L-1} (U'_{\text{eff}}(U'_{\text{eff}})^\top)^{\frac{L-l}{L}} \mathcal{G}(U'_{\text{eff}}) (U'_{\text{eff}}(U'_{\text{eff}})^\top)^{\frac{l-l}{L}} \\ &= \sum_{l=1}^{L-1} (I_d)^{\frac{L-l}{L}} \mathcal{G}(U'_{\text{eff}}) (I_d)^{\frac{l-l}{L}} \\ &= (L-1) \mathcal{G}(U'_{\text{eff}}),\end{aligned}$$

where we once again use the fact that U'_{eff} lies on the Stiefel manifold. Therefore, the change in the effective matrix can be described as:

$$\begin{aligned}\Delta \widehat{U'} &= U_L' \Delta U'_{\text{eff}} M \\ &= \eta (L-1) U_L' \left(\mathcal{G}(U'_{\text{eff}}) - U'_{\text{eff}} \mathcal{G}(U'_{\text{eff}}) U_{\text{eff}} \right) M + O(\eta^2).\end{aligned} \tag{31}$$

Therefore, the change in the effective matrix can be described as:

$$\Delta U'_{\text{eff}} \;=\; \eta L \Big(\mathcal{G}(U'_{\text{eff}}) - U'_{\text{eff}}\, \mathcal{G}(U'_{\text{eff}})^\top\, U'_{\text{eff}} \Big) + O(\eta^2), \tag{32}$$

where the effective gradient is

$$\mathcal{G}(U'_{\text{eff}}) \;:=\; \int_0^\infty e^{-\beta t}\, \delta_t\, (U_L')^\top o_t^\top\, \mathrm{d}t \;\in\; \mathbb{R}^{1 \times d}.$$

Now we derive the change in the low-dimensional summary matrix $\widehat{U'} := U'_{\text{eff}} M \in \mathbb{R}^{1 \times d_s}$. Multiplying equation 32 on the right by M gives

$$\Delta \widehat{U'} \;=\; U_L' \Delta U'_{\text{eff}} M \;=\; \eta L U_L' \Big(\mathcal{G}(U'_{\text{eff}}) M - U'_{\text{eff}}\, \mathcal{G}(U'_{\text{eff}})^\top\, U'_{\text{eff}}\, M \Big) + O(\eta^2). \tag{33}$$

To express this purely in terms of downstairs quantities, we use the observation model $o_t = M \tilde{s}_t^{\mathbf{W}} + \epsilon \xi_t$, $M^\top M = I_{d_s}$ (assumption 2) and $U_L'(U_L')^\top = 1$ (from orthonormal initialization and freezing the rectangular matrix). Then

$$\begin{aligned}U_L' \mathcal{G}(U'_{\text{eff}}) M &= \int_0^\infty e^{-\beta t}\, \delta_t\, o_t^\top M\, \mathrm{d}t \\ &= \int_0^\infty e^{-\beta t}\, \delta_t\, \big(\tilde{s}_t^{\mathbf{W}}\big)^\top \mathrm{d}t \;+\; \epsilon \int_0^\infty e^{-\beta t}\, \delta_t\, \xi_t^\top M\, \mathrm{d}t \\ &= \widehat{\mathcal{G}}(\widehat{U'}) \;+\; O(\epsilon),\end{aligned}$$

where we define the downstairs gradient for the linear component of the low-dimensional summary matrix:

$$\widehat{\mathcal{G}}(\widehat{U'}) \;:=\; \int_0^\infty e^{-\beta t}\, \delta_t\, \big(\tilde{s}_t^{\mathbf{W}}\big)^\top \mathrm{d}t \;\in\; \mathbb{R}^{1 \times d_s}.$$

For the Stiefel corrective term, note that

$$\begin{aligned} U'_L U'_{\text{eff}}\, \mathcal{G}(U'_{\text{eff}})^\top &= \int_0^\infty e^{-\beta t}\, \delta_t\, U'_{\text{eff}}\, o_t \,\mathrm{d}t \\ &= \int_0^\infty e^{-\beta t}\, \delta_t \Big(\underbrace{U'_L U'_{\text{eff}} M}_{=\widehat{U'}}\, \tilde{s}_t^{\mathbf{W}} + \epsilon\, U'_L U'_{\text{eff}} \xi_t \Big) \mathrm{d}t \\ &= \widehat{U'} \left(\widehat{\mathcal{G}}(\widehat{U'}) \right)^\top \;+\; O(\epsilon). \end{aligned}$$

Fruther, we observe that

$$U'_L U'_{\text{eff}}\, \mathcal{G}(U'_{\text{eff}})^\top\, U'_{\text{eff}}\, M = U'_L \Big(U'_{\text{eff}}\, \mathcal{G}(U'_{\text{eff}})^\top \Big)\, \widehat{U'} = \widehat{U'} \left(\widehat{\mathcal{G}}(\widehat{U'}) \right)^\top \widehat{U'} \;+\; O(\epsilon).$$

Substituting back into equation 33 yields the low-dimensional analogue:

$$\Delta \widehat{U'} = \eta L \Big(\widehat{\mathcal{G}}(\widehat{U'}) - \widehat{U'} \left(\widehat{\mathcal{G}}(\widehat{U'}) \right)^\top \widehat{U'} \Big) + O(\eta^2) + O(\eta\epsilon).$$

Meaning that, up to $O(\eta\epsilon)$ terms induced by observation noise, the learning dynamics of the high-dimensional linear value matrices are described by the downstairs dynamics as presented in the statement of the Lemma. □

F.3 Effective Dynamics of the Advantage Rate Estimate: State-Action Matrix

As seen in previous sections, the dynamics of the value estimate can be described using the downstairs parameterization. Similarly, the advantage-rate estimate can also be described using the downstairs parameterization. For the mixed quadratic component of the advantage rate function (Equation equation 26) Let the effective mixed matrix be

$$Z_{\text{eff}} := \Pi_{i=2}^{L} Z_i \in \mathbb{R}^{d \times d}. \tag{34}$$

Define the forward and backward products as follows:

$$\prod_{i=l+1}^{L} Z_i := Z_L \dots Z_{l+1}, \qquad \prod_{i=2}^{l-1} Z_i := Z_{l-1} \dots Z_1,$$

so that $Z_{\text{eff}} Z_1 = \prod_{i=l+1}^{L} Z_i\, Z_l\, \prod_{i=2}^{l-1} Z_i$. Define the low-dimensional mixed matrix as follows:

$$\widehat{Z} := M^\top Z_{\text{eff}} Z_1 \in \mathbb{R}^{d_s \times d_a}. \tag{35}$$

Lemma 6. *The one-step dynamics of $\widehat{Z}$ are as follows:*

$$\Delta \widehat{Z} = \eta(L-1) \Big(\widehat{\mathrm{G}}(\widehat{Z}) - \widehat{Z}\, \widehat{\mathrm{G}}(\widehat{Z})^\top\, \widehat{Z} \Big) + O(\eta^2) + O(\eta\epsilon),$$

$$\textit{with} \quad \widehat{\mathrm{G}}(\widehat{Z}) := \int_0^\infty e^{-\beta t}\, \delta_t\, \tilde{s}_t^{\mathbf{W}}\, \tilde{a}_t^\top \,\mathrm{d}t.$$

Proof. The mixed term in equation 7 is $o^\top Z_{\text{eff}} Z_1 a$. Hence, for any layer l,

$$\partial_{Z_l} \big(o^\top Z_{\text{eff}} Z_1 a \big) = \Big(\prod_{i=l+1}^{L} Z_i \Big)^\top o\, a^\top \Big(\prod_{i=2}^{l-1} Z_i \Big)^\top. \tag{36}$$

Under the reparameterization of Section 3 the mixed term is $o^\top Z_{\text{eff}} (a - \phi(o))$, so letting $\bar{a} := a - \phi(o)$,

$$\partial_{Z_l}\, \psi(o, a; \mathbf{Z}) = \Big(\prod_{i=l+1}^{L} Z_i \Big)^\top o\, \tilde{a}^\top \Big(\prod_{i=2}^{l-1} Z_i \Big)^\top.$$

Therefore, the gradient for any of the parameter matrices in the mixed term of the advantage rate estimate is:

$$\mathrm{G}(Z_l) = \int_0^\infty e^{-\beta t}\,\delta_t\,(\prod_{i=l+1}^{L} Z_i)^\top\, o_t\,\bar{a}_t^\top\,(\prod_{i=2}^{l-1} Z_i)^\top\,\mathrm{d}t, \qquad \bar{a}_t = a_t^{\mathbf{W}} - \phi(o_t).$$

Applying the same retraction expansion as before,

$$\Delta Z_l = \eta\Big(\mathrm{G}(Z_l) - Z_l\,\mathrm{G}(Z_l)^\top\,Z_l\Big) + O(\eta^2).$$

Hence the effective change can be derived as following:

$$\begin{aligned}\Delta Z_{\text{eff}} &= \sum_{l=2}^{L}\prod_{i=l+1}^{L} Z_i\,\Delta Z_l\prod_{i=2}^{l-1} Z_i + O(\eta^2)\\ &= \eta\sum_{l=1}^{L}\Big[\prod_{i=l+1}^{L} Z_i\,\mathrm{G}(Z_l)\prod_{i=2}^{l-1} Z_i - \prod_{i=l+1}^{L} Z_i\,Z_l\,\mathrm{G}(Z_l)^\top\,Z_l\prod_{i=2}^{l-1} Z_i\Big] + O(\eta^2).\end{aligned}$$

The Stiefel correction simplifies since the gradient already contains $\prod_{i=2}^{l-1} Z_i, \prod_{i=l+1}^{L} Z_i$:

$$\begin{aligned}\prod_{i=l+1}^{L} Z_i\,Z_l\,\mathrm{G}(Z_l)^\top\,Z_l\prod_{i=2}^{l-1} Z_i &= \int_0^\infty e^{-\beta t}\,\delta_t\;Z_{\text{eff}}\,Z_1\tilde{a}_t\,o_t^\top\;Z_{\text{eff}}\;\mathrm{d}t\\ &= Z_{\text{eff}}Z_1\Big(\int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{a}_t\,o_t^\top\,\mathrm{d}t\Big)Z_{\text{eff}} \;=\; Z_{\text{eff}}\,\mathrm{G}(Z_{\text{eff}})^\top\,Z_{\text{eff}},\end{aligned}$$

where

$$\mathrm{G}(Z_{\text{eff}}) \;:=\; \int_0^\infty e^{-\beta t}\,\delta_t\,o_t\;(Z_1\tilde{a}_t)^\top\;\mathrm{d}t \;\in\; \mathbb{R}^{d\times d}.$$

Therefore following the work by Arora et al. (2018) (also see Theorem 2 by Menon (2023)) combined with all the parameter matrices being on the Stiefel manifold, the first summand satisfies

$$\sum_{l=2}^{L}\prod_{i=l+1}^{L} Z_i\,\mathrm{G}(Z_l)\prod_{i=2}^{l-1} Z_i = (L-1)\,\mathrm{G}(Z_{\text{eff}}).$$

Therefore,

$$\Delta Z_{\text{eff}} = \eta(L-1)\Big(\mathrm{G}(Z_{\text{eff}}) - Z_{\text{eff}}\,\mathrm{G}(Z_{\text{eff}})^\top\,Z_{\text{eff}}\Big) + O(\eta^2).$$

Then

$$\begin{aligned}\Delta\widehat{Z} &= M^\top\Delta Z_{\text{eff}}Z_1\\ &= \eta(L-1)\Big(M^\top\mathrm{G}(Z_{\text{eff}})Z_1 - M^\top Z_{\text{eff}}\,\mathrm{G}(Z_{\text{eff}})^\top\,Z_{\text{eff}}Z_1\Big) + O(\eta^2).\end{aligned}$$

Using $o_t = M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t$ and $M^\top M = I_{d_s}$ (assumption 1), and the orthonormal columns of Z_1 which are fixed at initialization,

$$M^\top\mathrm{G}(Z_{\text{eff}})Z_1 = \int_0^\infty e^{-\beta t}\,\delta_t\,M^\top o_t\,\tilde{a}_t^\top(Z_1)^\top Z_1\,\mathrm{d}t = \int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{s}_t^{\mathbf{W}}\,\tilde{a}_t^\top\,\mathrm{d}t \;+\; O(\epsilon).$$

Define the downstairs gradient by

$$\widehat{\mathrm{G}}(\widehat{Z}) \;:=\; M^\top\,\mathrm{G}(Z_{\text{eff}})\,Z_1 \;\in\; \mathbb{R}^{d_s\times d_a}.$$

Then the Stiefel correction term satisfies

$$\begin{aligned}M^\top Z_{\text{eff}}\,\mathrm{G}(Z_{\text{eff}})^\top\,Z_{\text{eff}}Z_1 &= \big(M^\top Z_{\text{eff}}Z_1\big)\,\big(M^\top\mathrm{G}(Z_{\text{eff}})Z_1\big)^\top\,\big(M^\top Z_{\text{eff}}Z_1\big) \;+\; O(\epsilon)\\ &= \widehat{Z}\,\widehat{\mathrm{G}}(\widehat{Z})^\top\,\widehat{Z} \;+\; O(\epsilon),\end{aligned}$$

where the $O(\epsilon)$ term comes from using $o_t = M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t$ and $M^\top M = I_{d_s}$ when identifying $M^\top \mathrm{G}(Z_{\text{eff}})Z_1 = \int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{s}_t^{\mathbf{W}}\,\tilde{a}_t^\top\,\mathrm{d}t + O(\epsilon)$. Hence,

$$\Delta\widehat{Z} = \eta(L-1)\Big(\widehat{\mathrm{G}}(\widehat{Z}) - \widehat{Z}\,\widehat{\mathrm{G}}(\widehat{Z})^\top\,\widehat{Z}\Big) + O(\eta^2) + O(\eta\epsilon),$$

$$\text{with}\quad \widehat{\mathrm{G}}(\widehat{Z}) := \int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{s}_t^{\mathbf{W}}\,\tilde{a}_t^\top\,\mathrm{d}t.$$

Meaning that the learning dynamics of the mixed component are captured by the downstairs dynamics. □

F.4 Effective Dynamics of the Advantage Rate Estimate: Linear in Action Matrix

Let the effective linear action matrix be

$$\widehat{Z'} := \prod_{i=1}^{L} Z_i' \in \mathbb{R}^{1\times d_a} = Z_L' Z_{\text{eff}}' Z_1'.$$

We present a similar lemma to above results where we show that the change in low-dimensional linear in action matrix can also be reduced to depend on the downstairs formulation. Since the proof follows the same mechanism as above, we will omit some of the steps and keep the exposition compact.

Lemma 7. *The one-step dynamics of $\widehat{Z'}$ can be re-written as:*

$$\Delta\widehat{Z'} = \eta(L-2)\Big(\mathrm{G}(\widehat{Z'}) - \widehat{Z'}\,\mathrm{G}(\widehat{Z'})^\top\,\widehat{Z'}\Big) + O(\eta^2),$$

$$\textit{with } \mathrm{G}(\widehat{Z'}) := \int_0^\infty e^{-\beta t}\,\delta_t\,\tilde{a}_t^\top\,\mathrm{d}t \in \mathbb{R}^{1\times d_a}.$$

Proof. Define the forward and backward products as follows:

$$\prod_{i=l+1}^{L-1} Z_i' := Z_{L-1}' \dots Z_{l+1}', \qquad \prod_{i=2}^{l-1} Z_i' := Z_1' \dots Z_{l-1}',$$

so that $\widehat{Z'} = Z_L' \prod_{i=l+1}^{L-1} Z_i'\, Z_l' \prod_{i=2}^{l-1} Z_i' Z_1'$.

