Learning in Low-Dimensional Subspaces: Orthogonal Bottlenecks for Reinforcement Learning

Aleksandar Todorov, Matthia Sabatelli

Keywords: Low-dimensional Representations, Orthogonality, Deep Reinforcement Learning, Manifold Hypothesis

Summary

Deep reinforcement learning agents typically use high-dimensional neural representations, despite growing evidence that task-relevant value and policy structure may be intrinsically low-dimensional. We study a simple architectural prior that enforces such structure directly by inserting a fixed orthonormal projection between the encoder and downstream heads, constraining representations to operate within a low-dimensional subspace without auxiliary objectives, pretraining, or changes to the underlying RL algorithm. Under a linear realizability assumption, we show that once the bottleneck dimension exceeds the intrinsic rank of the optimal value function in feature space, the bottleneck preserves expressivity and does not alter the induced learning dynamics. Empirically, across single- and multi-task benchmarks, performance is typically recovered once the bottleneck dimension exceeds a small task-dependent threshold. In small domains, we visualize low-dimensional value manifolds, while in larger benchmarks, we observe sharp performance recovery as the bottleneck dimension increases. Diagnostic analyses further show that fixed orthogonal bottlenecks stabilize feature norms and are associated with higher effective rank, while learned projections can be unstable in some regimes. Together, these results suggest that deep reinforcement learning representations can often be faithfully compressed into low-dimensional orthogonal subspaces, and that fixed orthogonal bottlenecks offer a simple mechanism for shaping representation geometry.

Contribution(s)

1. We provide a theoretical analysis showing that fixed orthogonal bottlenecks preserve expressivity and the induced optimization dynamics under a linear realizability assumption.
   Context: Linear realizability assumptions are widely used in reinforcement learning theory to analyze value-based function approximation (Du et al., 2020; Weisz et al., 2023), but are rarely linked to architectural constraints in deep RL. We show that when the optimal value function is realizable by a linear map in feature space, inserting a fixed orthonormal bottleneck whose dimension meets or exceeds the intrinsic rank does not reduce representational capacity and yields learning dynamics equivalent to an explicit low-dimensional parameterization. This provides a principled justification for constraining learned representations via fixed orthogonal subspaces.
2. We empirically demonstrate that deep reinforcement learning representations can be compressed into low-dimensional orthogonal subspaces while preserving performance relative to the unconstrained baseline, and we analyze the resulting representation geometry.
   Context: Across Classic Control, Atari, Brax MuJoCo, and multi-task Meta-World benchmarks, we find that baseline performance is typically recovered once the bottleneck dimension exceeds a small task-dependent threshold. In small domains, we visualize low-dimensional value manifolds; in larger-scale settings, we analyze performance trends and diagnostics such as effective rank. We further compare fixed and trainable projections, identifying regimes in which learning the projection, instead of imposing it, introduces instability and representation collapse.

Learning in Low-Dimensional Subspaces: Orthogonal Bottlenecks for Reinforcement Learning

Aleksandar Todorov[1,⋆], **Matthia Sabatelli**[1]
a.todorov.4@student.rug.nl, m.sabatelli@rug.nl

[1]**University of Groningen, Groningen, The Netherlands**

⋆ Corresponding author

Abstract

Deep reinforcement learning (RL) agents commonly rely on high-dimensional neural representations, despite growing evidence that task-relevant value and policy structure may be intrinsically low-dimensional. In this work, we present a simple yet effective representation-level prior that inserts a fixed orthonormal projection to constrain encoder features to a low-dimensional subspace, requiring no auxiliary objectives, pretraining, or changes to the underlying RL algorithm. Under a linear realizability assumption, we prove that when the bottleneck dimension exceeds the intrinsic rank of the optimal value function in feature space, the bottleneck preserves expressivity and leaves the induced gradient dynamics unchanged up to an equivalent low-dimensional parameterization. Empirically, we find that across both single and multi-task benchmarks, baseline performance is either matched or improved once the bottleneck dimension exceeds a small task-dependent threshold; in many cases, value representations can be compressed to extremely low dimensions without loss, and the minimal sufficient dimension depends far more on environment complexity than encoder width. In addition, we analyze representation geometry and find that orthogonal bottlenecks stabilize feature norms and are associated with higher effective rank. Together, these results support a representation-space interpretation of the manifold hypothesis in reinforcement learning and position orthogonal bottlenecks as a lightweight, architecture-agnostic mechanism for shaping RL representations.

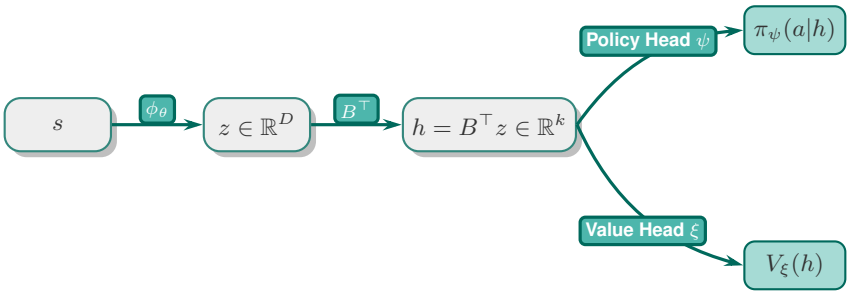


Figure 1: A simple, visual representation of the orthogonal bottleneck for deep reinforcement learning adapted to a typical Actor-Critic architecture. After encoding the state into features $z \in \mathbb{R}^D$, a fixed orthonormal projection constrains the representation to a k-dimensional subspace before feeding the policy and value heads.

1 Introduction

Reinforcement learning (RL) agents are routinely equipped with highly over-parameterized neural representations, even when solving tasks whose underlying decision structure is comparatively simple. While deep networks offer the flexibility to model complex policies and value functions, it is natural to question to what extent standard deep RL architectures allocate representational capacity far beyond what the task itself demands. This mismatch between network capacity and task complexity aligns naturally with the manifold hypothesis in machine learning, which posits that high-dimensional data and learned representations often concentrate near low-dimensional manifolds embedded in ambient space (Goldberg et al., 2008; Bengio et al., 2014; Fefferman et al., 2016; Meilă & Zhang, 2023). Recent evidence suggests that such low-dimensional structure also emerges in deep RL. On the policy side, Mutti et al. (2022) and Tenedini et al. (2026) show that the space of behaviorally distinct policies realized by RL agents is effectively low-dimensional, allowing policy networks to be compressed by several orders of magnitude in parameter space without loss of behavioral expressivity. On the representation learning side, both model-free and model-based approaches aim to recover compact latent state manifolds that reflect task dynamics, typically through auxiliary or contrastive objectives that guide an encoder toward structured representations (Oord et al., 2019; Zhang et al., 2021; Echchahed & Castro, 2025). These approaches, however, generally treat the manifold as a structure to be either discovered through optimization or to be recovered post hoc via generative modeling.

In this work, we take a different approach and study an alternative perspective: rather than encouraging the network to uncover a low-dimensional structure, we impose one explicitly through the architecture itself at the representation level. Among the many possible ways to impose a low-dimensional bottleneck on learned representations, our focus is on fixed orthonormal subspaces as they allow for representations that capture the intrinsic subspace of a task without redundancy or distortion. In fact, any linear map onto a k-dimensional subspace can be factorized into an orthonormal basis followed by a diagonal scaling; retaining only the orthonormal component yields a full-rank bottleneck in which all directions are treated uniformly and no single eigendirection dominates the representation. Orthogonal projections also enjoy favorable geometric properties, with random orthogonal maps approximately preserving distances with high probability, as formalized by the Johnson-Lindenstrauss lemma (Johnson & Lindenstrauss, 1984; Tipping & Bishop, 1999; Ghojogh et al., 2022), and orthogonal weight matrices are known to improve conditioning and gradient propagation in deep networks through non-expansiveness (Saxe et al., 2014; Hu et al., 2019). Motivated by these observations, we study a simple architectural inductive bias, namely, after an encoder produces features $z \in \mathbb{R}^D$, we project them onto a fixed orthonormal basis $B \in \mathbb{R}^{D \times k}$ with $B^\top B = I_k$ and $k \leq D$ via $h = B^\top z$, and feed only the compressed representation $h \in \mathbb{R}^k$ to all downstream value and/or policy heads. This explicitly constrains the agent to operate within a k-dimensional orthogonal representation subspace, without altering the underlying learning algorithm or training objective. Figure 1 provides a visual overview of this architecture in a standard actor-critic setting.

Our contributions are as follows. First, we provide theoretical guarantees showing that if the optimal value function is linearly realizable in feature space with intrinsic rank r, then a fixed orthogonal bottleneck of dimension $k \geq r$ preserves expressivity and does not alter the induced gradient dynamics of the effective feature-to-representation mapping. Second, we empirically validate this approach across both single-task and multi-task benchmarks and multiple deep RL algorithms. We show that once k exceeds a small task-dependent threshold, low-dimensional orthogonal subspaces typically match (and sometimes improve upon) baseline performance, while yielding more stable and uniformly utilized representations as measured by diagnostics such as feature norms and effective rank.

2 Related Work

Our work connects to two broad lines of research. The first studies low-dimensional structure in reinforcement learning, either in policy space or in learned state and value representations, motivated

by the manifold hypothesis. The second investigates the role of orthogonality in deep networks as a mechanism for stabilizing signal and gradient propagation. While both bodies of work provide important insights into representation structure and training dynamics, they have largely been explored independently. We combine these perspectives by inserting a fixed orthonormal projection between the encoder and the policy and value heads. In turn, this makes the dimension available to the heads a controlled architectural parameter, allowing us to study its effect without auxiliary objectives, pretraining, or changes to the underlying reinforcement learning algorithm.

Low-dimensional structure and representations. A large body of work in reinforcement learning assumes that task-relevant variability is intrinsically low-dimensional and exploits this structure to design compact policy and state representations, often motivated by the manifold hypothesis (Narayanan & Mitter, 2010; Fefferman et al., 2016). Unsupervised skill- and option-discovery methods explicitly construct low-dimensional latent spaces of behaviors or tasks using information-theoretic or variational objectives to encourage diversity and structure in the learned policy space (Frans et al., 2017; Hausman et al., 2018; Achiam et al., 2018; Eysenbach et al., 2018; Laskin et al., 2022). More recent work makes a policy-manifold perspective explicit by learning generative models over policy parameters and compressing the policy space into low-dimensional latent codes (Rakicevic et al., 2021; Mutti et al., 2022; Tenedini et al., 2026). Parallel lines of research in model-based and transfer reinforcement learning learn latent state, action, or policy embeddings to simplify control and enable fast adaptation across related tasks (Zhang et al., 2019; Arnekvist et al., 2019; Rana et al., 2022). State (and state-action) representation learning methods similarly aim to recover compact, task-relevant features, typically through contrastive, predictive, or reconstruction-based objectives (Oord et al., 2019; Zhang et al., 2021; Echchahed & Castro, 2025). On the theory side, work on linearly realizable value functions and good feature representations formalizes a related notion of low-dimensional linear structure in feature space, primarily from a worst-case sample complexity perspective (Du et al., 2020; Lattimore et al., 2020; Weisz et al., 2023). Finally, empirical studies of deep RL representations link geometric properties such as feature rank, isotropy, and neuron dormancy to loss of plasticity and poor long-term learning, highlighting how collapsed or unstable representations impede adaptation (Lyle et al., 2023; Sokar et al., 2023; Klein et al., 2024; Todorov et al., 2025).

Orthogonality and signal propagation. Orthogonal and orthonormal weight matrices have long been studied as a means of stabilizing signal and gradient propagation in deep networks (Xiao et al., 2018; Jia et al., 2019; Yang et al., 2019; Huang et al., 2020). In deep linear networks, Saxe et al. (2014) and Hu et al. (2019) show that orthogonal initialization can provably accelerate gradient descent compared to Gaussian initialization, and related techniques combining orthogonality with layerwise variance normalization enable reliable training of very deep convolutional architectures (Mishkin & Matas, 2016). In nonlinear feedforward networks, work on dynamical isometry demonstrates that when the singular values of the input-output Jacobian remain close to one, which is approximately achieved by random orthogonal weights in wide networks, gradients neither vanish nor explode, leading to substantially faster training (Pennington et al., 2017; 2018; Xiao et al., 2018). Closely related ideas have also been applied to recurrent architectures, where orthogonal or unitary transition matrices help preserve gradient norms over long sequences and improve learning of long-term dependencies (Henaff et al., 2016; Arjovsky et al., 2016; Chen et al., 2018; Gilboa et al., 2019).

3 Orthogonal Bottlenecks Preserve Expressivity and Optimization

We start by studying the representational and optimization properties induced by orthogonal bottlenecks from a theoretical perspective. Our goal is to understand whether inserting a fixed k-dimensional orthonormal projection between the encoder and downstream heads preserves expressivity, and whether learning with such a projection alters the gradient dynamics compared to directly training a k-dimensional representation. To this end, we analyze the value estimation problem underlying a broad class of modern RL algorithms under a linear realizability assumption. This setting allows us to make precise statements about representational sufficiency and optimization behavior, and provides a

way of interpreting empirical behavior in terms of low-rank linear structure in the learned feature space. We consider the standard supervised regression formulation of value learning, which serves as an inner loop for many reinforcement learning algorithms. For definitions on reinforcement learning and function approximation, see Appendix B.

We assume access to a sufficiently rich encoder

$$\phi : \mathcal{S} \to \mathbb{R}^D,$$

which maps states $s \in \mathcal{S}$ to a high-dimensional feature space. The target function (e.g., the optimal value function $V^\star$) is assumed to be linear in this feature space.