The corresponding term in equation 7 is $\widehat{Z'}a$, and under the re-parameterization (Section 3) it becomes $\widehat{Z'}\bar{a}$, where $\bar{a} := a - \phi(o)$. Hence, for any $l \in \{2, \dots, L-1\}$,

$$\partial_{Z_l'}\big(\widehat{Z'}\tilde{a}\big) = \Big(\prod_{i=l+1}^{L-1} Z_i'\Big)^\top \bar{a}^\top \Big(\prod_{i=2}^{l-1} Z_i'\Big)^\top. \tag{37}$$

Therefore the vanilla gradient is

$$\mathrm{G}(Z_l') = \int_0^\infty e^{-\beta t}\,\delta_t\,\Big(\prod_{i=l+1}^{L-1} Z_i'\Big)^\top \bar{a}_t^\top \Big(\prod_{i=2}^{l-1} Z_i'\Big)^\top \mathrm{d}t, \qquad \bar{a}_t := a_t^{\mathbf{W}} - \phi(o_t^{\mathbf{W}}).$$

Applying the same retraction expansion as before gives

$$\Delta Z_l' = \eta\Big(\mathrm{G}(Z_l') - Z_l'\,\mathrm{G}(Z_l')^\top\,Z_l'\Big) + O(\eta^2).$$

Thus the change in the effective matrix is

$$\begin{aligned}\Delta Z_{\text{eff}}' &= \sum_{l=2}^{L-1} \prod_{i=l+1}^{L-1} Z_i'\,\Delta Z_l' \prod_{i=2}^{l-1} Z_i' + O(\eta^2)\\ &= \eta \sum_{l=1}^{L} \Big[\prod_{i=l+1}^{L-1} Z_i'\,\mathrm{G}(Z_l') \prod_{i=2}^{l-1} Z_i' - \prod_{i=l+1}^{L-1} Z_i'\,Z_l'\,\mathrm{G}(Z_l')^\top\,Z_l' \prod_{i=2}^{l-1} Z_i'\Big] + O(\eta^2).\end{aligned}$$

Since the expressions are all the same as in previous proofs, using similar argumeents as above using the mutual orthogonality of Z'_L, Z'_1 and the results by Arora et al. (2018) we have:

$$\Delta\widehat{Z'} = \eta(L-2)\Big(\mathrm{G}(\widehat{Z'}) - \widehat{Z'}\,\mathrm{G}(\widehat{Z'})^\top\,\widehat{Z'}\Big) + O(\eta^2),$$

implying that the low-dimensional effective linear-in-action component has closed dynamics. □

F.5 Effective Dynamics of the Policy Matrix

The deep linear policy is defined as

$$\phi(o;\mathbf{W}) \;=\; W_L W_{\text{eff}}\, o, \qquad W_{\text{eff}} := \prod_{i=1}^{L-1} W_i \in \mathbb{R}^{d_a\times d}. \tag{38}$$

Lemma 8. *The one-step dynamics of $\widehat{W}$ are defined as:*

$$\Delta\widehat{W} = \eta(L-1)\Big(\widehat{\mathbb{G}}(\widehat{W}) - \widehat{W}\,\widehat{\mathbb{G}}(\widehat{W})^\top\,\widehat{W}\Big) + O(\eta^2) + O(\eta\epsilon),$$
$$\textit{with } \widehat{\mathbb{G}}(\widehat{W}) = \int_0^\infty e^{-\beta t}\left(\widehat{Z}^\top \tilde{s}_t^{\mathbf{W}} + \widehat{Z'}^\top\right)(\tilde{s}_t^{\mathbf{W}})^\top\,\mathrm{d}t \;+\; O(\epsilon).$$

Proof. Once again the forward and backward products so that $W_{\text{eff}} = \prod_{i=l+1}^{L-1} W_i\, W_l\, \prod_{i=1}^{l-1} W_i$. Let

$$g_t \;:=\; \partial_a \psi\big(o_t^{\mathbf{W}}, a; \mathbf{Z}\big)\Big|_{a=\phi(o_t^{\mathbf{W}};\mathbf{W})} \in \mathbb{R}^{d_a},$$

then substituing the Jacobian $\partial_{W_l}\phi(o_t^{\mathbf{W}};\mathbf{W})$ into the expression for the gradient of the policy parameters in Equation equation 11 alongside g_t gives

$$\mathbb{G}(W_l) = \int_0^\infty e^{-\beta t}\,(\prod_{i=l+1}^{L-1} W_i)^\top W_L^\top\, g_t\,(o_t^{\mathbf{W}})^\top\,(\prod_{i=1}^{l-1} W_i)^\top\,\mathrm{d}t \in \mathbb{R}^{d\times d}. \tag{39}$$

With the Stiefel retraction expansion (as in the value and advantage rate sections),

$$\Delta W_l = \eta\Big(\mathbb{G}(W_l) - W_l\,\mathbb{G}(W_l)^\top\,W_l\Big) + O(\eta^2).$$

Therefore the effective matrix changes as

$$\begin{aligned}\Delta W_{\text{eff}} &= \sum_{l=1}^{L-1}\prod_{i=l+1}^{L-1} W_i\,\Delta W_l\prod_{i=1}^{l-1} W_i \\ &= \eta\sum_{l=1}^{L-1}\Big[\prod_{i=l+1}^{L-1} W_i\,\mathbb{G}(W_l)\prod_{i=1}^{l-1} W_i - \prod_{i=l+1}^{L-1} W_i\,W_l\,\mathbb{G}(W_l)^\top\,W_l\prod_{i=1}^{l-1} W_i\Big] + O(\eta^2).\end{aligned}$$

For the Stiefel corrective term, using equation 39 and $\prod_{i=l+1}^{L-1} W_i W_l \prod_{i=1}^{l-1} W_i = W_{\text{eff}}$,

$$\begin{aligned}\prod_{i=l+1}^{L-1} W_i\,W_l\,\mathbb{G}(W_l)^\top\,W_l\prod_{i=1}^{l-1} W_i &= \int_0^\infty e^{-\beta t}\,W_{\text{eff}}\,o_t^{\mathbf{W}}\,g_t^\top W_L\,W_{\text{eff}}\,\mathrm{d}t \\ &= W_{\text{eff}}\left(\int_0^\infty e^{-\beta t}\,o_t^{\mathbf{W}}\,g_t^\top W_L\,\mathrm{d}t\right)W_{\text{eff}} \;=\; W_{\text{eff}}\,W_L^\top\mathbb{G}(W_{\text{eff}})^\top\,W_{\text{eff}},\end{aligned}$$

where

$$\mathbb{G}(W_{\text{eff}}) \;:=\; \int_0^\infty e^{-\beta t}\,g_t\,(o_t^{\mathbf{W}})^\top\,\mathrm{d}t \;\in\; \mathbb{R}^{d_a\times d}.$$

Further, as in Arora et al. (2018) and Theorem 2 by Menon (2023) combined with all the parameter matrices being on the Stiefel manifold,

$$\sum_{l=1}^{L-1}\prod_{i=l+1}^{L-1} W_i\,\mathbb{G}(W_l)\prod_{i=1}^{l-1} W_i = (L-1)\,\mathbb{G}(W_{\text{eff}}).$$

Hence,

$$\Delta W_{\text{eff}} = \eta(L-1)\Big(\mathbb{G}(W_{\text{eff}}) - W_{\text{eff}}\,\mathbb{G}(W_{\text{eff}})^\top W_{\text{eff}}\Big) + O(\eta^2).$$

Now we derive the downstairs dynamics for this. Remember that the low-dimensional policy matrix is defined as:

$$\widehat{W} \;:=\; W_{\text{eff}}\,M \in \mathbb{R}^{d_a\times d_s}.$$

Then

$$\begin{aligned}\Delta\widehat{W} &= \Delta W_{\text{eff}}\,M \\ &= \eta(L-1)\Big(\mathbb{G}(W_{\text{eff}})M - W_{\text{eff}}\,\mathbb{G}(W_{\text{eff}})^\top W_{\text{eff}}\,M\Big) + O(\eta^2).\end{aligned}$$

Using $o_t^{\mathbf{W}} = M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t$, columns of W_L being orthonormal, and $M^\top M = I_{d_s}$ (Assumption 2),

$$\begin{aligned}W_L\mathbb{G}(W_{\text{eff}})M &= \int_0^\infty e^{-\beta t}\,W_L^\top g_t\,(o_t^{\mathbf{W}})^\top M\,\mathrm{d}t \\ &= \int_0^\infty e^{-\beta t}\,g_t\,(\tilde{s}_t^{\mathbf{W}})^\top\,\mathrm{d}t \;+\; O(\epsilon) \;:=\; \widehat{\mathbb{G}}(\widehat{W}) \;+\; O(\epsilon),\end{aligned}$$

where

$$\widehat{\mathbb{G}}(\widehat{W}) \;:=\; \int_0^\infty e^{-\beta t}\,g_t\,(\tilde{s}_t^{\mathbf{W}})^\top\,\mathrm{d}t \;\in\; \mathbb{R}^{d_a\times d_s}.$$

Moreover, since $(I - MM^\top)o_t^{\mathbf{W}} = O(\epsilon)$, the correction term reduces to the downstairs form up to $O(\epsilon)$:

$$W_{\text{eff}}\,\mathbb{G}(W_{\text{eff}})^\top W_{\text{eff}}\,M = \widehat{W}\,\widehat{\mathbb{G}}(\widehat{W})^\top\,\widehat{W} \;+\; O(\epsilon).$$

Therefore,

$$\Delta\widehat{W} = \eta(L-1)\Big(\widehat{\mathbb{G}}(\widehat{W}) - \widehat{W}\,\widehat{\mathbb{G}}(\widehat{W})^\top\,\widehat{W}\Big) + O(\eta^2) + O(\eta\epsilon).$$

Meaning that the learning dynamics of the high-dimensional policy matrices are described by the downstairs dynamics.

Policy Gradient Reparameterization with Low-Dimensional Advantage Rate Using the advantage parametrization and the re-parameterization $\psi(o,a) = \Psi(o,a) - \Psi(o,\phi(o))$, the only terms that depend on a are the mixed and linear-in-action parts:

$$\psi(o,a;\mathbf{Z}) = o^\top Z_{\text{eff}}\,Z_1\,(a-\phi(o)) + \widehat{Z'}\,(a-\phi(o)),$$

where $Z_{\text{eff}} = \prod_{i=2}^{L} Z_i$ and $\widehat{Z'} = \prod_{i=1}^{L} Z_i' \in \mathbb{R}^{1\times d_a}$. Therefore

$$g_t \;:=\; \partial_a\psi(o_t^{\mathbf{W}},a;\mathbf{Z})\Big|_{a=\phi(o_t^{\mathbf{W}};\mathbf{W})} = Z_1^\top Z_{\text{eff}}^\top o_t^{\mathbf{W}} + \widehat{Z'}^\top.$$

Now substitute the observation model $o_t^{\mathbf{W}} = M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t$ and the downstairs mixed parameter $\widehat{Z} := M^\top Z_{\text{eff}}$:

$$Z_1^\top Z_{\text{eff}}^\top o_t^{\mathbf{W}} = Z_1^\top Z_{\text{eff}}^\top M\,\tilde{s}_t^{\mathbf{W}} + \epsilon\,Z_1^\top Z_{\text{eff}}^\top\xi_t = \widehat{Z}^\top\tilde{s}_t^{\mathbf{W}} + O(\epsilon),$$

hence

$$g_t = \widehat{Z}^\top\tilde{s}_t^{\mathbf{W}} + \widehat{Z'}^\top + O(\epsilon). \tag{40}$$

Plugging equation 40 into the downstairs policy gradient $\widehat{\mathbb{G}}(\widehat{W}) = \int_0^\infty e^{-\beta t} g_t(\tilde{s}_t^{\mathbf{W}})^\top \mathrm{d}t$, we obtain

$$\widehat{\mathbb{G}}(\widehat{W}) = \int_0^\infty e^{-\beta t} \left(\widehat{Z}^\top \tilde{s}_t^{\mathbf{W}} + \widehat{Z'}^\top \right) (\tilde{s}_t^{\mathbf{W}})^\top \, \mathrm{d}t \; + \; O(\epsilon),$$

and therefore the downstairs update becomes

$$\Delta \widehat{W} = \eta(L-1) \Big(\widehat{\mathbb{G}}(\widehat{W}) - \widehat{W} \, \widehat{\mathbb{G}}(\widehat{W})^\top \, \widehat{W} \Big) + O(\eta^2) + O(\eta\epsilon),$$

where $\widehat{W} = W_L W_{\text{eff}} M$. This gives us the required expression.

□

F.6 Low-Dimensional Structure of the TD-Error

We have shown in all preceding lemmas that the learning dynamics has an underlying low-dimensional structure because the one-step parameter time dynamics of the summarized matrices are only dependent on other summarized matrices and the TD-error δ_t. We further would like to show that this TD-error is dependent on the underlying matrices up to an error term.

Lemma 9. *The continuous time TD-error* $\delta_t = \varphi(o_t^{\mathbf{W}}; \mathbf{U}) - \int_t^{t+\ell} e^{-\beta(u-t)} \left[r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \psi(o_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \mathbf{Z}) \right] \mathrm{d}u \; - \; e^{-\beta\ell} \varphi(o_{t+\ell}^{\mathbf{W}}; \mathbf{U})$ *in its upstairs formulation can be re-written in the downstairs formulation up to an error of* $O(\epsilon)$*:*

$$\delta_t = \bar{\varphi}(o_t^{\mathbf{W}}; \widehat{U}) - \int_t^{t+\ell} e^{-\beta(u-t)} \left[r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \bar{\Psi}(o_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \widehat{Z}, \widehat{Z'}) \right] \mathrm{d}u - e^{-\beta\ell} \bar{\varphi}(o_{t+\ell}^{\mathbf{W}}; \widehat{U}, \widehat{U'}) + O(\epsilon).$$

Proof. Consider the error δ_t we would like to "de-construct" it into the underlying low-dimensional structure:

$$\delta_t = \varphi(o_t^{\mathbf{W}}; \mathbf{U}) - \int_t^{t+\ell} e^{-\beta(u-t)} \left[r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \psi(o_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \mathbf{Z}) \right] \mathrm{d}u - e^{-\beta\ell} \varphi(o_{t+\ell}^{\mathbf{W}}; \mathbf{U}).$$

To simplify this TD-error the advantage rate as a function of the state, as opposed to the observation, is defined as follows:

$$\bar{\psi}(s, a; \widehat{Z}, \widehat{Z'}) \; := \; s^\top \widehat{Z} \, (a - \widehat{W} \, s) \; + \; \widehat{Z'} \, (a - \widehat{W} \, s).$$

The advantage rate function in the upstairs formulation is defined as

$$\begin{aligned} \psi(o, a; \mathbf{Z}) &:= \Psi(o, a; \mathbf{Z}) - \Psi(o, \phi(o); \mathbf{Z}) \\ &= o^\top Z_{\text{eff}} \big(a - \phi(o) \big) \; + \; \widehat{Z'} \big(a - \phi(o) \big). \end{aligned}$$

The action as a function of the observation at time t is:

$$\phi(o_t^{\mathbf{W}}) \; = \; W_{\text{eff}} \, o_t^{\mathbf{W}} \; = \; W_{\text{eff}} (M \tilde{s}_t^{\mathbf{W}} + \epsilon \xi_t) \; = \; \widehat{W} \, \tilde{s}_t^{\mathbf{W}} \; + \; \epsilon \, W_{\text{eff}} \, \xi_t.$$

The centered action, according to the reparameterization of the advantage function, is as follows:

$$a_t^{\mathbf{W}} - \phi(o_t^{\mathbf{W}}) \; = \; \bar{\Sigma}(\mathbf{W}) \, \zeta_t \; - \; \epsilon \, W_{\text{eff}} \, \xi_t.$$