Assumption 3.1 (Linear realizability)**.** *There exists a matrix $\Theta^\star \in \mathbb{R}^{m \times D}$ such that for all $s \in \mathcal{S}$,*

$$V^\star(s) = \Theta^\star \phi(s).$$

The linear realizability assumption is standard in reinforcement learning theory (Lattimore et al., 2020; Du et al., 2020; Weisz et al., 2023) and formalizes the idea that a sufficiently expressive encoder can transform raw observations into a feature space where values are approximately linear. Importantly, this assumption does not require the overall network to be linear, as the encoder ϕ may be arbitrarily deep and nonlinear, and the heads following the bottleneck may also be nonlinear. The assumption only concerns the existence of a linear representation of $V^\star$ in feature space.

Now, let $z \in \mathbb{R}^D$ denote the encoder output, and let $B \in \mathbb{R}^{D \times k}$ be a matrix with orthonormal columns, i.e., $B^\top B = I_k$ with $k \leq D$. The bottleneck representation is defined as $h = B^\top z \in \mathbb{R}^k$, and only h is provided as input to the value and policy heads.

For value learning, we consider a generic differentiable function approximator

$$H(h; \theta) \approx V(s),$$

where θ collects all parameters after the bottleneck, $H : \mathbb{R}^k \to \mathbb{R}^m$ is expressive enough to realize at least a linear layer, as is the case for standard multi-layer perceptron (MLP) heads, and V denotes the value function. A natural question is then: for which values of k does this architecture retain the ability to represent $V^\star$, and does the presence of the fixed projection B alter the optimization dynamics relative to directly learning a k-dimensional representation? These questions are addressed by the following proposition. The full proof is deferred to Appendix B.

Proposition 3.2. *Assume $V^\star$ is linearly realizable in feature space with rank $r = \operatorname{rank}(\Theta^\star)$, and let $H(h; \theta)$ be any head expressive enough to realize at least a linear layer. For any $k \geq r$ and orthonormal $B \in \mathbb{R}^{D \times k}$:*

1. ***(Representational sufficiency).*** *There exist encoder parameters and head parameters $\theta^\star$ such that the network*

$$s \;\mapsto\; H\big(B^\top z(s); \theta^\star\big)$$

exactly realizes $V^\star(s)$ for all $s \in \mathcal{S}$. In particular, once $k \geq r$, inserting a fixed orthogonal bottleneck does not reduce expressivity relative to the given feature space.

2. ***(Trainability):*** *Let $W \in \mathbb{R}^{D \times D}$ be the encoder's final layer and $A_t = B^\top W_t$ the composite feature-to-bottleneck map. Training (θ, W) by gradient descent on loss $\mathcal{L}$ evolves A_t identically to training the direct parameterization $h = C\phi(s)$ on (θ, C), given $C_0 = A_0$.*

Proposition 3.2 establishes that once k matches the intrinsic rank, expressivity is preserved, and orthogonality avoids introducing an additional linear preconditioner into the updates of the feature-to-bottleneck map. In addition, sharing a single orthogonal bottleneck across all value and/or policy heads constrains different learning objectives to operate within the same k-dimensional subspace, rather than allowing each head to learn an independent low-rank projection. This encourages a common representation geometry across objectives.

Why Orthogonality Matters. A key requirement in Proposition 3.2 is that the projection matrix satisfies the orthogonality condition $B^\top B = I_k$. In particular, it ensures that the induced gradient dynamics on the effective bottleneck map $A_t = B^\top W_t$ are identical to those of a standard k-dimensional parameterization. If B is fixed but non-orthogonal, the update for A_t is instead preconditioned by $B^\top B$, which can amplify dominant singular directions and lead to unstable scaling. To illustrate why this matters in practice, Figure 2 compares a fixed orthonormal projection to a fixed projection sampled from a standard Gaussian distribution of the same bottleneck dimension. While both bottlenecks impose the same dimension k, the Gaussian projection exhibits rapidly growing feature norms and degraded performance, whereas the orthonormal projection trains reliably with stable feature scales. This also provides empirical support for Assumption 3.1 (linear realizability). If the optimal value function were not well-approximated by a low-rank linear map in the learned feature space, then small bottleneck dimensions would necessarily limit performance. In the following sections, we interpret the recovery threshold in k as an empirical falsification test: across environments, performance is preserved once k exceeds a small task-dependent value, consistent with the presence of approximately low-rank linear structure in learned value representations.

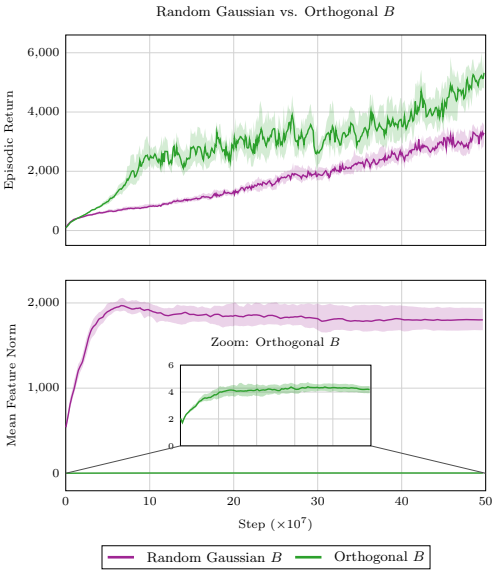


Figure 2: For PPO in Humanoid, feature norms explode when using a fixed Gaussian projection B and lower performance is achieved, while a fixed orthonormal B learns reliably. Both bottlenecks use $k = 8$.

4 Experimental Setup

We evaluate orthogonal projection as a representation prior across a diverse suite of environments and algorithms with the goal of testing how performance depends on the bottleneck dimension k, and to empirically probe the theoretical predictions of Section 3 regarding representational sufficiency and the role of orthogonality. We compare a standard baseline agent with no bottleneck to variants that insert a k-dimensional projection, varying the bottleneck dimension and whether the projection matrix B is fixed orthogonal or trained end-to-end. In selected experiments, we additionally vary the encoder width D while keeping k fixed to assess the role of encoder capacity once the representation dimension is constrained.

4.1 Environments and Algorithms

We consider five families of environments spanning increasing perceptual and control complexity: Classic Control (Towers et al., 2025), MinAtar (Young & Tian, 2019), Atari (Bellemare et al., 2013), Brax MuJoCo (Todorov et al., 2012; Freeman et al., 2021), and Meta-World MT10 (Yu et al., 2021). These domains allow us to study representation compression in low-dimensional state spaces, pixel-based environments, continuous control, and multi-task learning. Across these environments, we use standard model-free deep RL algorithms appropriate to each setting: DQN (Mnih et al., 2015) for Classic Control, PQN (Gallici et al., 2025) for Atari, and PPO (Schulman et al., 2017) for MinAtar, MuJoCo, and Meta-World.

4.2 Architecture and Implementation

All agents share a common architecture of the form as shown in Figure 1, where ϕ_θ is a convolutional encoder for MinAtar and Atari and an MLP encoder for all other environments. The matrix $B \in$

$\mathbb{R}^{D\times k}$ is sampled once by QR decomposition of a Gaussian matrix and held fixed throughout training, with only the encoder and heads updated. This enforces a strict k-dimensional information bottleneck without introducing additional trainable parameters. The projection requires $\mathcal{O}(Dk)$ operations for the forward pass and a similar cost for backpropagation, and for typical configurations ($D = 256, k = 4$), this is negligible compared to evaluating the encoder or interacting with the environment.

4.3 Training and Evaluation Protocol

Each configuration is run with 10 random seeds. For all experiments, we report the interquartile mean (IQM) of returns with 95% stratified bootstrap confidence intervals, following the recommendations of Agarwal et al. (2021). To ensure a fair comparison, we use the best hyperparameters found for the unconstrained baseline agent. Complete hyperparameters, training budgets, and diagnostic definitions are provided in Appendix A.

5 Results

We evaluate orthogonal bottlenecks across environments of increasing complexity, with the goal of understanding how small a representation subspace can be without sacrificing performance, and how orthogonal projections shape the geometry of learned value representations. We begin with small domains where representation structure can be visualized directly, then move to large-scale and multi-task benchmarks to test whether similar compressibility holds at scale.

5.1 Classic Control and MinAtar: Small Bottlenecks and Value Manifolds

Across Classic Control tasks, we find that a bottleneck of size $k = 2$ is sufficient to recover baseline performance, while $k = 1$ leads to clear degradation or failure. Increasing the bottleneck dimension beyond $k = 2$ provides no additional benefit. MinAtar exhibits the same qualitative behavior: despite pixel-based observations, baseline performance is matched once $k \in \{1, 2, 3\}$, indicating that task-relevant value information lies in a very low-dimensional subspace. Full learning curves are provided in Appendix E. To understand how such small representations suffice, we visualize bottleneck activations collected along evaluation trajectories (greedy for DQN). Figure 3 shows two-dimensional embeddings for Acrobot-v1 (DQN) and Freeway-MinAtar (PPO) with $k = 2$. In both cases, the embeddings concentrate on a thin, low-dimensional manifold rather than filling the ambient space. Coloring by the agents' value estimates reveals a smooth gradient along the manifold, while coloring by action reveals structured, partially overlapping regions on the manifold. Qualitatively, for Acrobot, a single episode trajectory progresses from low-value regions toward higher-value regions in a largely monotone fashion. For Freeway, trajectories are less monotone and repeatedly revisit parts of the manifold as the agent alternates between waiting, positioning, and crossing behaviors. This difference is consistent with the more reactive, cyclic dynamics of Freeway compared to the more goal-directed progression in Acrobot. When the bottleneck dimension is increased to $k = 3$, the embeddings remain strongly concentrated near a two-dimensional structure in $\mathbb{R}^3$ (Figure 4) with only modest thickness along the third coordinate. Notably, these additional variations are not aligned with the agent's value estimates, suggesting that the additional dimension is largely unused and that the intrinsic value representation remains effectively two-dimensional even when extra capacity is available.

5.2 Large-Scale Benchmarks: Atari and Brax MuJoCo

We next evaluate on the Atari-5 benchmark (Battle Zone, Double Dunk, Name This Game, Phoenix, and Q*bert) (Aitchison et al., 2022) and on four MuJoCo tasks (Reacher, Pusher, HalfCheetah, and Humanoid), chosen to span a range of perceptual and dynamical complexity.

Figure 5 reports final performance as a function of bottleneck dimension k. Across all tasks, learning fails when k is too small but reliably recovers once it exceeds a modest, task-dependent threshold.

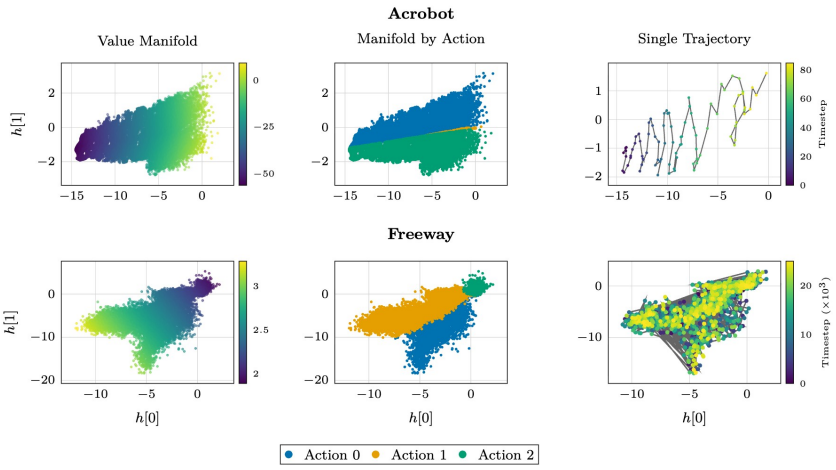


Figure 3: Bottleneck manifolds for Acrobot-v1 (DQN, top) and Freeway-MinAtar (PPO, bottom) with a fixed orthogonal bottleneck of size $k = 2$. Each point is a visited state-action pair encoded into $(h[0], h[1])$, colored by the agent's value estimates (left) or action (middle); in both tasks, representations concentrate on a thin manifold with a smooth value gradient and structured action regions. The right column shows a single evaluation episode colored by timestep: Acrobot trajectories progress largely monotonically toward higher-value regions, while Freeway trajectories repeatedly revisit parts of the manifold reflecting more reactive task dynamics.

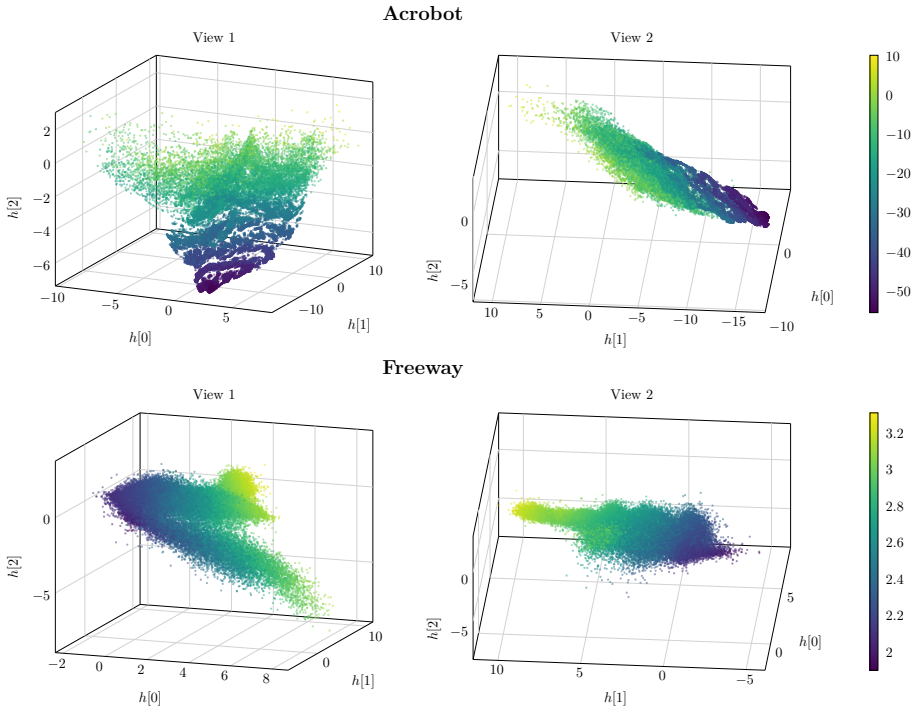


Figure 4: Three-dimensional bottleneck embeddings for Acrobot-v1 (DQN, top) and Freeway-MinAtar (PPO, bottom) with a fixed orthogonal bottleneck of size $k = 3$, colored by the agent's value estimates. Each environment is shown from two viewing angles. In both cases, the representations concentrate near a low-dimensional structure in $\mathbb{R}^3$ rather than filling the volume, with slight thickness along the third coordinate. These additional variations are not strongly aligned with the value estimates, consistent with the observation that increasing k beyond the recovery threshold does not change performance.