To simplify our notation we define d_a-dimensional Gaussian projections of the observation noise which will feature in the following analysis of the TD-error:

$$\upsilon_t \; := \; Z_1^\top Z_{\text{eff}}^\top \xi_t \in \mathbb{R}^{d_a}, \qquad \kappa_t \; := \; W_L W_{\text{eff}} \, \xi_t \in \mathbb{R}^{d_a}.$$

where Z_{eff} is defined in Equation equation 34, W_{eff} is defined in Equation equation 38, and ξ_t is the observation noise as defined in Section A.7. With these notations consider the advantage rate function in the upstairs formulation by plugging $o_t = M s_t + \epsilon \xi_t$ and the residual into ψ:

$$\begin{aligned} \psi(o_t^{\mathbf{W}}, a_t^{\mathbf{W}}; \mathbf{Z}) &= (o_t^{\mathbf{W}})^\top Z_{\text{eff}} Z_1 \left(a_t^{\mathbf{W}} - \phi(o_t^{\mathbf{W}}) \right) \; + \; \widehat{Z'} \big(a_t^{\mathbf{W}} - \phi(o_t^{\mathbf{W}}) \big) \\ &= \big(M \tilde{s}_t^{\mathbf{W}} + \epsilon \xi_t \big)^\top Z_{\text{eff}} Z_1 \left(\bar{\Sigma}(\mathbf{W}) \zeta_t - \epsilon W_L W_{\text{eff}} \xi_t \right) \; + \; \widehat{Z'} \big(\bar{\Sigma}(\mathbf{W}) \zeta_t - \epsilon W_L W_{\text{eff}} \xi_t \big) \\ &= \Big((\tilde{s}_t^{\mathbf{W}})^\top \widehat{Z} \; + \; \epsilon \, \upsilon_t^\top \Big) \Big(\bar{\Sigma}(\mathbf{W}) \zeta_t \; - \; \epsilon \, \kappa_t \Big) \; + \; \widehat{Z'} \Big(\bar{\Sigma}(\mathbf{W}) \zeta_t \; - \; \epsilon \, \kappa_t \Big), \end{aligned} \tag{41}$$

where $\widehat{Z} = M^\top Z_{\text{eff}}$ (Equation equation 35). This expression simplifies all the quantities of the advantage rate function in lower-dimensional summary vectors and matrices. To do the same for the value estimate function in the upstairs formulation, we define the following summary random variables:

$$\chi_t \;:=\; M^\top\big(U_{\text{eff}} + U_{\text{eff}}^\top\big)\xi_t \in \mathbb{R}^{d_s}, \qquad \nu_t \;:=\; U'_{\text{eff}}\,\xi_t \in \mathbb{R}, \qquad \rho_t \;:=\; \xi_t^\top U_{\text{eff}}\,\xi_t \in \mathbb{R}.$$

Once again substituting the expression for $o_t^{\mathbf{W}}$ into the value estimate formulation, we have the following:

$$\begin{aligned}\varphi(o_t^{\mathbf{W}};\mathbf{U}) &= (M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t)^\top U_{\text{eff}}(M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t) \;+ U'_L\; U'_{\text{eff}}(M\tilde{s}_t^{\mathbf{W}} + \epsilon\xi_t) \\ &= (\tilde{s}_t^{\mathbf{W}})^\top\widehat{U}\,\tilde{s}_t^{\mathbf{W}} \;+\; \widehat{U'}\,\tilde{s}_t^{\mathbf{W}} \;+\; \epsilon\,(\tilde{s}_t^{\mathbf{W}})^\top\chi_t \;+\; \epsilon\,\nu_t \;+\; \epsilon^2\,\rho_t. \end{aligned} \tag{42}$$

This simplifies the expression for the value function in the TD-error into low-dimensional expressions. Let $A_{\text{cl}}(\widehat{W}) = G + H\widehat{W}$ (as in Section 4) and we will omit the argument $\widehat{W}$ for the sake of succinctness. Let $T := t + \ell$.

For compactness, write

$$s_u := \tilde{s}_u^{\mathbf{W}}, \qquad a_u := a_u^{\mathbf{W}}, \qquad \eta_u := a_u - \widehat{W}s_u.$$

By the policy parametrization,

$$\eta_u = \bar{\Sigma}(\mathbf{W})\zeta_u.$$

Hence the downstairs advantage rate is

$$\bar{\psi}(s_u, a_u; \widehat{Z}, \widehat{Z'}) = s_u^\top\widehat{Z}\eta_u + \widehat{Z'}\eta_u.$$

Using equation 41, we obtain

$$\begin{aligned}\psi(o_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \mathbf{Z}) &= \Big(s_u^\top\widehat{Z} + \epsilon\upsilon_u^\top\Big)\Big(\eta_u - \epsilon\kappa_u\Big) + \widehat{Z'}\Big(\eta_u - \epsilon\kappa_u\Big) \\ &= s_u^\top\widehat{Z}\eta_u + \widehat{Z'}\eta_u \\ &\quad + \epsilon\left[\upsilon_u^\top\eta_u - s_u^\top\widehat{Z}\kappa_u - \widehat{Z'}\kappa_u\right] - \epsilon^2\upsilon_u^\top\kappa_u \\ &= \bar{\psi}(s_u, a_u; \widehat{Z}, \widehat{Z'}) + \epsilon\mathcal{R}_u^A, \end{aligned} \tag{43}$$

where

$$\mathcal{R}_u^A := \upsilon_u^\top\eta_u - s_u^\top\widehat{Z}\kappa_u - \widehat{Z'}\kappa_u - \epsilon\upsilon_u^\top\kappa_u.$$

Thus the upstairs advantage rate equals the downstairs advantage rate up to an $O(\epsilon)$ error.

Similarly, define the downstairs value estimate by

$$\bar{\varphi}(s; \widehat{U}, \widehat{U'}) := s^\top\widehat{U}s + \widehat{U'}s.$$

Then equation 42 gives

$$\begin{aligned}\varphi(o_u^{\mathbf{W}};\mathbf{U}) &= s_u^\top\widehat{U}s_u + \widehat{U'}s_u + \epsilon s_u^\top\chi_u + \epsilon\nu_u + \epsilon^2\rho_u \\ &= \bar{\varphi}(s_u; \widehat{U}, \widehat{U'}) + \epsilon\mathcal{R}_u^V, \end{aligned} \tag{44}$$

where

$$\mathcal{R}_u^V := s_u^\top\chi_u + \nu_u + \epsilon\rho_u.$$

Substituting equation 43 and equation 44 into the TD-error gives

$$\begin{aligned}\delta_t &= \varphi(o_t^{\mathbf{W}};\mathbf{U}) - \int_t^T e^{-\beta(u-t)}\left[r(s_u,a_u) - \psi(o_u^{\mathbf{W}},a_u^{\mathbf{W}};\mathbf{Z})\right]\mathrm{d}u - e^{-\beta\delta}\varphi(o_T^{\mathbf{W}};\mathbf{U}) \\ &= \bar{\varphi}(s_t;\widehat{U},\widehat{U'}) + \epsilon\mathcal{R}_t^V \\ &\quad - \int_t^T e^{-\beta(u-t)}\left[r(s_u,a_u) - \bar{\psi}(s_u,a_u;\widehat{Z},\widehat{Z'}) - \epsilon\mathcal{R}_u^A\right]\mathrm{d}u \\ &\quad - e^{-\beta\delta}\left[\bar{\varphi}(s_T;\widehat{U},\widehat{U'}) + \epsilon\mathcal{R}_T^V\right] \\ &= \bar{\varphi}(s_t;\widehat{U},\widehat{U'}) - \int_t^T e^{-\beta(u-t)}\left[r(s_u,a_u) - \bar{\psi}(s_u,a_u;\widehat{Z},\widehat{Z'})\right]\mathrm{d}u - e^{-\beta\delta}\bar{\varphi}(s_T;\widehat{U},\widehat{U'}) \\ &\quad + \epsilon\left[\mathcal{R}_t^V + \int_t^T e^{-\beta(u-t)}\mathcal{R}_u^A\mathrm{d}u - e^{-\beta\delta}\mathcal{R}_T^V\right]. \end{aligned} \tag{45}$$

It remains to control the last line. On any finite bootstrapping interval $[t,T]$, assume the summarized state process, the policy covariance, and the effective matrices appearing above have bounded norm, and that the noise projections $\upsilon_u,\kappa_u,\chi_u,\nu_u,\rho_u$ are bounded in the relevant sense, either almost surely on the event under consideration or in L^p for the desired moment bound. Then

$$\mathcal{R}_t^V + \int_t^T e^{-\beta(u-t)}\mathcal{R}_u^A\mathrm{d}u - e^{-\beta\delta}\mathcal{R}_T^V = O(1).$$

Therefore,

$$\epsilon\left[\mathcal{R}_t^V + \int_t^T e^{-\beta(u-t)}\mathcal{R}_u^A\mathrm{d}u - e^{-\beta\delta}\mathcal{R}_T^V\right] = O(\epsilon).$$

Combining this with equation 45, we obtain

$$\begin{aligned}\delta_t = \bar{\varphi}(s_t;\widehat{U},\widehat{U'}) &- \int_t^{t+\ell} e^{-\beta(u-t)}\left[r(s_u,a_u) - \bar{\psi}(s_u,a_u;\widehat{Z},\widehat{Z'})\right]\mathrm{d}u \\ &- e^{-\beta\delta}\bar{\varphi}(s_{t+\ell};\widehat{U},\widehat{U'}) + O(\epsilon).\end{aligned}$$

This is precisely the desired downstairs representation of the TD-error up to an $O(\epsilon)$ perturbation.

Finally, the only quantities appearing in the leading-order term are the summarized matrices

$$\widehat{W}, \qquad \widehat{U}, \qquad \widehat{U'}, \qquad \widehat{Z}, \qquad \widehat{Z'},$$

together with the low-dimensional trajectory (s_u,a_u). Moreover, this trajectory is itself determined by the closed-loop low-dimensional dynamics

$$\dot{s}_u = A_{\mathrm{cl}}s_u + H\bar{\Sigma}(\mathbf{W})\zeta_u, \qquad a_u = \widehat{W}s_u + \bar{\Sigma}(\mathbf{W})\zeta_u.$$

Equivalently,

$$s_u = e^{A_{\mathrm{cl}}(u-t)}s_t + \int_t^u e^{A_{\mathrm{cl}}(u-\tau)}H\bar{\Sigma}(\mathbf{W})\zeta_\tau\mathrm{d}\tau.$$

Thus the leading-order TD-error is fully determined by the downstairs summarized matrices and the low-dimensional exogenous exploration noise.

□

F.7 Proof of Theorem 3

Here we present the proof of one step gradient update dynamics theorem.

Theorem 3. *For the control problem with high-dimensional observations with low-dimensional latent dynamics, and with deep linear networks parameterizing the controller, value estimate, and advantage rate estimate, such that the observation mapping has orthonormal columns, with non-degenerate exploration and under the discounted stability assumption with Stiefel retraction at learning rate*

η and observation noise coefficient ϵ, the change in the objective function over one gradient step depends only on the underlying low-dimensional matrices $\{\widehat{U}, \widehat{U'}, \widehat{Z}, \widehat{Z'}, \widehat{W}\}$, the scalar variable c for the value estimate, and the following downstairs formulation of the TD error:

$$\bar{\delta}_t = \bar{\varphi}(\tilde{s}_t^{\mathbf{W}}; \widehat{U}, \widehat{U'}) - \int_t^{t+\ell} e^{-\beta(u-t)} \left[r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \bar{\psi}(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \widehat{Z'}, \widehat{Z}) \right] \mathrm{d}u - e^{-\beta\ell} \bar{\varphi}(s_t^{\mathbf{W}}; \widehat{U}, \widehat{U'}),$$

up to an error of $O(\eta\epsilon)$. Moreover, the one-step gradient updates are:

$$\Delta\widehat{U} = \eta L \Big(\widehat{\mathcal{G}}(\widehat{U}) - \widehat{U}\, \widehat{\mathcal{G}}(\widehat{U})^\top\, \widehat{U} \Big) + O(\eta^2) + O(\eta\epsilon), \quad \widehat{\mathcal{G}}(\widehat{U}) = \int_0^\infty e^{-\beta t}\, \bar{\delta}_t\, \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top \mathrm{d}t,$$

$$\Delta\widehat{U'} = \eta(L-1) \Big(\widehat{\mathcal{G}}(\widehat{U'}) - \widehat{U'} \left(\widehat{\mathcal{G}}(\widehat{U'}) \right)^\top \widehat{U'} \Big) + O(\eta^2) + O(\eta\epsilon), \quad \widehat{\mathcal{G}}(\widehat{U'}) = \int_0^\infty e^{-\beta t}\, \bar{\delta}_t\, (\tilde{s}_t^{\mathbf{W}})^\top \mathrm{d}t,$$

$$\Delta\widehat{Z} = \eta(L-1) \Big(\widehat{\mathrm{G}}(\widehat{Z}) - \widehat{Z}\, \widehat{\mathrm{G}}(\widehat{Z})^\top\, \widehat{Z} \Big) + O(\eta^2) + O(\eta\epsilon), \quad \widehat{\mathrm{G}}(\widehat{Z}) = \int_0^\infty e^{-\beta t}\, \bar{\delta}_t\, \tilde{s}_t^{\mathbf{W}}\, \tilde{a}_t^\top \mathrm{d}t,$$

$$\Delta\widehat{Z'} = \eta(L-2) \Big(\widehat{\mathrm{G}}(\widehat{Z'}) - \widehat{Z'}\, \widehat{\mathrm{G}}(\widehat{Z'})^\top\, \widehat{Z'} \Big) + O(\eta^2) + O(\eta\epsilon), \quad \widehat{\mathrm{G}}(\widehat{Z'}) = \int_0^\infty e^{-\beta t}\, \bar{\delta}_t\, \tilde{a}_t^\top \mathrm{d}t,$$

$$\Delta\widehat{W} = \eta(L-1) \Big(\widehat{\mathbb{G}}(\widehat{W}) - \widehat{W}\, \widehat{\mathbb{G}}(\widehat{W})^\top\, \widehat{W} \Big) + O(\eta^2) + O(\eta\epsilon),$$
$$\widehat{\mathbb{G}}(\widehat{W}) = \int_0^\infty e^{-\beta t} \left(\widehat{Z}^\top \tilde{s}_t^{\mathbf{W}} + \widehat{Z'}^\top \right) (\tilde{s}_t^{\mathbf{W}})^\top \mathrm{d}t,$$

where the dependence of the gradients on the underlying low-dimensional matrices are omitted for succinctness. Finally, the variable c has the following one-step dynamic:

$$\Delta c = \eta \left[\left(1 - e^{-\beta\ell}\right) \int_0^\infty e^{-\beta t}\, \bar{\delta}_t \, \mathrm{d}t \right] + O(\eta^2) + O(\eta\epsilon).$$

Proof. First note that for the dynamics described in Equation equation 12, under Assumption 1, the objective function can be simplified. To do so first note the following:

$$\begin{aligned} \widehat{W} &:= \left(\Pi_{l=1}^L W_l \right) M, \\ A_{\mathrm{cl}}(\widehat{W}) &:= G + H\widehat{W}, \\ Q_{\mathrm{cl}}(\widehat{W}) &:= Q + \widehat{W}^\top R \widehat{W}. \end{aligned}$$

Define $C(\widehat{W})$ as the unique symmetric solution of the discounted Lyapunov equation

$$A_{\mathrm{cl}}(\widehat{W})^\top C(\widehat{W}) + C(\widehat{W}) A_{\mathrm{cl}}(\widehat{W}) - \beta\, C(\widehat{W}) + Q_{\mathrm{cl}}(\widehat{W}) = 0.$$

For deterministic $x_0 = s_0$, the infinite-horizon discounted cumulative reward is therefore

$$J(\mathbf{W}) = \mathbb{E}\left[\int_0^\infty e^{-\beta t}\, x_t^\top Q_{\mathrm{cl}}(\widehat{W})\, x_t \, \mathrm{d}t \right] = s_0^\top P(\widehat{W})\, s_0 + \frac{1}{\beta}\, \mathrm{tr}\Big(C(\widehat{W})\, \Sigma(\mathbf{W}) \Sigma(\mathbf{W})^\top \Big). \quad (46)$$

The above equation follows from solving the infinite-horizon discounted quadratic cost for a linear stochastic system (Bertsekas, 2012). Since $\Sigma(\mathbf{W})$ is the principal square root of $\sigma^2 H H^\top + \Sigma\Sigma^\top + \epsilon^2 H \left(\Pi_{l=1}^L W_l \right) \left(\Pi_{l=1}^L W_l \right)^\top H$ we have:

$$\Sigma(\mathbf{W}) \Sigma(\mathbf{W})^\top = \sigma^2 H H^\top + \Sigma\Sigma^\top + \epsilon^2 H \left(\Pi_{l=1}^L W_l \right) \left(\Pi_{l=1}^L W_l \right)^\top H^\top,$$

further we have that W_l have mutually orthogonal columns and therefore we can simplify this to:

$$\Sigma(\mathbf{W}) \Sigma(\mathbf{W})^\top = \sigma^2 H H^\top + \Sigma\Sigma^\top + \epsilon^2 H W_L W_L^\top H^\top,$$

and is therefore independent of the trainable policy parameter matrices. Together, we observe that the discounted return (Equation equation 46) depend on $\widehat{W}$.