Beyond this point, performance saturates and remains statistically indistinguishable from the no-projection baseline. While the precise value of $k_{\min}$ varies across tasks, reflecting differences in dynamical complexity and control requirements, in all cases, it remains small relative to the encoder width (256 in MuJoCo, 512 in Atari). In some environments, such as Reacher, moderate bottlenecks slightly outperform the baseline, whereas in others, performance saturates near the lower end of the baseline confidence interval. These results mirror the behavior observed in smaller domains: high-dimensional encoder features can be compressed into a low-dimensional orthogonal subspace without loss of performance, provided the bottleneck dimension exceeds the intrinsic rank of the task.

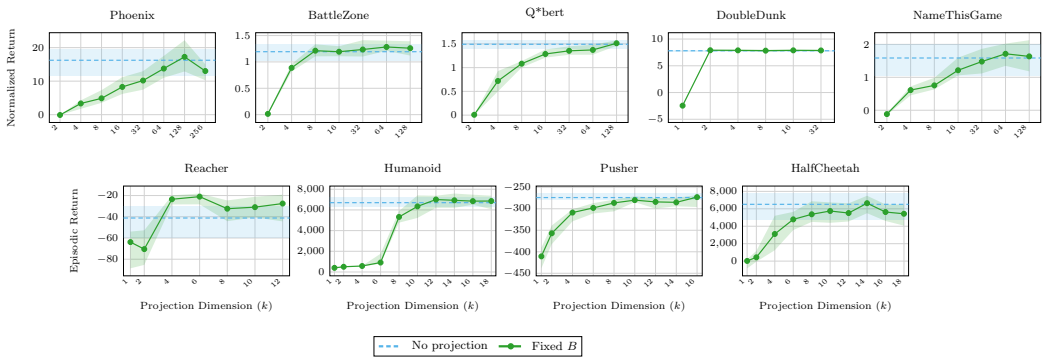


Figure 5: Final performance (IQM over seeds) as a function of bottleneck dimension k for Atari (top) and MuJoCo (bottom) tasks. For each task, performance recovers once k exceeds a small, task-dependent threshold and saturates thereafter, demonstrating that high-dimensional encoder features can be compressed into a very small subspace without loss of performance.

To further disentangle encoder capacity from representation dimensionality, we vary the width of the encoder's final layer on Humanoid while fixing the bottleneck dimension to $k = 8$. Across a wide range of encoder widths, learning curves and final performance remain similar, indicating that bottleneck dimension, rather than encoder width, is the primary factor governing expressivity in this setting (see Figure 8 in Appendix C).

5.3 Trainable Projections and Representation Geometry

We further compare fixed orthogonal projections against fully trainable end-to-end projection matrices. This comparison is intended to isolate the effect of the fixed projection assumed in Proposition 3.2, assess whether learning the bottleneck subspace provides additional benefits, and examine how this choice interacts with representation geometry. Figure 6 summarizes these results on two representative Atari (Phoenix) and MuJoCo (Humanoid) tasks, shown for both small and large bottleneck dimensions.

For Humanoid, allowing the projection to be trainable yields slightly higher returns at small bottleneck dimensions, but this advantage disappears as the bottleneck dimension increases. At larger k, fixed and trainable projections achieve similar performance. In contrast, for Phoenix, trainable projections exhibit unstable behavior at larger bottleneck dimensions and can collapse performance entirely, whereas fixed orthogonal projections remain reliable across both small and large k. Overall, this indicates that making the projection trainable might introduce additional sensitivity to the task and bottleneck size, while fixed orthogonal projections exhibit more consistent behavior across settings.

The corresponding effective-rank diagnostics in Figure 6 help contextualize these performance differences. Effective rank measures how many dimensions of the representation are actively used, and prior work has linked rank collapse to loss of plasticity and degraded learning dynamics in deep reinforcement learning (Kumar et al., 2021; Lyle et al., 2022; Klein et al., 2024). Fixed orthogonal projections consistently maintain high effective rank relative to their dimensionality, indicating uniform usage of the available subspace. In contrast, trainable projections and unconstrained baselines

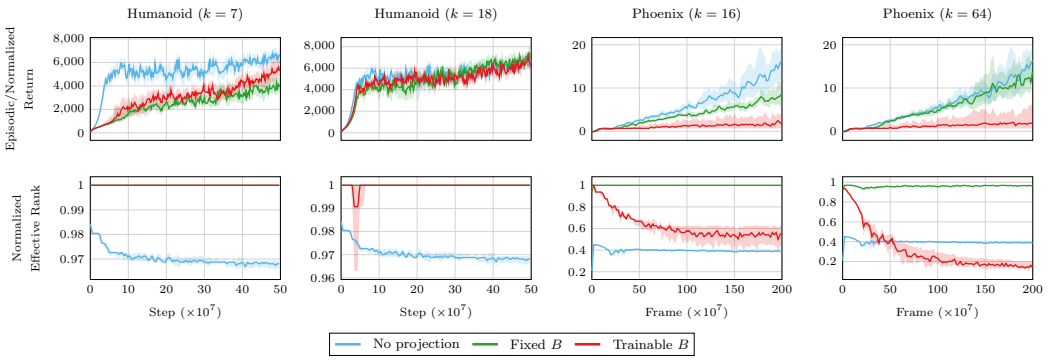


Figure 6: Performance (top row) and normalized mean effective rank (bottom row) for no projection, fixed orthogonal projection, and trainable projection on Humanoid and Phoenix at small and large bottleneck dimensions. Trainable projections can be beneficial or unstable depending on the environment, with performance degradation coinciding with severe rank collapse. Fixed orthogonal projections remain stable across tasks and bottleneck sizes.

exhibit more variable rank dynamics, with effective rank often substantially lower than the ambient feature dimension and a higher prevalence of weakly used or inactive directions.

Importantly, low effective rank does not, by itself, imply learning failure: the baseline agents learn successfully despite exhibiting comparatively low effective rank. However, when learning becomes unstable or collapses, this failure is typically accompanied by a pronounced drop in the effective rank. This relationship is most clearly visible for Phoenix at large bottleneck dimensions, where the collapse of trainable projections coincides with a sharp reduction in effective rank, while fixed orthogonal projections at the same dimensionality maintain both stable learning and high rank.