The one-step dynamics of $\widehat{W}$ can be described in a closed form using the matrices: $\widehat{U}, \widehat{U'}, \widehat{Z}, \widehat{Z'}$ by Lemma 8 and the dynamics of these matrices can be described in a closed form by one another (See Lemmas 4, 5, 6, 7). The gradient update for c is trivial.

Finally, substituting the breakdown of the TD-error δ_t into its downstairs formulation $\bar{\delta}$, and the remainder $\mathcal{R}_t$ (Lemma 9) gives us the expressions in the statement of the theorem. □

Note that $\tilde{s}_t^{\mathbf{W}}$ is the state random variable dependent on time which can also be denoted as $\tilde{s}_t^{\widehat{W}}$ since Equation $equation$ 12, which governs the dynamics of the variable, can be re-written using only the downstairs policy parameterization, $\widehat{W}$. Therefore, all variables can define the one-step learning dynamics. This leads to the theorem for learning dynamics in the limit of diminishing learning rate in the following section.

G Proof of Main Result: Stiefel Manifold Optimization Learning Dynamics

Given the one step update (Theorem 3) for non-zero learning rate η, and observation noise scale ϵ. For high-dimensional settings, one way to theoretically study the learning dynamics in continuous parameter time is obtained by taking the limit $\eta \to 0$ (Kushner & Yin, 2003; Borkar et al., 2021) as the dimensionality of the data approaches ∞ (Ben Arous et al., 2022). We first prove how learning converges to an ordinary differential equation (ODE) in the limit and then prove *local convergence* of the parameters. This section is structured as follows: firstly we bound the moments and alsoshow global Lipschitz continuity for the mean of the gradients, secondly we show how the learning process converges to an ODE with low-dimensional underlying structure.

G.1 Bounding the Moments of Random Variables for LQR

First, we prove moment bounds on the state and action random variables for matrices on the Stiefel manifold.

Lemma 10 (First-moment bound under discounted stability)**.** *For $\widehat{W} \in \mathrm{St}(d_a, d_s)$,let*

$$m_s(t) := \mathbb{E}[\tilde{s}_t^{\widehat{W}}], \qquad m_a(t) := \mathbb{E}[\tilde{a}_t^{\widehat{W}}].$$

Then

$$m_s(t) = e^{A_{\mathrm{cl}}(\widehat{W})t} s_0, \qquad m_a(t) = \widehat{W} e^{A_{\mathrm{cl}}(\widehat{W})t} s_0.$$

Moreover, under the discounted stability assumption, if there exists $\gamma > 0$ such that

$$\alpha(A_{\mathrm{cl}}(\widehat{W})) \le \frac{\beta}{2} - \gamma,$$

where $\alpha(A) := \max_{\lambda \in \mathrm{spec}(A)} \Re(\lambda)$, then there exists a constant $C < \infty$ such that

$$\|m_s(t)\| \le C e^{(\beta/2-\gamma)t} \|s_0\|,$$

and, since $\|\widehat{W}\|_{\mathrm{op}} = 1$ on the Stiefel manifold,

$$\|m_a(t)\| \le C e^{(\beta/2-\gamma)t} \|s_0\|.$$

Consequently,

$$e^{-\beta t}\|m_s(t)\|^2 \le C^2 e^{-2\gamma t}\|s_0\|^2, \qquad e^{-\beta t}\|m_a(t)\|^2 \le C^2 e^{-2\gamma t}\|s_0\|^2.$$

In particular, both discounted first-moment-square quantities are integrable over $[0, \infty)$.

Proof. The proof follows from the Hurwitz assumption (Assumption 1) and expressions for time-dependent state for the LQR setting (Oksendal, 2013; Zheng et al., 2021). □

Lemma 11 (Pointwise bound on the bootstrapped TD error)**.** *Let $\ell > 0$ then the bootstrapped TD error satisfies:*

$$|\bar{\delta}_t| \le C_\ell \Big[1 + \|\widehat{U}\|_F \sup_{u\in[t,t+\ell]} \|\tilde{s}_u^{\mathbf{W}}\|^2 + \|\widehat{U'}\|_F \sup_{u\in[t,t+\ell]} \|\tilde{s}_u^{\mathbf{W}}\| + |c| + \sup_{u\in[t,t+\ell]} \|a_u^{\mathbf{W}}\|^2 + \|\widehat{Z}\|_F \sup_{u\in[t,t+\ell]} \|\tilde{s}_u^{\mathbf{W}}\| \|a_u^{\mathbf{W}}\| + \|\widehat{Z'}\|_F \sup_{u\in[t,t+\ell]} \|a_u^{\mathbf{W}}\| \Big].$$

Proof. The downstairs value estimate satisfies (for some s)

$$\left|\bar{\varphi}(s;\widehat{U},\widehat{U'})\right| \le C_V \left(\|\widehat{U}\|_F \|s\|^2 + \|\widehat{U'}\|_F \|s\| + |c| \right),$$

the reward satisfies

$$|r(s,a)| \le C_r \left(1 + \|s\|^2 + \|a\|^2\right),$$

and the downstairs advantage-rate estimate satisfies

$$\left|\bar{\psi}(s,a;\widehat{Z'},\widehat{Z})\right| \le C_A \left(\|\widehat{Z}\|_F \|\tilde{s}\| \|a\| + \|\widehat{Z'}\|_F \|a\| \right).$$

Then the downstairs TD error

$$\begin{aligned} \bar{\delta}_t = {} & \bar{\varphi}(\tilde{s}_t^{\mathbf{W}};\widehat{U},\widehat{U'}) \\ & - \int_t^{t+\ell} e^{-\beta(u-t)} \left[r(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}) - \bar{\psi}(\tilde{s}_u^{\mathbf{W}}, a_u^{\mathbf{W}}; \widehat{Z'}, \widehat{Z}) \right] \mathrm{d}u \\ & - e^{-\beta\ell} \bar{\varphi}(\tilde{s}_t^{\mathbf{W}};\widehat{U},\widehat{U'}) \end{aligned}$$

satisfies the pointwise bound

$$\begin{aligned} |\bar{\delta}_t| \le {} & C \left(1 + e^{-\beta\ell}\right) \left(\|\widehat{U}\|_F \|\tilde{s}_t^{\mathbf{W}}\|^2 + \|\widehat{U'}\|_F \|\tilde{s}_t^{\mathbf{W}}\| + |c| \right) \\ & + C \int_t^{t+\ell} e^{-\beta(u-t)} \Big[1 + \|\tilde{s}_u^{\mathbf{W}}\|^2 + \|a_u^{\mathbf{W}}\|^2 \\ & \qquad + \|\widehat{Z}\|_F \|\tilde{s}_u^{\mathbf{W}}\| \|a_u^{\mathbf{W}}\| + \|\widehat{Z'}\|_F \|a_u^{\mathbf{W}}\| \Big] \mathrm{d}u. \end{aligned}$$

Taking the supremum yields the result from the Lemma.

□

Notice that we can combine the pointwise bound on the bootstrapped TD-error with the first moment bounds on the state and action random variables above to prove a bound the one-step gradients.

Lemma 12 (Fourth-moment bound for one stochastic gradient block)**.** *Let $K \subset \mathcal{X}$ be compact, and suppose that $\Theta \in K$. Suppose the discounted stability assumption holds, and suppose the initial state, process noise, and exploration noise have sufficiently high finite moments. Then there exists a constant $C_K < \infty$, depending on K but independent of η, such that*

$$\mathbb{E}\left[\left\| \widehat{\mathcal{G}}(\widehat{U}) \right\|_F^4 \right] \le C_K.$$

Proof sketch. By definition,

$$\widehat{\mathcal{G}}(\widehat{U}) = \int_0^\infty e^{-\beta t} \bar{\delta}_t \, \tilde{s}_t^{\mathbf{W}} (\tilde{s}_t^{\mathbf{W}})^\top \, \mathrm{d}t.$$

Hence

$$\left\| \widehat{\mathcal{G}}(\widehat{U}) \right\|_F^4 \le C \left(\int_0^\infty e^{-\beta t} |\bar{\delta}_t| \|\tilde{s}_t^{\mathbf{W}}\|^2 \, \mathrm{d}t \right)^4.$$

By Lemma 11, the TD error is bounded by a polynomial in the parameter norms and in the state-action process over the finite window $[t, t+\ell]$. Since $\Theta \in K$, all parameter norms are uniformly bounded. Thus

$$|\bar{\delta}_t| \|\tilde{s}_t^{\mathbf{W}}\|^2$$

is bounded by a polynomial in

$$\sup_{u\in[t,t+\ell]} \|\tilde{s}_u^{\mathbf{W}}\|, \qquad \sup_{u\in[t,t+\ell]} \|a_u^{\mathbf{W}}\|.$$

The discounted stability assumption gives exponentially discounted moment bounds for the state and action processes. Together with sufficiently high finite moments of the initial state and noise, this implies

$$\mathbb{E}\left[\left(\int_0^\infty e^{-\beta t}|\bar{\delta}_t|\|\tilde{s}_t^{\mathbf{W}}\|^2 \,\mathrm{d}t\right)^4\right] \le C_K.$$

Therefore

$$\mathbb{E}\left[\left\|\widehat{\mathcal{G}}(\widehat{U})\right\|_F^4\right] \le C_K.$$

□

The same argument applies to the other components of Θ, since their stochastic update directions are also exponentially discounted integrals of the TD error multiplied by polynomial functions of the state-action process and bounded parameter blocks. Hence the stacked stochastic update direction $\widehat{F}(\Theta)$ satisfies

$$\mathbb{E}\left[\|\widehat{F}(\Theta)\|_{\mathcal{X}}^4\right] \le C_K, \qquad \Theta \in K.$$

G.2 Limit Theorem for Manifold Muon Learning Dynamics

To derive the learning dynamics, that is, how the parameters and the objective change over gradient steps, we do so in parameter time τ. We present the following theorem describing the parameter time dynamics which shows that upstairs dynamics match the downstairs dynamics.

Theorem 4 (Formal Upstairs/Downstairs)**.** *For the control problem with high-dimensional observations with low-dimensional latent dynamics, with the learning rate η, and noise, ϵ, scaling as $O(1/\sqrt{d})$, the learning process of the parameters in the limit $d \to \infty$, in parameter time τ,*

$$\Theta_\tau = \left\{\widehat{U}_\tau, \widehat{U'}_\tau, \widehat{Z}_\tau, \widehat{Z'}_\tau, \widehat{W}_\tau, c_\tau\right\},$$

is governed by an ordinary differential equation:

$$\dot{\Theta}_\tau = F(\Theta_\tau), \text{ where}$$

$$F = \begin{bmatrix} \mathbb{E}\left[L\left(\widehat{\mathcal{G}}(\widehat{U}) - \widehat{U}\,\widehat{\mathcal{G}}(\widehat{U})^\top \widehat{U}\right)\right] \\ \mathbb{E}\left[(L-1)\left(\widehat{\mathcal{G}}(\widehat{U'}) - \widehat{U'}\left(\widehat{\mathcal{G}}(\widehat{U'})\right)^\top \widehat{U'}\right)\right] \\ \mathbb{E}\left[(L-1)\left(\widehat{\mathrm{G}}(\widehat{Z}) - \widehat{Z}\,\widehat{\mathrm{G}}(\widehat{Z})^\top \widehat{Z}\right)\right] \\ \mathbb{E}\left[(L-2)\left(\widehat{\mathrm{G}}(\widehat{Z'}) - \widehat{Z'}\,\widehat{\mathrm{G}}(\widehat{Z'})^\top \widehat{Z'}\right)\right] \\ \mathbb{E}\left[(L-1)\left(\widehat{\mathbb{G}}(\widehat{W}) - \widehat{W}\,\widehat{\mathbb{G}}(\widehat{W})^\top \widehat{W}\right)\right] \\ \mathbb{E}\left[\left(1-e^{-\beta\ell}\right)\int_0^\infty e^{-\beta t}\,\bar{\delta}_t \,\mathrm{d}t\right], \end{bmatrix} \tag{47}$$

for the initial condition $\Theta_0 = \left\{\widehat{U}_0, \widehat{U'}_0, \widehat{Z}_0, \widehat{Z'}_0, \widehat{W}_0, c_0\right\}$, which corresponds to the limit of the transforms:

$$\widehat{U} := M^\top \Pi_{l=1}^L U_l M \in \mathbb{R}^{d_s\times d_s}, \quad \widehat{U'} := \Pi_{l=1}^L U_l' M \in \mathbb{R}^{1\times d_s}, \quad \widehat{Z} := M^\top \Pi_{l=1}^L Z_l \in \mathbb{R}^{d_s\times d_a},$$
$$\widehat{Z'} := \Pi_{l=1}^L Z_l' \in \mathbb{R}^{1\times d_a}, \quad \widehat{W} := \Pi_{l=1}^L W_l\, M \in \mathbb{R}^{d_a\times d_s}. \tag{48}$$

of the parameters at initialization in the above formulation: $\{\mathbf{W}_0, \mathbf{U}_0, \mathbf{Z}_0, c_0\}$. Further, the objective evaluated at parameter time $\tau : J_\tau$, depends only on these parameters in the downstairs formulation in limit.

Proof. To prove this, we use a strategy similar to the proof of Theorem 2.3 by Ben Arous et al. (2022) where we show that the process converges weakly to a process with differential generator of the learning process, in parameter time τ, is same as that of the ODE above and therefore is the same. Their proof in turn follows the strategy used by Stroock & Varadhan (2006).

We first introduce the notation for the expected update direction functions (Equation equation 47). We denote by $\bar{\mathcal{G}}, \bar{\mathbb{G}}$, and $\bar{\mathrm{G}}$ the expectation of the functions $\widehat{\mathcal{G}}, \widehat{\mathbb{G}}$, and $\widehat{\mathrm{G}}$ under the stochasticity of the environment corresponding to the value estimate, policy and advantage estimate parameters. We denote by $\{\mathbf{W}_\tau, \mathbf{U}_\tau, \mathbf{Z}_\tau, c_\tau\}$ the parameters in the downstairs formulation. Similarly, $\Theta_0 = \left\{\widehat{U}_\tau, \widehat{U'}_\tau, \widehat{Z}_\tau, \widehat{Z'}_\tau, \widehat{W}_\tau, c_\tau\right\}$ are the corresponding parameters in the downstairs formulation obtained from the mapping described in Equation equation 8. For the functions above we will use the subscript τ to denote functions evaluated at parameter set Θ_τ. Similarly, denote by $J_\tau = J(\mathbf{W}_\tau) = J(\widehat{W}_\tau)$, the primary quantity we are interested in tracking the dynamics of.

Using this notation, we first define the interpolated process for a fixed learning rate η. Let

$$N_\tau := \left\lfloor \frac{\tau}{\eta} \right\rfloor, \qquad \tau_i := i\eta,$$

where $[\tau/\eta]$ is the greatest integer less than or equal to τ/η, like a floor function. For the discrete iterates, write

$$J_i := J(\mathbf{W}_i) = J(\widehat{W}_i).$$

We define the piecewise-linear interpolation of the objective by

$$J_\tau^\eta = J_{N_\tau} + \frac{\tau - \eta N_\tau}{\eta} \left(J_{N_\tau+1} - J_{N_\tau}\right).$$

Similarly, all the parameters in Θ_τ are defined using interpolation. For the discrete-time iterates, write

$$\Theta_i := \Theta_{\tau_i}.$$

We define the piecewise-linear interpolation by

$$\Theta_\tau^\eta = \Theta_{N_\tau} + \frac{\tau - \eta N_\tau}{\eta} \left(\Theta_{N_\tau+1} - \Theta_{N_\tau}\right).$$

For notational simplicity, we suppress the dependence on η and write Θ_τ in place of Θ_τ^η.

This process Θ_i, J_i is defined for the canonical filtration corresponding to the parameter time process: $\mathcal{F}_i$ and the corresponding filtration $\{\mathcal{F}_i\}_{i\geq 0}$. Note that this not the same as the augmented filtration for the continuous time stochastic control problem as described in Sections A.1 for processes defined in 3. Let $\mathcal{F}_i$ denote the sigma-algebra generated by the learning process up to iteration i. Define the one-step increment

$$\Delta\Theta_i := \Theta_{i+1} - \Theta_i.$$

We decompose this increment into its conditional mean and martingale part:

$$\Delta\bar{\Theta}_i := \mathbb{E}\left[\Delta_i\Theta \mid \mathcal{F}_i\right],$$
$$\Delta\widetilde{\Theta}_i := \Delta_i\Theta - \Delta\bar{\Theta}_i.$$

Then

$$\mathbb{E}\left[\Delta\widetilde{\Theta}_i \mid \mathcal{F}_i\right] = 0.$$

Equivalently, writing

$$\Delta\widetilde{\Theta}_i = \eta M_{i+1},$$

we have

$$\mathbb{E}[M_{i+1} \mid \mathcal{F}_i] = 0.$$

By the one-step update theorem for the parameters (Theorem 3), each increment can be written in the form

$$\Delta\Theta_i = \eta\widehat{F}_i + q_i^\eta,$$

where $\widehat{F}_i$ is the stacked stochastic update direction obtained fromthe stochastic quantities$\bar{\delta}_t$, $\widehat{\mathcal{G}}$,$\widehat{\mathrm{G}}$, and $\widehat{\mathbb{G}}$, andq_i^η collects the higher-order errors coming from the retraction expansion and the upstairs-downstairs approximation. In particular, uniformly on compact parameter-time intervals,

$$\|q_i^\eta\| \le C(\eta^2 + \eta\epsilon) + \eta o_d(1).$$

We define the limiting drift by

$$F(\Theta_i) = \mathbb{E}[\widehat{F}_i \mid \mathcal{F}_i] - b_i^\eta,$$

where b_i^η denotes the error that emerges from d being finite and satisfies

$$\sup_{i\eta \le T} \|b_i^\eta\| \longrightarrow 0 \qquad \text{as } d \to \infty.$$

Therefore,

$$\Delta\bar{\Theta}_i = \mathbb{E}[\Delta\Theta_i \mid \mathcal{F}_i] = \eta F(\Theta_i) + \eta r_i^\eta,$$

where

$$r_i^\eta := b_i^\eta + \frac{1}{\eta}\mathbb{E}[q_i^\eta \mid \mathcal{F}_i].$$

Thus the remainder satisfies

$$\sup_{i\eta \le T} \|r_i^\eta\| = O(\eta + \epsilon) \longrightarrow_{d\to\infty} 0, \tag{49}$$

since $\eta = O(d^{-1/2})$ and $\epsilon = O(d^{-1/2})$. Consequently, the one-step update may be written as

$$\Delta\Theta_{i+1} = \eta F(\Theta_i) + \eta M_{i+1} + \eta r_i^\eta,$$

with

$$\mathbb{E}[M_{i+1} \mid \mathcal{F}_i] = 0.$$

Using the interpolation, we may write Θ_τ as

$$\Theta_\tau = \Theta_0 + \int_0^\tau F(\Theta_s)\,\mathrm{d}s + \mathcal{M}_\tau^\eta + \mathcal{R}_\tau^\eta, \tag{50}$$

where

$$\mathcal{M}_\tau^\eta := \sum_{i=0}^{N_\tau - 1} \eta M_{i+1},$$
$$\mathcal{R}_\tau^\eta := \sum_{i=0}^{N_\tau - 1} \eta r_i^\eta + \left[\sum_{i=0}^{N_\tau - 1} \eta F(\Theta_i) - \int_0^\tau F(\Theta_s)\,\mathrm{d}s\right] + \rho_\tau^\eta.$$

Now to show that the limit $\eta \to 0$ converges to the ODE in the statement of the theorem we seek to apply the Kolmogorov continuity theorem in order to show that in the limit $\eta \to 0$ (as $d \to \infty$) this interpolated process converges to a process with sample-continuous trajectories. To apply the Kolmogorov continuity theorem we first need to show Holder continuity of the interpolated process for finite η and then infer that limit point is also Holder continuous due to compactness. This is the same proof strategy as used by Ben Arous et al. (2022), albeit a lot less general. Note that $\Theta \in \mathbb{R}^{d_s\times d_s} \times \mathbb{R}^{1\times d_s} \times \mathbb{R}^{d_s\times d_a} \times \mathbb{R}^{1\times d_a} \times \mathbb{R}^{d_a\times d_s} \times \mathbb{R}$ and the Kolmogorov continuity theorem (Stroock & Varadhan, 2006) requires an underlying metric and we use the Frobenius norm for the matrices and canonical squared difference for c, which gives a metric over the product space.

Let K be some compact set of the aforementioned product space and τ_K be the exit time from the set K of the interpolated process and we obtain (from results in Section G.1) and moment bounds on the martingale component we have:

$$\mathbb{E}\left[\|\Theta_{\tau_K\wedge\tau} - \Theta_{\tau_K\wedge\tau'}\|_{\mathcal{X}}^4\right] \le C_{K,T}|\tau - \tau'|^2, \qquad \forall \tau, \tau' \in [0, T].$$

Therefore, Θ_τ^K has sample paths that are a.s. α- Holder continuous for every $\alpha < 1/4$. Therefore, the sequence with $d \to \infty$ is tight with $1/4$-Holder limit points.

Now to show that the limit point indeed converges to the solution of the ODE in Equation 47 we consider the limit $\eta \to 0$. In this setting the jumps in the martingale process are $\Delta||\mathcal{M}_\tau^\eta||^2 = O(\eta^2)$ and therefore go to zero, meaning the process cnverges to the mean gradient flow dynamics. We show this by considering some function $f \in C^2$:

$$
\begin{aligned}
&f\left(\Theta_{(\tau+\eta)\wedge\tau_K}\right) - f\left(\Theta_{\tau\wedge\tau_K}\right) \\
&= \mathbf{1}_{\{\tau<\tau_K\}} \left\langle \nabla f(\Theta_\tau), \Theta_{\tau+\eta} - \Theta_\tau \right\rangle_{\mathcal{X}} + \frac{1}{2}\mathbf{1}_{\{\tau<\tau_K\}} \left\langle \Theta_{\tau+\eta} - \Theta_\tau, \nabla^2 f(\widetilde{\Theta}_\tau)\left(\Theta_{\tau+\eta} - \Theta_\tau\right) \right\rangle_{\mathcal{X}},
\end{aligned}
$$

now substituting the one-step expansion of Θ we obtain:

$$
\begin{aligned}
&f\left(\Theta_{(\tau+\eta)\wedge\tau_K}\right) - f\left(\Theta_{\tau\wedge\tau_K}\right) \\
&= \eta\mathbf{1}_{\{\tau<\tau_K\}} \left\langle \nabla f(\Theta_\tau), F(\Theta_\tau) \right\rangle_{\mathcal{X}} + \eta\mathbf{1}_{\{\tau<\tau_K\}} \left\langle \nabla f(\Theta_\tau), M_{\tau+\eta} \right\rangle_{\mathcal{X}} \\
&\quad + \eta\mathbf{1}_{\{\tau<\tau_K\}} \left\langle \nabla f(\Theta_\tau), r_\tau^\eta \right\rangle_{\mathcal{X}} + O\left(\|\Theta_{\tau+\eta} - \Theta_\tau\|_{\mathcal{X}}^2\right).
\end{aligned}
$$

Taking conditional expectation with respect to the appropriate filtration we get:

$$
\mathbb{E}\left[M_{\tau+\eta} \mid \mathcal{F}_\tau\right] = 0,
$$

from the Martingality. Therefore,

$$
\begin{aligned}
\frac{1}{\eta}\mathbb{E}\left[f\left(\Theta_{(\tau+\eta)\wedge\tau_K}\right) - f\left(\Theta_{\tau\wedge\tau_K}\right) \mid \mathcal{F}_\tau\right] &= \mathbf{1}_{\{\tau<\tau_K\}} \left\langle \nabla f(\Theta_\tau), F(\Theta_\tau) \right\rangle_{\mathcal{X}} \\
&\quad + \mathbf{1}_{\{\tau<\tau_K\}} \left\langle \nabla f(\Theta_\tau), r_\tau^\eta \right\rangle_{\mathcal{X}} + O(\eta).
\end{aligned}
$$

Since $r_\tau^\eta \to 0$ as $d \to \infty$ (Equation equation 49) we have the limiting differential operator:

$$
\mathcal{L}f(\Theta) = \left\langle \nabla f(\Theta), F(\Theta) \right\rangle_{\mathcal{X}}.
$$

This is also from the fact that the square of the martingale term in $\mathbb{E}[(\delta f)^2] - \mathbb{E}[(\delta f)^2] \to 0$ at rate η^2, meaning the stochastic term vanishes conditioned on the appropriate filtration.

We have therefore shown that the stopped process follows the ODE in equation 47. Finally, the initial condition follows from the convergence of the transforms in Equation 48 as $d \to \infty$ due to the compactness of the Stiefel manifold.

□

G.3 Proof Sketch

The proof proceeds by firsst deriving the one step update for each of the matrix in the downstairs formulation. Then we derive the one step update of the objective with residual terms (Theorem 3). We then bound various quantities to show that as learning rate goes to zero these quantities are "well behaved". Using this bound and the proof technique form Ben Arous et al. (2022) we show that the parameter time process converges to an ODE that remains on the product Stiefel manifold of the downstairs formulation. We show how these ideas are connected in figure 8.

H Continuous time Deterministic Policy Gradient with White Noise Exploration

Here we describe the precise algorithm we use for Stiefl manifold based learning, using the Cayley transform. First we re-iterate the Cayley transform:

$$
\mathrm{StiefelStep}(X, D, \eta) := \left(I_d + \frac{\eta}{2}\Omega_X\right)^{-1}\left(I_d - \frac{\eta}{2}\Omega_X\right)X, \qquad \Omega_X := DX^\top - XD^\top.
$$

Using this notation we present Algorithm 1 below for a sufficiently large episode horizon K.

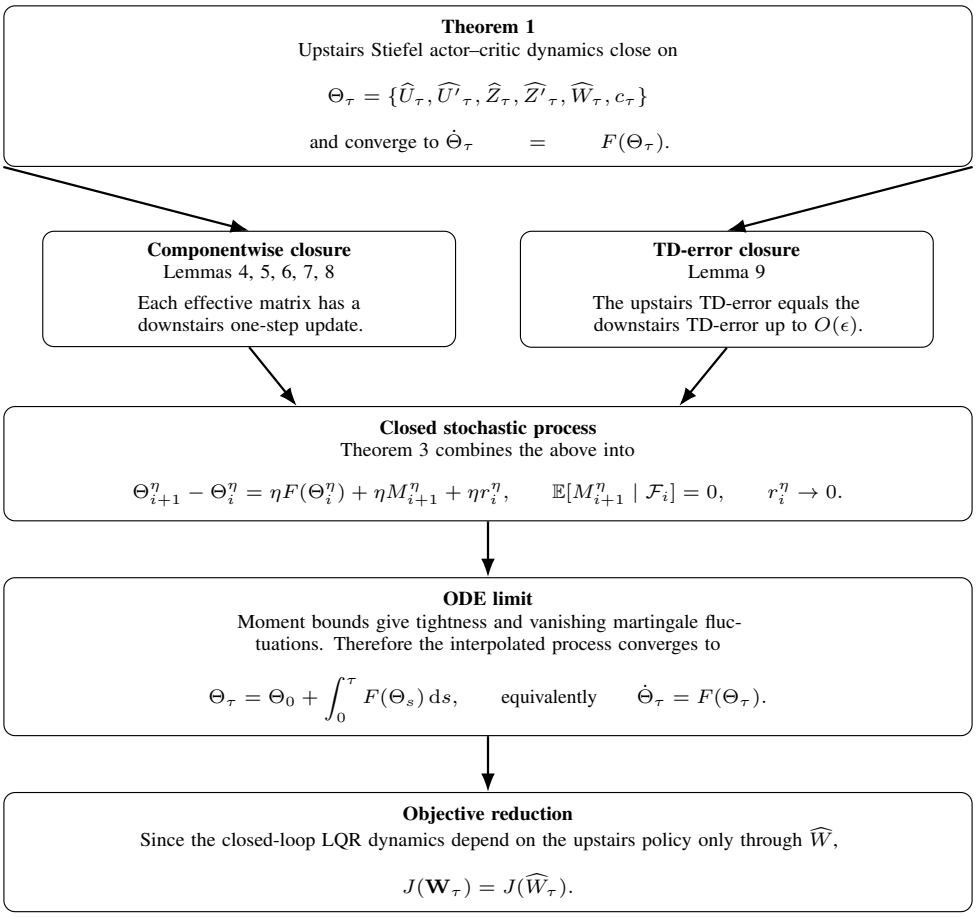


Figure 8: Proof sketch of Theorem 1.

I Implementation Details

I.1 PixelBrax

We simulate environments using the pixelbrax framework (McInroe & Garcin, 2025) which is a wrapper around the classical MuJoCo environments (Todorov et al., 2012). The pixel inputs are rendeded into $84x84$ pixel frames and stacked into 4-frame blocks and normalized. All of our experiments were run on Nvidia H100s, and the max wall clock time per experiment was 8 hours.

I.2 Basic CNN Encoder

Encoder Details Our basic encoder uses three convolutional layers which with layer normalization and ReLU activations shown in Figure 9. The features are flattened and projection to a shared latent representation. The actor and the critic are distinct two-layer MLP heads (256 units per layer, Swish activations (Ramachandran et al., 2017)); the actor produces Gaussian means with a learned log-standard-deviation vector, and the critic produces a scalar value estimate. We apply a scaled hyperbolic tangent bottleneck, $h = \tanh(\alpha z)$ with $\alpha = 0.5$, to control the magnitude of the representation and reduce saturation-induced instability. The full set of hyperparameters can be found in Table 5.