5.4 Multi-task Meta-World

We finally turn to the multi-task setting, where a single agent must solve multiple tasks simultaneously using shared parameters. Multi-task RL is often limited by negative transfer and representational interference, since tasks compete for shared capacity and updates from one task can degrade performance on others, and such settings are known to be particularly sensitive to how representational capacity is allocated across tasks (Yu et al., 2020; Yang et al., 2020). This issue has motivated a variety of methods in multi-task and continual learning that explicitly modify the geometry of the learning process, including PCGrad, which projects conflicting gradients (Yu et al., 2020), MOORE, which uses orthogonal expert representations to promote diversity and reduce interference (Hendawy et al., 2024), and low-rank orthogonal subspaces for reducing task interference in continual learning (Chaudhry et al., 2020).

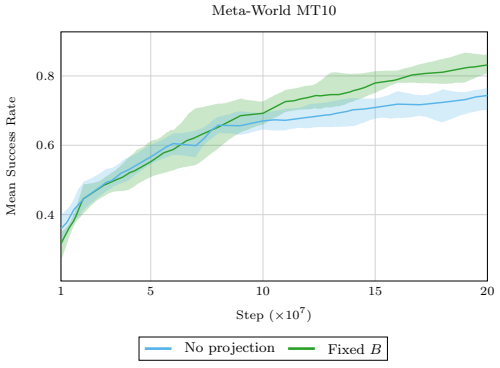


Figure 7: Meta-World MT10 performance of a baseline PPO agent and an agent equipped with a fixed bottleneck of dimension $k = 24$.

Figure 7 compares a standard PPO baseline against an agent equipped with a fixed orthogonal bottleneck of dimension $k = 24$ on Meta-World MT10. With a suitably chosen bottleneck dimension, the orthogonal bottleneck modestly improves performance relative to the no-projection baseline. This result indicates that constraining the agent to operate within a shared low-dimensional subspace does not hinder multi-task learning and can, in fact, even be beneficial, suggesting that appropriately

structured low-dimensional representations can be compatible with effective parameter sharing across tasks. We view this improvement over the unconstrained baseline as a secondary regularization effect, in line with a large body of prior work that has shown that structured constraints, such as sparsity, can improve learning by reducing interference or preserving plasticity (Graesser et al., 2022; Obando-Ceron et al., 2024; Todorov et al., 2025). These findings are also consistent with our broader finding that orthogonal bottlenecks encourage uniformly utilized shared representations, which may help mitigate degenerate representation collapse in settings with competing objectives.

6 Discussion and Conclusion

Our results provide empirical evidence that many reinforcement learning tasks admit low intrinsic representation dimensions, even when solved with highly overparameterized neural architectures. Across our benchmarks, the minimal sufficient bottleneck dimension varies substantially across environments, but performance is comparatively insensitive to encoder width in our Humanoid sweep at a fixed k. This pattern is consistent with recent conjectures that intrinsic manifold dimensionality is driven primarily by environment complexity rather than network size, though Tenedini et al. (2026) formulate this for policy manifolds rather than value representations. In small domains, such structures can be visualized explicitly as thin value manifolds embedded in an ambient feature space. While these manifolds exhibit smooth and coherent geometry, they remain largely uninterpretable: the learned axes do not correspond to semantically meaningful factors of the environment. An interesting direction for future work is to connect these findings to object-centric reinforcement learning, where representations are explicitly structured around entities, relations, and compositional structure (Greff et al., 2019; Locatello et al., 2020; Haramati et al., 2023). Such approaches may offer more interpretable low-dimensional representations than those learned implicitly by standard encoders. Studying how structured bottlenecks interact with object-centric architectures could help determine whether the low-dimensional manifolds observed here can be aligned with meaningful latent factors rather than remaining purely geometric.

From a theoretical perspective, the empirical success of small orthogonal bottlenecks provides indirect evidence that the linear realizability assumption commonly used in RL theory can be made to hold approximately by modern neural encoders. While this assumption cannot be verified directly, and alternative explanations cannot be ruled out, the fact that performance is preserved once the bottleneck dimension exceeds a small task-dependent threshold is consistent with the idea that neural networks can learn feature spaces in which the value or action-value function is well-approximated by a low-rank linear map. If such a structure were absent, the representational sufficiency guarantees implied by our analysis would fail, and small bottlenecks would necessarily degrade performance. That this does not occur across a wide range of tasks and algorithms suggests that deep RL representations may be more amenable to low-rank and linear analysis than is commonly assumed.

Overall, we show that the representations learned by modern deep RL agents can often be compressed into surprisingly low-dimensional subspaces without loss of performance. Orthogonal bottlenecks provide a lightweight and architecture-agnostic mechanism for exposing and exploiting this low-dimensional structure, offering a step toward more principled and geometry-aware representation design in reinforcement learning.

Acknowledgements

The authors thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Hábrók high-performance computing cluster. We also greatly appreciate everyone who was involved in providing feedback for the final version of the manuscript. Aleksandar is highly grateful for the financial support provided by the Department of Artificial Intelligence at the University of Groningen and his supervisor, Matthia Sabatelli, who allowed the execution and presentation of the current work to happen.

References

Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational Option Discovery Algorithms, July 2018. URL http://arxiv.org/abs/1807.10299. arXiv:1807.10299 [cs].

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29304–29320. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/f514cec81cb148559cf475e7426eed5e-Abstract.html.

Matthew Aitchison, Penny Sweetser, and Marcus Hutter. Atari-5: Distilling the Arcade Learning Environment down to Five Games, October 2022. URL http://arxiv.org/abs/2210.02019. arXiv:2210.02019 [cs].

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary Evolution Recurrent Neural Networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1120–1128. PMLR, June 2016. URL https://proceedings.mlr.press/v48/arjovsky16.html. ISSN: 1938-7228.

Isac Arnekvist, Danica Kragic, and Johannes A. Stork. VPE: Variational Policy Embedding for Transfer Reinforcement Learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 36–42, May 2019. DOI: 10.1109/ICRA.2019.8793556. URL https://ieeexplore.ieee.org/document/8793556. ISSN: 2577-087X.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research*, 47:253–279, June 2013. ISSN 1076-9757. DOI: 10.1613/jair.3912. URL https://www.jair.org/index.php/jair/article/view/10819.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives, April 2014. URL http://arxiv.org/abs/1206.5538. arXiv:1206.5538 [cs].

Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual Learning in Low-rank Orthogonal Subspaces. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9900–9911. Curran Associates, Inc., 2020. URL https://papers.neurips.cc/paper_files/paper/2020/hash/70d85f35a1fdc0ab701ff78779306407-Abstract.html.

Minmin Chen, Jeffrey Pennington, and Samuel Schoenholz. Dynamical Isometry and a Mean Field Theory of RNNs: Gating Enables Signal Propagation in Recurrent Neural Networks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 873–882. PMLR, July 2018. URL https://proceedings.mlr.press/v80/chen18i.html. ISSN: 2640-3498.

Simon S. Du, Sham M. Kakade, Ruosong Wang, and Lin F. Yang. Is a Good Representation Sufficient for Sample Efficient Reinforcement Learning?, February 2020. URL http://arxiv.org/abs/1910.03016. arXiv:1910.03016 [cs].