Let $o_t \in \{0, \ldots, 255\}^{84\times84\times3}$ denote the raw pixel observation at time t. With frame stacking $F = 4$, the encoder input is

$$x_t = \mathrm{concat}(o_{t-3}, o_{t-2}, o_{t-1}, o_t) \in \mathbb{R}^{84\times84\times(4\times3)},$$

followed by per-pixel normalization $x_t \leftarrow x_t/255$.

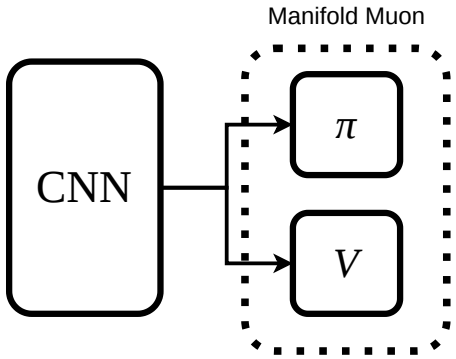


Figure 9: A diagram of the CNN with Actor and Critic Heads

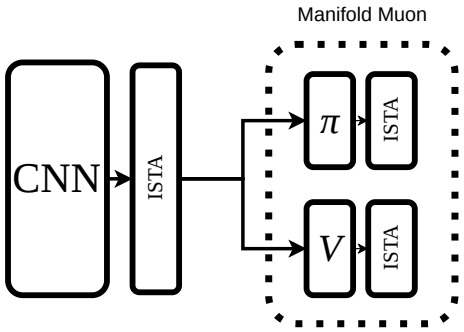


Figure 10: Diagram of the ISTA encoder with ISTA actor and critic heads

The CNN encoder is

$$
\begin{aligned}
h_1 &= \mathrm{ReLU}(\mathrm{LN}(\mathrm{Conv}(x_t;\, 32,\, 8{\times}8,\, s{=}4,\,)))\,, \\
h_2 &= \mathrm{ReLU}(\mathrm{LN}(\mathrm{Conv}(h_1;\, 64,\, 4{\times}4,\, s{=}2)))\,, \\
h_3 &= \mathrm{ReLU}(\mathrm{LN}(\mathrm{Conv}(h_2;\, 64,\, 3{\times}3,\, s{=}1)))\,, \\
z &= \mathrm{LN}(\mathrm{Dense}_{512}(\mathrm{Flatten}(h_3)))\,, \\
\phi_t &= \tanh(0.5\, z).
\end{aligned}
$$

For $84{\times}84$ inputs, spatial sizes are $20 \times 20 \to 9 \times 9 \to 7 \times 7$, so $\mathrm{Flatten}(h_3) \in \mathbb{R}^{3136}$, and $\phi_t \in [-1, 1]^{512}$.

PPO Details Actor and critic heads are separate MLPs fed by ϕ_t:

$$\text{Actor trunk: } \phi_t \to 256 \to 256 \;\; (\mathrm{Swish}), \qquad \text{Critic trunk: } \phi_t \to 256 \to 256 \;\; (\mathrm{Swish}).$$

The actor outputs a diagonal-Gaussian policy with mean $\mu_t = \mathrm{Dense}_{d_a}(\cdot)$ and a learned state-independent $\log \sigma \in \mathbb{R}^{d_a}$ parameter; actions are sampled from $\mathcal{N}(\mu_t, \mathrm{diag}(\exp(\log \sigma)^2))$ and clipped to $[-1, 1]$. The critic outputs $V_t = \mathrm{Dense}_1(\cdot)$.

All dense/conv kernels use orthogonal initialization.

I.3 ISTA Network

ISTA-CNN Encoder Our ISTA-CNN encoder replaces the standard tanh-bottleneck CNN representation with a learned ISTA-style bottleneck shown in Figure 10. The encoder uses two convolutional layers with layer normalization and ReLU activations, followed by a 512-dimensional projection and

Table 5: Hyperparameters

Category	Hyperparameter	Value
Environment/Data	Observation resolution	84×84 pixels
Environment/Data	Frame stack	4
Environment/Data	Action repeat	4
Environment/Data	Number of parallel environments	128
Environment/Data	Environments (swept)	{halfcheetah, walker2d, ant, humanoid, reacher, swimmer, pusher, hopper, inverted_pendulum}
Environment/Data	Physics backend (by env)	{spring, generalized, positional} mapping in script
Environment/Data	Seeds (swept)	$\{0, 1, 2, 3, 4, 5\}$
Architecture	Encoder type	CNN
Architecture	Conv stack	$(32, 8\times8, s4) \rightarrow (64, 4\times4, s2) \rightarrow (64, 3\times3, s1)$, VALID
Architecture	Encoder projection dim	512
Architecture	Encoder output nonlinearity	$\tanh(\alpha z),\ \alpha = 0.5$
Architecture	Actor head	MLP $256 \rightarrow 256$ (Swish), Gaussian mean + learned log-std
Architecture	Critic head	MLP $256 \rightarrow 256$ (Swish), scalar value output
PPO	Total timesteps	10^7 (script default)
PPO	Rollout horizon (`num_steps`)	10
PPO	Batch size ($n_{\text{env}} \times$steps)	1280
PPO	Mini-batches	32
PPO	Mini-batch size	40
PPO	Update epochs per rollout	4
PPO	Discount factor γ	0.99
PPO	GAE λ	0.95
PPO	PPO clip ϵ	0.1
PPO	Entropy coefficient	0.0
PPO	Value loss coefficient	0.5
PPO	Advantage normalization	True
PPO	Value clipping (`clip_vloss`)	True
Optimization	LR annealing	Enabled
Optimization	Encoder optimizer	Adam, lr $= 3 \times 10^{-4}$, $\epsilon = 10^{-5}$, grad clip $= 0.05$
Optimization	Head optimizer condition (swept)	Adam-only vs mixed Muon/Adam
Optimization	Head Adam lr	3×10^{-4}
Optimization	Head Muon lr (matrix params)	10^{-3}
Optimization	Muon dual lr	10^{-2}
Optimization	Muon dual steps	5
Optimization	Muon matrix-sign iterations	5
Optimization	Actor Muon grad clip	100
Optimization	Critic Muon grad clip	1
Optimization	Weight decay	0 (Adam, not AdamW)
Optimization	Muon partition rule	Matrix-valued actor/critic params use Muon; vector/scalar params use Adam

a single ISTA feed-forward block. The actor and critic are distinct ISTA heads: each applies a 256-dimensional Swish layer followed by a ISTA feed-forward block. The actor produces tanh-bounded Gaussian means and a learned bounded state-independent log-standard-deviation vector, while the critic produces a scalar value estimate. The full set of hyperparameters can be found in Table 6.

Let $o_t \in \{0, \ldots, 255\}^{84\times84\times3}$ denote the raw pixel observation at time t. With frame stacking $F = 4$, the encoder input is

$$x_t = \text{concat}(o_{t-3}, o_{t-2}, o_{t-1}, o_t) \in \mathbb{R}^{84\times84\times(4\times3)},$$

followed by per-pixel normalization $x_t \leftarrow x_t/255$.The ISTA feed-forward block is a single learned ISTA-like update,

$$\text{ISTA}_\eta(\alpha) = \text{Swish}\big(Z - \alpha W^\top (ZW - Z)\big),$$

where W is a learned square matrix and $\alpha = 0.1$ is the step size.

The ISTA-CNN encoder is

$$\begin{aligned} h_1 &= \text{ReLU}(\text{LN}(\text{Conv}(x_t;\ 64,\ 4{\times}4,\ s{=}2))), \\ h_2 &= \text{ReLU}(\text{LN}(\text{Conv}(h_1;\ 64,\ 3{\times}3,\ s{=}1))), \\ z &= \text{LN}(\text{Dense}_{512}(\text{Flatten}(h_2))), \\ \phi_t &= \text{ISTA}_{0.1}(z). \end{aligned}$$

ISTA Actor and Critic Heads Actor and critic heads are separate ISTA heads fed by ϕ_t:

Actor trunk: $\phi_t \to 256 \to \text{ISTA}_{0.1}$ (Swish), Critic trunk: $\phi_t \to 256 \to \text{ISTA}_{0.1}$ (Swish).

The actor outputs a diagonal-Gaussian policy with tanh-bounded mean

$$\mu_t = \tanh(\text{Dense}_{d_a}(\cdot)),$$

using scale 1.0. The log-standard deviation is a learned state-independent vector $\log\sigma \in \mathbb{R}^{d_a}$, initialized to -2.0 and sigmoid-parameterized to remain in

$$\log\sigma \in [-5.0, -1.4].$$

Actions are sampled from

$$\mathcal{N}(\mu_t, \text{diag}(\exp(\log\sigma)^2))$$

and clipped to $[-1, 1]$. The critic outputs $V_t = \text{Dense}_1(\cdot)$. All dense/conv kernels use orthogonal initialization except the learned ISTA matrices, which use Kaiming-uniform initialization.

J Additional Experiments

We provide additional experiments with more environments in Figure 11. Our findings are consistent in that the Stiefel update generally matches or improves on Adam for the CNN, but the improved ISTA-Encoder helps both optimizers. We also include the addition of vanilla Muon Jordan et al. (2024) to the actor and critic heads. We use identical global hyperparameter, and default Optax DeepMind et al. (2020) Muon-specific hyperparameters. Vanilla Muon orthonoromalizes the gradient update, but does not restrict the weights to lie on the Stiefel manifold. We find that this change actively prevents learning in almost all environments indicating that the Stiefel retraction of the actor/critic weight matrices is the key intervention to improving performance. We were able to tune a learning rate so that vanilla Muon could perform well on individual environments, but unable to find a learning rate that performed well across all environments.

Table 6: Hyperparameters for ISTA-CNN with ISTA heads

Category	Hyperparameter	Value
Environment/Data	Observation resolution	84×84 pixels
Environment/Data	Frame stack	4
Environment/Data	Action repeat	4
Environment/Data	Number of parallel environments	128
Environment/Data	Environments (swept)	{halfcheetah, walker2d, ant, humanoid, reacher, swimmer, pusher, hopper, inverted_pendulum}
Environment/Data	Physics backend (by env)	{spring, generalized, positional} mapping in script
Environment/Data	Seeds (swept)	$\{0, 1, 2, 3\}$
Architecture	Encoder type	ISTA-CNN
Architecture	Conv stack	$(64, 4\times4, s2) \rightarrow (64, 3\times3, s1)$, VALID
Architecture	Encoder projection dim	512
Architecture	Encoder bottleneck	$\text{ISTA}_{0.1}$ block
Architecture	Actor head	Dense 256 + $\text{ISTA}_{0.1}$, tanh-bounded Gaussian mean + bounded learned log-std
Architecture	Critic head	Dense 256 + $\text{ISTA}_{0.1}$, scalar value output
Architecture	Actor mean bound	$\mu = \tanh(\cdot)$, scale 1.0
Architecture	Actor log-std bounds	initialized -2.0, constrained to $[-5.0, -1.4]$
PPO	Total timesteps	10^7 (script default)
PPO	Rollout horizon (`num_steps`)	10
PPO	Batch size ($n_{\text{env}} \times$steps)	1280
PPO	Mini-batches	32
PPO	Mini-batch size	40
PPO	Update epochs per rollout	4
PPO	Discount factor γ	0.99
PPO	Entropy coefficient	0.0
PPO	Value loss coefficient	0.5
PPO	Advantage normalization	True
PPO	Value clipping (`clip_vloss`)	True
Optimization	LR annealing	Enabled
Optimization	Encoder optimizer	Adam, lr $= 3 \times 10^{-4}$, grad clip $= 0.05$
Optimization	Head optimizer	Mixed Stiefel/Adam
Optimization	Head Adam lr	3×10^{-4}
Optimization	Head Stiefel lr (matrix params)	10^{-3}
Optimization	Stiefel dual lr	10^{-2}
Optimization	Stiefel dual steps	5
Optimization	Actor Stiefel grad clip	100
Optimization	Critic Stiefel grad clip	1
Optimization	Weight decay	0 (Adam, not AdamW)
Optimization	Stiefel partition rule	Matrix-valued actor/critic params use Stiefel; vector/scalar params use Adam

Algorithm 1 Episodic Online Actor–Critic

Require: discretization step h, episode horizon K, number of episodes N_{ep}, discount rate β, bootstrap length M, learning rates η_π, η_v, η_A, exploration scale σ, level of observation noise ϵ.
Require: policy $\phi(\cdot;\mathbf{W})$, value estimate $\varphi(\cdot;\mathbf{U})$, advantage-rate estimate $\psi(\cdot,\cdot;\mathbf{Z})$

Initialize all square trainable matrices on the Stiefel manifold.
Initialize the scalar critic parameter c.
for $n = 1, \dots, N_{\mathrm{ep}}$ **do**
  Initialize $s_0^{(n)}$.
  for $k = 0, \dots, K-1$ **do**
    Set $t_k \leftarrow kh$.
    Roll out the exploratory continuous-time policy on $[t_k, t_{k+1}]$:

$$\pi(o_t)\mathrm{d}t = \phi(o_t;\mathbf{W})\mathrm{d}t + \sigma \mathrm{d}w'_t.$$

    Equivalently, sample the state increment from the observation-free SDE in Equation equation 12.
    Receive reward $r_k = r(\tilde{s}_k, a_k)$ and observe o_{k+1}.
  end for
  Initialize episode gradient directions

$$D_X^v \leftarrow 0, \quad D_Y^A \leftarrow 0, \quad D_{W_l}^\pi \leftarrow 0$$

  for every trainable critic matrix $X \in \mathbf{U}$, advantage-rate matrix $Y \in \mathbf{Z}$, and policy matrix $W_l \in \mathbf{W}$.
  Initialize scalar critic direction $D_c^v \leftarrow 0$.
  for $j = 0, \dots, K-M$ **do**
    Compute the M-step continuous-time TD error

$$\widehat{\delta}_j = \varphi(o_j;\mathbf{U}) - \sum_{q=0}^{M-1} h e^{-\beta q h}\left[r_{j+q} - \psi(o_{j+q}, a_{j+q};\mathbf{Z})\right] - e^{-\beta M h}\varphi(o_{j+M};\mathbf{U}).$$

    Accumulate critic loss-gradient directions, for every trainable critic matrix $X \in \mathbf{U}$:

$$D_X^v \leftarrow D_X^v + e^{-\beta t_j}\partial_X \varphi(o_j;\mathbf{U})\widehat{\delta}_j.$$

    Accumulate scalar critic loss-gradient direction:

$$D_c^v \leftarrow D_c^v + e^{-\beta t_j}\partial_c \varphi(o_j;\mathbf{U})\widehat{\delta}_j.$$

    Accumulate advantage-rate loss-gradient directions, for every trainable matrix $Y \in \mathbf{Z}$:

$$D_Y^A \leftarrow D_Y^A + e^{-\beta t_j}\partial_Y \psi(o_j, a_j;\mathbf{Z})\widehat{\delta}_j.$$

    Accumulate policy-gradient ascent directions, for every trainable policy matrix W_l:

$$G_{W_l}^\pi \leftarrow G_{W_l}^\pi + e^{-\beta t_j}\partial_{W_l}\phi(o_j;\mathbf{W})\partial_a \psi(o_j, a;\mathbf{Z})\Big|_{a=\phi(o_j;\mathbf{W})}.$$

  end for
  Update every square critic matrix $X \in \mathbf{U}$ by descent:

$$X \leftarrow \mathrm{StiefelStep}(X, D_X^v, \eta_v).$$

  Update the scalar critic parameter by descent:

$$c \leftarrow c - \eta_v D_c^v.$$

  Update every square advantage-rate matrix $Y \in \mathbf{Z}$ by descent:

$$Y \leftarrow \mathrm{StiefelStep}(Y, D_Y^A, \eta_A).$$

  Update every trainable policy matrix $W_l \in \mathbf{W}$ by ascent:

$$W_l \leftarrow \mathrm{StiefelStep}(W_l, -G_{W_l}^\pi, \eta_\pi).$$

end for

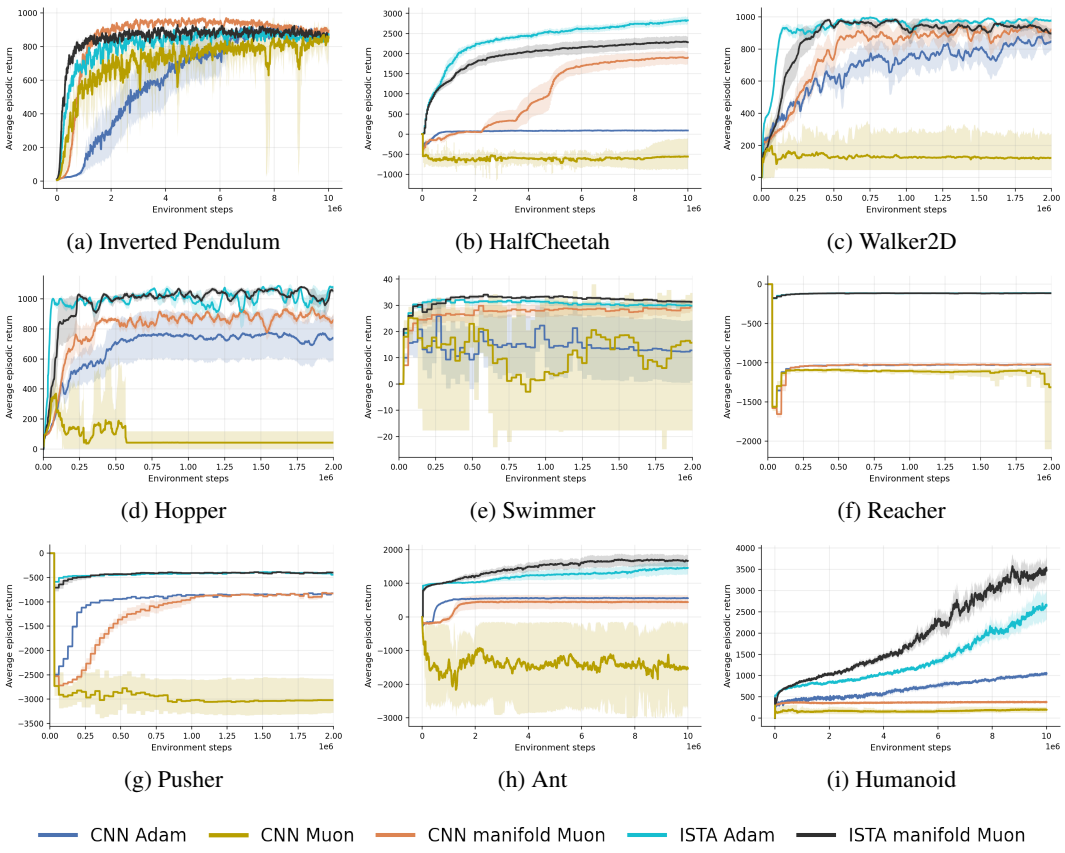


Figure 11: Additional RL control experiments with Muon baselines. Curves show mean performance with standard-error bands over seeds where available.

K CASE STUDY: INVERTED PENDULUM

We investigate the specific inverted pendulum environment from Brax (Freeman et al., 2021), to show how our method is able to extract the low-dimensional features from the high-dimensional upstairs pixel observations. This environment is a continuous-action task in which the agent applies a horizontal force to a cart in order to balance a pole upright (also known as cart-pole). Recall the Markov decision process (MDP) (Bellman, 1966), defined by the tuple $(\mathcal{S}, \mathcal{A}, r, P, s_0, \gamma, T)$ where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions available to the agent, P is the transition kernel with $P(s, a, s')$ being the probability of the agent transitioning from state s to s' after taking action a, r is the reward, s_0 is the initial state, γ is discounting, and T is the time horizon. In inverted pendulum, $s \in \mathcal{S} \subset \mathbb{R}^4$

$$s_{t_k} := \begin{bmatrix} x_{\text{cart}} \\ \theta \\ \dot{x}_{\text{cart}} \\ \dot{\theta} \end{bmatrix} \in \mathbb{R}^{d_s}, \qquad d_s = 4,$$

where θ is the pole angle measured from the upright configuration (so $\theta = 0$ is perfectly upright in the task description), x_{cart} is the x-position, $\dot{x}\text{cart}$ is the velocity and $\dot{\theta}$ is the angular velocity of the pole. Since the simulator runs in discrete time, we introduce the time instants $t_k := k\Delta t$ as subscript to the state (i.e., s_{t_k} describes the state at time k where the intervals of Δt). The action dimension, $a \in \mathcal{A} \subset \mathbb{R}$ so $d_a = 1$. The environment receives a normalized control $u_k \in [-1, 1]$ which is generated from the policy network $\bar{\phi}(s^{\pi}_{t_k}; \mathbf{W})$. The action is linearly scaled to the actuator force range:

$$a_k \;=\; \frac{u_k + 1}{2}\,(a_{\max} - a_{\min}) + a_{\min} \;\in\; [a_{\min}, a_{\max}].$$

To align with our formulation, we treat u_k as the policy output, and interpret a_k as the physical force applied by the environment. The episode terminates when the pole angle leaves a fixed neighborhood of the upright configuration:

$$\texttt{done}_k = 1 \quad \Longleftrightarrow \quad |\theta_{t_k}| > \theta_{\max}, \qquad \theta_{\max} = 0.2 \text{ rad}.$$

Let $\tau := \min(1000, \inf\{t_k : |\theta_{t_k}| > \theta_{\max}\}$ denote the stopping time. If $\tau = 1000$ and $|\theta_{t_k}| \leq \theta_{\max}\}$, the task is considered solved. In the Brax implementation, the agent receives a constant reward of $+1$ at each non-terminal step. To match our infinite-horizon discounted objective (Equation equation 4) in discrete time, we can write the episodic return as

$$J(\mathbf{W}) \;:=\; \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k\, \mathbf{1}\{t_k < \tau\}\right], \qquad \gamma \in (0, 1), \tag{51}$$

which corresponds to the continuous time analog $J(\mathbf{W}) = \mathbb{E}[\int_0^{\tau} e^{-\beta t}\,\mathrm{d}t]$ when $\gamma = e^{-\beta \Delta t}$. The discrete-time simulator dynamics can be written as

$$s_{t_{k+1}} = P_{\Delta t}(s_{t_k}, a_{t_k}),$$

where Δt is the simulator step size and P is the transition kernel (Tedrake, 2009, Chapter 3). Let $(s^\star, a^\star) = (0, 0)$ denote the upright equilibrium and define perturbations

$$\delta s_{t_k} := s_{t_k} - s^\star \in \mathbb{R}^4, \qquad \delta a_{t_k} := a_{t_k} - a^\star \in \mathbb{R}.$$

The linearized discrete-time equations of motion around the equilibrium is (Tedrake, 2009):

$$\delta s_{t_{k+1}} \approx G_{\Delta t}\,\delta s_{t_k} + H_{\Delta t}\,\delta a_{t_k},$$

where

$$G_{\Delta t} := \left.\frac{\partial P_{\Delta t}}{\partial s}\right|_{(s^\star, a^\star)} \in \mathbb{R}^{4\times 4}, \qquad H_{\Delta t} := \left.\frac{\partial P_{\Delta t}}{\partial a}\right|_{(s^\star, a^\star)} \in \mathbb{R}^{4\times 1}.$$

Taking $\Delta t \to 0$ yields the local continuous-time model

$$\mathrm{d}s_t = (Gs_t + Ha_t)\,\mathrm{d}t + \Sigma\,\mathrm{d}w_t,$$

which is precisely the form of our continuous time dynamics in Equation equation 12. Thus, the discrete Brax simulator can be viewed locally as a time-discretization of the continuous-time linear system used in our theory. To define the associated LQR, choose quadratic state and control weights

$$Q \in \mathbb{R}^{4\times 4}, \qquad R \in \mathbb{R}^{1\times 1}, \qquad Q = Q^\top \succeq 0, \qquad R = R^\top \succ 0.$$

The standard continuous-time LQR cost is then

$$\int_0^\infty e^{-\beta t}\big(s_t^\top Q s_t + a_t^\top R a_t\big)\,\mathrm{d}t.$$

Therefore, the Brax inverted pendulum is locally an LQR-stabilizable system around the upright equilibrium, even though the full environment is nonlinear, and the specific cost/reward structure is different.

K.1 Pixel Observations

Unlike the state-observation version of the task, PixelBrax (McInroe & Garcin, 2025) does not expose s_{t_k} directly, meaning that is can be described as a POMDP. Recall that a POMDP is defined as the tuple $(\mathcal{S}, \mathcal{A}, r, P, s_0, \gamma, T, \mathcal{O}, O)$.

In PixelBrax, each simulator state is rendered to an image by the observation function:

$$O(s_{t_k}) \in \mathcal{O} \subset \mathbb{R}^{84\times 84\times C},$$

where C is the number of channels per frame. Recall that we assume that the mapping from state to observation is linear:

$$o_t \mathrm{d}t := \sum_{i=1}^{d_s} \mu_i s_t^i \mathrm{d}t + \epsilon \mathrm{d}b_t.$$

In particular, $\mu_i \in \mathbb{R}^d$ is the mean vector that is representative of the observation corresponding to the state dimension i. Since, in practice, the agent then receives a stack of the F most recent rendered frames we define the augmented observation space, $\tilde{\mathcal{O}}$, with an augmented observation:

$$\tilde{o}_{t_k} \;:=\; \mathrm{Stack}\Big(\mathcal{O}(s_{t_{k-F+1}}), \mathcal{O}(s_{t_{k-F+2}}), \ldots, \mathcal{O}(s_{t_k})\Big) \in \tilde{\mathcal{O}} \subset \mathbb{R}^{84\times 84\times (CF)}.$$

In our experiments, $F = 4$, so the augmented observation consists of four stacked $84 \times 84 \times C$ frames. When flattened, the upstairs observation dimension becomes

$$d \;=\; 84 \cdot 84 \cdot C \cdot F.$$

K.2 Linear Probe of the Downstairs State

Since the agent needs to learn from pixel observations, we endow the agent with a CNN encoder which learns mappings from the augmented observations to a latent space: $\psi : \tilde{\mathcal{O}} \to \mathbb{R}^{d_z}$. In order to interpret whether the encoder is learning a useful feature representation, we train a linear probe to fit the frozen CNN latents back to the downstairs state. To do so, we freeze the trained encoder ψ and extract the latents from the final linear layer:

$$z_{t_k} \;:=\; \psi(\tilde{o}_{t_k}) \in \mathbb{R}^{d_z}.$$

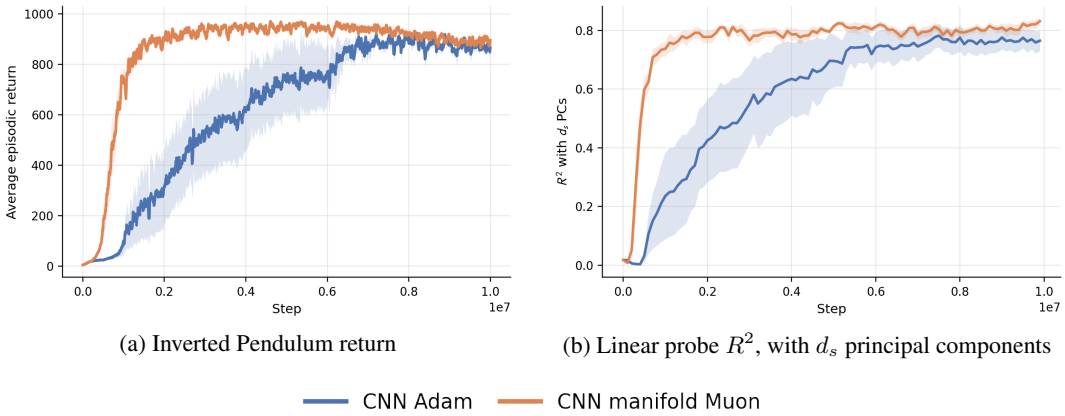
These latents are then paired with the current downstairs simulator state, $\tilde{s}^\pi_{t_k}$ so that the linear probe learns the mapping

$$\psi(\tilde{o}_{t_k}) \quad \mapsto \quad s_{t_k}.$$

In particular, we build a dataset of feature, state pairs by sampling N policy rollouts from the frozen encoder and policy, π, to generate the dataset $(z^\pi_{t_k,n}, \tilde{s}^\pi_{t_k,n})_{n=0}^N$. We then fit a ridge-regression probe:

$$\tilde{s}^\pi_{t_k} = W z^\pi_{t_k} + b, \qquad (W, b) = \arg\min_{W,b} \sum_{k,n} \|W z^\pi_{t_k,n} + b - \tilde{s}^\pi_{t_k,n}\|_2^2 + \alpha \|W\|_F^2,$$

To verify that the frozen pixel encoder has learned representations that linearly map to the downstairs state. That is, can we recover $z_t \approx \chi(\sum_{i=1}^{d_s} \mu_i s_t^i)$, where χ is a learnable map $\mathbb{R}^{d_z} \to \mathbb{R}^{d_s}$.



(a) Inverted Pendulum return

(b) Linear probe R^2, with d_s principal components

Figure 12: The learning curve and R^2 score of Inverted Pendulum.

K.2.1 Results

We plot the the episodic returns and the R^2 of the linear fit with $d_s = 4$ principal components in Figure 12. Manifold Muon learns a representation that can be linearly fit to the downstairs state earlier than Adam. Manifold Muon also achieves a much stronger fit and better episodic returns. Adam's R^2 score improves well before the returns improve, indicating that goodness-of-fit of the representations is indeed a leading indicator of performance. These results provide evidence that the encoder is organizing the upstairs observation stream into a representation that is geometrically matched to the downstairs state. Taken together, these results support the claim that successful learning from upstairs pixel observations is proceeded by constructing an internal, low-dimensional representation that fits the downstairs state.

K.3 A frozen orthonormal image encoder

To ask what a *structurally constrained, frozen* image representation costs relative to learning an encoder end-to-end, we train an encoder in isolation on `inverted_pendulum` and then freeze it and learn only the policy and value heads on top. The environment uses the generalized backend with 84×84 pixel observations and an action repeat of 4. All encoder work is carried out in a separate trainer so that the shared PPO implementation is untouched.

Supervision data. The encoder is trained by regression onto the environment's own proprioceptive state, so it needs observations drawn from policies of varying competence rather than from a single fixed policy. We collect rollouts from 3 seeds $\times$ 50 checkpoints of an earlier pixel-PPO run on the same environment, 100 episodes per checkpoint, recording the rendered frames alongside the 4-dimensional state $z = (\text{cart position}, \text{pole angle}, \text{cart velocity}, \text{pole angular velocity})$. Every fifth checkpoint is held out, so the validation set consists of *policies the encoder was never trained on* rather than held-out frames from seen policies; this measures whether the representation generalises across behaviour, which is what the downstream use requires.

Architecture and the orthonormality constraint. The encoder is

$$\phi(I) \;=\; M\,h(I), \qquad M \in \mathbb{R}^{4\times 256}, \quad MM^\top = I_4, \tag{52}$$

where h is a convolutional trunk followed by a non-linear feed-forward layer of width 256 with swish activations, and M has orthonormal rows. The constraint is maintained by a QR retraction applied after every optimizer step, which holds $\|MM^\top - I\|$ at $\approx 3 \times 10^{-7}$ throughout training. Note that pixel normalisation happens *inside* the network: the trunk divides by 255, so raw 0–255 frames are passed in, matching the convention of the PPO encoders the representation is compared against.