Ayoub Echchahed and Pablo Samuel Castro. A Survey of State Representation Learning for Deep Reinforcement Learning. *Transactions on Machine Learning Research*, March 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=gOk34vUHtz.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is All You Need: Learning Skills without a Reward Function, October 2018. URL http://arxiv.org/abs/1802.06070. arXiv:1802.06070 [cs].

Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, February 2016. ISSN 0894-0347, 1088-6834. DOI: 10.1090/jams/852. URL https://www.ams.org/jams/2016-29-04/S0894-0347-2016-00852-4/.

Kevin Frans, Jonathan Ho, Xi Chen, Pieter Abbeel, and John Schulman. Meta Learning Shared Hierarchies, October 2017. URL http://arxiv.org/abs/1710.09767. arXiv:1710.09767 [cs].

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, June 2021. URL http://arxiv.org/abs/2106.13281. arXiv:2106.13281 [cs].

Matteo Gallici, Mattie Fellows, Benjamin Ellis, Bartomeu Pou, Ivan Masmitja, Jakob Nicolaus Foerster, and Mario Martin. Simplifying Deep Temporal Difference Learning, April 2025. URL http://arxiv.org/abs/2407.04811. arXiv:2407.04811 [cs].

Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Factor Analysis, Probabilistic Principal Component Analysis, Variational Inference, and Variational Autoencoder: Tutorial and Survey, May 2022. URL http://arxiv.org/abs/2101.00734. arXiv:2101.00734 [stat].

Dar Gilboa, Bo Chang, Minmin Chen, Greg Yang, Samuel S. Schoenholz, Ed H. Chi, and Jeffrey Pennington. Dynamical Isometry and a Mean Field Theory of LSTMs and GRUs, May 2019. URL http://arxiv.org/abs/1901.08987. arXiv:1901.08987 [cs].

Y. Goldberg, A. Zakai, D. Kushnir, and Y. Ritov. Manifold Learning: The Price of Normalization, June 2008. URL http://arxiv.org/abs/0806.2646. arXiv:0806.2646 [stat].

Laura Graesser, Utku Evci, Erich Elsen, and Pablo Samuel Castro. The State of Sparse Training in Deep Reinforcement Learning, June 2022. URL http://arxiv.org/abs/2206.10369. arXiv:2206.10369 [cs].

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-Object Representation Learning with Iterative Variational Inference. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2424–2433. PMLR, May 2019. URL https://proceedings.mlr.press/v97/greff19a.html. ISSN: 2640-3498.

Dan Haramati, Tal Daniel, and Aviv Tamar. Entity-Centric Reinforcement Learning for Object Manipulation from Pixels. October 2023. URL https://openreview.net/forum?id=uDxeSZ1wdI.

Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. Learning an Embedding Space for Transferable Robot Skills. February 2018. URL https://openreview.net/forum?id=rk07ZXZRb.

Mikael Henaff, Arthur Szlam, and Yann LeCun. Recurrent Orthogonal Networks and Long-Memory Tasks. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 2034–2042. PMLR, June 2016. URL https://proceedings.mlr.press/v48/henaff16.html. ISSN: 1938-7228.

Ahmed Hendawy, Jan Peters, and Carlo D'Eramo. Multi-Task Reinforcement Learning with Mixture of Orthogonal Experts, May 2024. URL http://arxiv.org/abs/2311.11385. arXiv:2311.11385 [cs].

Wei Hu, Lechao Xiao, and Jeffrey Pennington. Provable Benefit of Orthogonal Initialization in Optimizing Deep Linear Networks. September 2019. URL https://openreview.net/forum?id=rkgqN1SYvr.

Lei Huang, Li Liu, Fan Zhu, Diwen Wan, Zehuan Yuan, Bo Li, and Ling Shao. Controllable Orthogonalization in Training DNNs. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6428–6437, Seattle, WA, USA, June 2020. IEEE. ISBN 978-1-7281-7168-5. DOI: 10.1109/CVPR42600.2020.00646. URL https://ieeexplore.ieee.org/document/9157676/.

Kui Jia, Shuai Li, Yuxin Wen, Tongliang Liu, and Dacheng Tao. Orthogonal Deep Neural Networks, October 2019. URL http://arxiv.org/abs/1905.05929. arXiv:1905.05929 [cs].

William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In Richard Beals, Anatole Beck, Alexandra Bellow, and Arshag Hajian (eds.), *Contemporary Mathematics*, volume 26, pp. 189–206. American Mathematical Society, Providence, Rhode Island, 1984. ISBN 978-0-8218-5030-5 978-0-8218-7611-4. DOI: 10.1090/conm/026/737400. URL http://www.ams.org/conm/026/.

Timo Klein, Lukas Miklautz, Kevin Sidak, Claudia Plant, and Sebastian Tschiatschek. Plasticity Loss in Deep Reinforcement Learning: A Survey, November 2024. URL http://arxiv.org/abs/2411.04832. arXiv:2411.04832 [cs].

Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit Under-Parameterization Inhibits Data-Efficient Deep Reinforcement Learning, October 2021. URL http://arxiv.org/abs/2010.14498. arXiv:2010.14498 [cs].

Michael Laskin, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel. Unsupervised Reinforcement Learning with Contrastive Intrinsic Control. *Advances in Neural Information Processing Systems*, 35:34478–34491, December 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/debf482a7dbdc401f9052dbe15702837-Abstract-Conference.html.

Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. Learning with Good Feature Representations in Bandits and in RL with a Generative Model. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 5662–5670. PMLR, November 2020. URL https://proceedings.mlr.press/v119/lattimore20a.html. ISSN: 2640-3498.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-Centric Learning with Slot Attention. In *Advances in Neural Information Processing Systems*, volume 33, pp. 11525–11538. Curran Associates, Inc., 2020. URL https://papers.neurips.cc/paper_files/paper/2020/hash/8511df98c02ab60aea1b2356c013bc0f-Abstract.html.

Chris Lu, Jakub Grudzien Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered Policy Optimisation, October 2022. URL http://arxiv.org/abs/2210.05639. arXiv:2210.05639 [cs].

Clare Lyle, Mark Rowland, and Will Dabney. Understanding and Preventing Capacity Loss in Reinforcement Learning, May 2022. URL http://arxiv.org/abs/2204.09560. arXiv:2204.09560 [cs].

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks, November 2023. URL http://arxiv.org/abs/2303.01486. arXiv:2303.01486 [cs].

Reginald McLean, Evangelos Chatzaroulas, Luc McCutcheon, Frank Röder, Tianhe Yu, Zhanpeng He, K. R. Zentner, Ryan Julian, J. K. Terry, Isaac Woungang, Nariman Farsad, and Pablo Samuel Castro. Meta-World+: An Improved, Standardized, RL Benchmark. July 2025. URL https://openreview.net/forum?id=eYZ9ebLIXo.

Marina Meilă and Hanyu Zhang. Manifold learning: what, how, and why, November 2023. URL http://arxiv.org/abs/2311.03757. arXiv:2311.03757 [stat].