Objective. ϕ is regressed directly onto the *standardised raw state* z, not onto a principal-component projection of it. This matters: the two velocity coordinates are correlated at ≈ -0.99 on-policy,

so the state has a participation-ratio rank of only 2.15 despite being 4-dimensional, and an earlier 2-D PCA target discarded exactly the direction separating the two velocities. Redundancy is not unpredictability—all four coordinates regress well when they are all targeted.

Frame stacking. Positions are recoverable from a short history but velocities are not. Increasing the input from 3 stacked frames (9 channels) to 12 (36 channels) leaves the positional coordinates essentially unchanged while transforming the velocities:

K.3.1 Theory–Experiment Bridge for Orthogonality

We measured the orthogonality of learned inverted-pendulum encoder representations using the feature cosine matrix whose (i, j)-th entry is defined as

$$C_{ij} = \frac{\langle \mathbf{w}_i, \mathbf{w}_j \rangle}{\|\mathbf{w}_i\|_2 \, \|\mathbf{w}_j\|_2} = \cos\theta_{ij}, \tag{53}$$

where $\mathbf{w}_i$ is the row of W associated with state dimension i:

$$W = \arg\min_W \|SW - \Phi\|_F^2 + \alpha \|W\|_F^2 = \left(S^\top S + \alpha I_d\right)^{-1} S^\top \Phi, \qquad \alpha = 10^{-3}, \tag{54}$$

where Φ is the feature matrix. The matrix for the inverted pendulum is shown in Table 7.

Table 7: Correlation matrix (mean $\pm$ std).

	`cart_pos`	`pole_ang`	`cart_vel`
`cart_pos`	$+1.000 \pm 0.000$	$+0.524 \pm 0.062$	$+0.025 \pm 0.096$
`pole_ang`	$+0.524 \pm 0.062$	$+1.000 \pm 0.000$	-0.365 ± 0.051
`cart_vel`	$+0.025 \pm 0.096$	-0.365 ± 0.051	$+1.000 \pm 0.000$

L Details of Distillation Fine Tuning for Large Language Models

We first describe the student teacher setup and how rewards are defined for this setting. Then we describe the details of factored manifold Muon optimizer which we used, including the underlying method.

L.1 Reward structure in distillation fine-tuning

LLMs can be viewed as autoregressive policies over tokens. Given a prompt x and a generated prefix $y_{<t}$, the state is the token history $(x, y_{<t})$ and the action is the next token $y_t \in \mathcal{V}$, where $\mathcal{V}$ is the token vocabulary. Thus a language model with parameters θ defines the policy

$$\pi_\theta(y_t \mid y_{<t}, x).$$

Distillation fine-tuning replaces an externally specified scalar reward with a dense reward derived from the disagreement between a student policy and a teacher policy. In self-distillation fine-tuning (Shenfeld et al., 2026), the teacher is obtained by conditioning the model on additional demonstration context c, while the student is conditioned only on the original prompt. Thus the teacher distribution is

$$\pi_T(y_t \mid y_{<t}, x, c),$$

whereas the student distribution is

$$\pi_\theta(y_t \mid y_{<t}, x).$$

The distillation objective minimizes the reverse KL divergence

$$\mathcal{L}_{\mathrm{DFT}}(\theta) = D_{\mathrm{KL}}(\pi_\theta(\cdot \mid x) \,\|\, \pi_T(\cdot \mid x, c)) \,.$$

Equivalently, at the token level, this objective induces the implicit reward

$$r_t^{\mathrm{DFT}}(y_t \mid y_{<t}, x, c) = \log \pi_T(y_t \mid y_{<t}, x, c) - \log \pi_{\theta_k}(y_t \mid y_{<t}, x),$$

where θ_k denotes the current student parameters at the beginning of the policy update. The trajectory-level reward is therefore

$$R^{\mathrm{DFT}}(y \mid x, c) = \sum_t r_t^{\mathrm{DFT}}(y_t \mid y_{<t}, x, c).$$

This reward is large when the generated token is assigned high probability by the demonstration-conditioned teacher and low probability by the current student. Hence the reward encourages the student to move toward the behavior induced by the teacher context, while still training on trajectories sampled from the student itself.

In our setting, the distinction is that the teacher has access to the additional context c, but its weights are not updated as an exponential moving average of the student. That is, we use a fixed contextual teacher. This differs from the default self-distillation parameterization by Shenfeld et al. (2026), where the teacher is demonstration-conditioned but its weights may be taken as an exponential moving average of the student. This is done to save on compute while retaining meaningful performance.

Our code for this is based off of the code provided by Shenfeld et al. (2026).

L.2 Factored Manifold Muon

Manifold Muon (Bernstein, 2025a) is a first-order method for optimizing a matrix parameter $W \in \mathbb{R}^{m\times n}$, $m \geq n$, under the Stiefel constraint $W^\top W = I$. Given a gradient $G \in \mathbb{R}^{m\times n}$ and step size $\eta > 0$, the method chooses a tangent update direction by solving

$$\min_A \langle A, G\rangle \quad \text{s.t.} \quad \|A\|_* \leq \eta, \qquad A^\top W + W^\top A = 0. \tag{55}$$

The second constraint enforces $A \in T_W \,\mathrm{St}(m,n)$, while the nuclear-norm constraint gives the Muon update its spectral normalization.For any matrix M, write its singular value decomposition as

$$M = X\Gamma Y^\top.$$

The exact matrix sign is then

$$\mathrm{msign}(M) = XY^\top.$$

The parameter is then retracted back to the Stiefel manifold by

$$W \leftarrow \mathrm{msign}(W + A),$$

The primal problem equation 55 does not admit a simple closed-form solution. Manifold Muon therefore solves the equivalent dual problem

$$\min_\Lambda \left\| G + W(\Lambda + \Lambda^\top)\right\|_*,$$

originally using subgradient descent. Since the nuclear norm is non-smooth, this inner optimization has the slow $O(k^{-1/2})$ convergence rate.

Buchanan (2025) accelerate this inner loop using the Alternating Direction Method of Multipliers (ADMM) (Neal et al., 2011). They introduce a slack variable X and impose the linear constraint

$$X = G + W(\Lambda + \Lambda^\top),$$

which separates the non-smooth nuclear-norm term from the quadratic coupling through W. This variable-splitting reformulation yields closed-form updates: a symmetric least-squares step for Λ and a singular-value-thresholding step for X. One ADMM iteration is

$$\Lambda_{k+1} = \frac{1}{2}\,\mathrm{sym}\left[W^\top\left(\frac{1}{\rho}\Omega_k + Z_k - G\right)\right], \tag{56}$$

$$M_{k+1} = 2W\Lambda_{k+1} + G - \frac{1}{\rho}\Omega_k, \tag{57}$$

$$M_{k+1} = X_{k+1}\Gamma_{k+1}Y_{k+1}^\top, \tag{58}$$

$$Z_{k+1} = X_{k+1}\left(\Gamma_{k+1} - \frac{1}{\rho}I\right)_+ Y_{k+1}^\top, \tag{59}$$

$$\Omega_{k+1} = \Omega_k + \rho\left(Z_{k+1} - 2W\Lambda_{k+1} - G\right). \tag{60}$$

where

$$\mathrm{sym}(M) = \frac{1}{2}(M + M^\top), \qquad (\Gamma - \alpha I)_+ = \mathrm{diag}\left((\gamma_i - \alpha)_+\right).$$

Here $\rho > 0$ is the ADMM penalty parameter and Ω is the dual variable. After K ADMM iterations, the tangent update direction is recovered as

$$A_\star = -\eta \ \mathrm{msign}\left(G + 2W\Lambda_K\right).$$

Equivalently, if

$$G + 2W\Lambda_K = X'\Gamma'Y'^\top,$$

then

$$A_\star = -\eta X'Y'^\top.$$

Empirically, the ADMM inner loop reaches an accurate update in far fewer iterations than subgradient descent and is less sensitive to the choice of ρ. We denote by $\mathrm{ADMM}\ -\mu$ the ADMM muon operator that given G returns the appropriate A_* over k-steps.

Riemannian gradient on the symmetric positive definite manifold. In the fixed-rank factorization

$$M = X'\Gamma'Y'^\top,$$

the middle factor $\Gamma' \in \mathbb{R}^{r\times r}$ is symmetric positive definite. Thus

$$\Gamma' \in \mathcal{S}^r_{++} := \{\Gamma' \in \mathbb{R}^{r\times r} : \Gamma' = \Gamma'^\top,\ \Gamma' \succ 0\}.$$

This factor should be distinguished from the singular-value matrix Γ in the SVD $M = X\Gamma Y^\top$; here Γ' is an SPD factor, not necessarily diagonal.

We equip $\mathcal{S}^r_{++}$ with the affine-invariant metric. At a point $\Gamma' \in \mathcal{S}^r_{++}$, the tangent space is $T_{\Gamma'}\mathcal{S}^r_{++} = \mathrm{Sym}^r$, and for $U, V \in T_{\Gamma'}\mathcal{S}^r_{++}$ the metric is

$$\langle U, V\rangle_{\Gamma'} = \mathrm{tr}\Big(\Gamma'^{-1}U\Gamma'^{-1}V\Big). \tag{61}$$

Let $\mathcal{L}$ be a smooth loss and let $\nabla_{\Gamma'}\mathcal{L} \in \mathbb{R}^{r\times r}$ denote its Euclidean gradient with respect to Γ'. The Riemannian gradient $\mathrm{grad}_{\Gamma'}\,\mathcal{L} \in \mathrm{Sym}^r$ is defined by

$$\langle \mathrm{grad}_{\Gamma'}\,\mathcal{L}, \xi\rangle_{\Gamma'} = \mathrm{tr}\big((\nabla_{\Gamma'}\mathcal{L})^\top\xi\big) \qquad \forall \xi \in \mathrm{Sym}^r. \tag{62}$$

Since tangent directions on $\mathcal{S}^r_{++}$ are symmetric, only the symmetric part of the Euclidean gradient contributes. Therefore

$$\mathrm{grad}_{\Gamma'}\,\mathcal{L} = \Gamma'\ \mathrm{sym}(\nabla_{\Gamma'}\mathcal{L})\,\Gamma', \qquad \mathrm{sym}(M) := \frac{1}{2}(M + M^\top). \tag{63}$$

A Riemannian gradient step with step size $\alpha > 0$ is given by the affine-invariant exponential map:

$$\Gamma' \leftarrow \mathrm{Exp}_{\Gamma'}(-\alpha\,\mathrm{grad}_{\Gamma'}\,\mathcal{L}) = \Gamma'^{1/2}\exp\Big(-\alpha\,\Gamma'^{1/2}\,\mathrm{sym}(\nabla_{\Gamma'}\mathcal{L})\,\Gamma'^{1/2}\Big)\,\Gamma'^{1/2}. \tag{64}$$

This update preserves symmetry and positive definiteness, so the middle factor remains on $\mathcal{S}^r_{++}$ throughout training.

Finally, the induced change in the full matrix W follows by differentiating the factorization

$$W = X'\Gamma'Y'^\top.$$

By the product rule,

$$\dot W = \dot X'\Gamma'Y'^\top + X'\dot\Gamma'Y'^\top + X'\Gamma'\dot Y'^\top.$$

Thus changes in W decompose into changes in the left orthogonal factor, the SPD middle factor, and the right orthogonal factor. If X' and Y' follow Stiefel-gradient flow and Γ' follows the affine-invariant SPD gradient flow, then

$$\dot W = -\,\mathrm{grad}_{\mathrm{St}}(X';\nabla_{X'}\mathcal{L})\Gamma'Y'^\top - X'\Gamma'\,\mathrm{sym}(\nabla_{\Gamma'}\mathcal{L})\,\Gamma'Y'^\top - X'\Gamma'\left(\mathrm{grad}_{\mathrm{St}}(Y';\nabla_{Y'}\mathcal{L})\right)^\top.$$

Equivalently, to first order,

$$\Delta W = \Delta X'\Gamma'Y'^\top + X'\Delta\Gamma'Y'^\top + X'\Gamma'\Delta Y'^\top.$$

This makes explicit how the orthogonal factors move along their Stiefel manifolds, while the central factor remains on $\mathcal{S}^r_{++}$.

Table 8: Summary of fine-tuning datasets used for DFT evaluation.

Dataset	Task	Train	Test	Description
Simplified ToolUse	Function calling	∼13,000	∼1,000	Single-turn instruction-following tasks requiring the model to select and invoke the correct API tool from a predefined schema given a natural-language query; simplified to single-step calls without multi-turn chaining (Zhuang et al., 2023).
Banking77	Intent classification	4,046	500	Fine-grained banking intent detection benchmark (Casanueva et al., 2020) in which the model must classify a customer's natural-language query into exactly one of 77 closely related banking intents (e.g., distinguishing *card_not_working* from *card_swallowed*), with the full intent inventory provided in the prompt and evaluation by normalized exact match on the predicted label.
CoT Math	Mathematical reasoning	∼7,500	∼1,319	Grade-school and competition-level arithmetic and algebra problems requiring multi-step chain-of-thought (Wei et al., 2022) reasoning; the target output is a full reasoning trace followed by a boxed numerical answer (Cobbe et al., 2021).



Figure 13: Vanilla Muon fails to learn on tooluse dataset, with DFT, which we use to tune the hyper-parameters.

L.3 Runtime and Implementation Details

We obtain about 0.042 steps per second with factored manifold muon on average and 0.055 with AdamW. All training is on H100 GPUs with 80 GB RAM. We search over three learning rates: 1.5e-3, 5e-4, 1.5e-4, on tooluse datset to find 1.5e-3 to be the best learning rate. We always scale down the LR for the symmetric positive definite part (Γ') by 3 and this proved to stabilize the learning: meaning loss term varied a lot less across steps. We also scale the learning rates for ADMM manifold muon and the positive symmetric positive definite matix manifold. We also grid search over three learning rates for AdamW: 1e-6, 1e-5, 5e-5, and find 1e-6 to be the best. We also tried vanilla Muon (Jordan et al., 2024) for DFT on tooluse and sweep over the learning rate combinations for muon and AdamW: (1e-3, 5e-4), (5e-3, 2e-3), (1e-4, 5e-5), (3e-4, 1e-4), (7e-4, 1e-4), but none of them yield any improvement on the loss (see Figure 13). The best hyper-parameters for our factored manifold Muon are presented in Table 9. all other hyper-parameters related to DFT remain the same.

M Impact Statement

Potential positive impacts include more stable and sample-efficient reinforcement learning from high-dimensional observations which would lower energy costs and be more accessible to smaller

Table 9: Hyperparameters for factored manifold Muon.

Category	**Hyperparameter**	**Value**
Model	Base model	Qwen3-4B
	Parameters	4.02B
Training	$(\beta_1, \beta_2, \varepsilon)$	$(0.9, 0.999, 10^{-8})$
	LR schedule	Cosine
	Warmup ratio	0.1
	Per-device batch size	1
	Gradient accumulation	64
	Max grad norm	1.0
	Epochs	1
	Precision	bfloat16
Projected gradient	UV update scale	1.5e-3
	SPD update scale (Γ')	5e-4
	Retraction	Exponential map
DFT	Loss type	DFT
	Clip ε	0.2
	KL penalty β	0
	Temperature	1.0
	Reward scaling	Group normalization
	Completions per prompt	8
	Max prompt length	1024
	Max completion length	512
	Importance sampling	Token-level
	IS cap	2
	Ref model mixup α	0.00
	Steps per generation	64
	Generation batch size	64
Dataset	Task	CoT Math, Banking77, Tooluse

research groups. Of course the advancement of such methods can be misappropriated to train harmful robots or LLMs, however we view these risks as indirect.