Dmytro Mishkin and Jiri Matas. All you need is a good init, February 2016. URL http://arxiv.org/abs/1511.06422. arXiv:1511.06422 [cs].

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. DOI: 10.1038/nature14236. URL https://www.nature.com/articles/nature14236. Publisher: Nature Publishing Group.

Mirco Mutti, Stefano Del Col, and Marcello Restelli. Reward-Free Policy Space Compression for Reinforcement Learning. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pp. 3187–3203. PMLR, May 2022. URL https://proceedings.mlr.press/v151/mutti22a.html. ISSN: 2640-3498.

Hariharan Narayanan and Sanjoy Mitter. Sample Complexity of Testing the Manifold Hypothesis. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL https://papers.nips.cc/paper_files/paper/2010/hash/8a1e808b55fde9455cb3d8857ed88389-Abstract.html.

Johan Obando-Ceron, Aaron Courville, and Pablo Samuel Castro. In value-based deep reinforcement learning, a pruned network is a good network, June 2024. URL http://arxiv.org/abs/2402.12479. arXiv:2402.12479 [cs].

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding, January 2019. URL http://arxiv.org/abs/1807.03748. arXiv:1807.03748 [cs].

Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/d9fc0cdb67638d50f411432d0d41d0ba-Abstract.html.

Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. The emergence of spectral universality in deep networks. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, pp. 1924–1932. PMLR, March 2018. URL https://proceedings.mlr.press/v84/pennington18a.html. ISSN: 2640-3498.

Nemanja Rakicevic, Antoine Cully, and Petar Kormushev. Policy Manifold Search: Exploring the Manifold Hypothesis for Diversity-based Neuroevolution. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 901–909, June 2021. DOI: 10.1145/3449639.3459320. URL http://arxiv.org/abs/2104.13424. arXiv:2104.13424 [cs].

Krishan Rana, Ming Xu, Brendan Tidd, Michael Milford, and Niko Sünderhauf. Residual Skill Policies: Learning an Adaptable Skill-based Action Space for Reinforcement Learning for Robotics, November 2022. URL http://arxiv.org/abs/2211.02231. arXiv:2211.02231 [cs].

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, February 2014. URL http://arxiv.org/abs/1312.6120. arXiv:1312.6120 [cs].

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017. URL http://arxiv.org/abs/1707.06347. arXiv:1707.06347 [cs].

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The Dormant Neuron Phenomenon in Deep Reinforcement Learning, June 2023. URL http://arxiv.org/abs/2302.12902. arXiv:2302.12902 [cs].

Davide Tenedini, Riccardo Zamboni, Mirco Mutti, and Marcello Restelli. From parameters to behaviors: Unsupervised compression of the policy space. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=VqnBaeu43F.

Michael E. Tipping and Christopher M. Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 61(3):611–622, September 1999. ISSN 1369-7412. DOI: 10.1111/1467-9868.00196. URL https://doi.org/10.1111/1467-9868.00196.

Aleksandar Todorov, Juan Cardenas-Cartagena, Rafael F. Cunha, Marco Zullich, and Matthia Sabatelli. Sparsity-Driven Plasticity in Multi-Task Reinforcement Learning. *Transactions on Machine Learning Research*, May 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=9L4Z23EfE9.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, October 2012. DOI: 10.1109/IROS.2012.6386109. URL https://ieeexplore.ieee.org/document/6386109. ISSN: 2153-0866.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments, November 2025. URL http://arxiv.org/abs/2407.17032. arXiv:2407.17032 [cs].

Gellért Weisz, András György, and Csaba Szepesvári. Online RL in Linearly q^π-Realizable MDPs Is as Easy as in Linear MDPs If You Learn What to Ignore, December 2023. URL http://arxiv.org/abs/2310.07811. arXiv:2310.07811 [cs].

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington. Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5393–5402. PMLR, July 2018. URL https://proceedings.mlr.press/v80/xiao18a.html. ISSN: 2640-3498.

Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. A Mean Field Theory of Batch Normalization, March 2019. URL http://arxiv.org/abs/1902.08129. arXiv:1902.08129 [cs].

Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. Multi-Task Reinforcement Learning with Soft Modularization, December 2020. URL http://arxiv.org/abs/2003.13661. arXiv:2003.13661 [cs].

Kenny Young and Tian Tian. MinAtar: An Atari-Inspired Testbed for Thorough and Reproducible Reinforcement Learning Experiments, June 2019. URL http://arxiv.org/abs/1903.03176. arXiv:1903.03176 [cs].

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient Surgery for Multi-Task Learning, December 2020. URL http://arxiv.org/abs/2001.06782. arXiv:2001.06782 [cs].

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Avnish Narayan, Hayden Shively, Adithya Bellathur, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A Benchmark and

Evaluation for Multi-Task and Meta Reinforcement Learning, June 2021. URL http://arxiv.org/abs/1910.10897. arXiv:1910.10897 [cs].

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning Invariant Representations for Reinforcement Learning without Reconstruction, April 2021. URL http://arxiv.org/abs/2006.10742. arXiv:2006.10742 [cs].

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew Johnson, and Sergey Levine. SOLAR: Deep Structured Representations for Model-Based Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 7444–7453. PMLR, May 2019. URL https://proceedings.mlr.press/v97/zhang19m.html. ISSN: 2640-3498.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Implementation and Hyperparameters

In this appendix, we report the full hyperparameter configurations used in all experiments. Tables are grouped by benchmark family: Classic Control (CartPole-v1, Acrobot-v1), MinAtar, Atari, MuJoCo, and Meta-World. Unless otherwise stated, hyperparameters are shared across tasks within the same family and algorithm, and differences are explicitly noted in the table captions. The projection dimensions are chosen to be the smallest k that matches baseline performance for each benchmark family. Moreover, as stated in the main text, we report the interquartile mean (IQM) of returns with 95% stratified bootstrap confidence intervals, following the recommendations of Agarwal et al. (2021).

For Classic Control, we use the `purejaxrl` DQN implementation by Lu et al. (2022). Similarly, for Atari, we use the `purejaxql` PQN implementation by Gallici et al. (2025). For MinAtar (Young & Tian, 2019) and Brax MuJoCo (Freeman et al., 2021) we use PPO, and for Meta-World, we use the multi-task PPO implementation from McLean et al. (2025). Hyperparameters were chosen slightly differently for each environment:

- DQN in CartPole: the original implementation provided by `purejaxrl` provides an already-tuned algorithm.
- PPO in MinAtar: we tune the hyperparameters so that they reach (or exceed) the baseline PPO performance as in Young & Tian (2019) and Gallici et al. (2025).
- PQN in Atari: the implementation by Gallici et al. (2025) provides an already-tuned implementation. We follow the original PQN theory and architecture and apply LayerNorm after every layer. Accordingly, when inserting the orthogonal bottleneck, we apply LayerNorm after the projection to maintain consistent feature scaling.
- PPO in MuJoCo Brax: given that the network architecture with a shared encoder is non-standard for MuJoCo, we tune the algorithm until it reaches (or exceeds) the baseline Brax PPO performance as reported in Freeman et al. (2021).
- PPO in Meta-World: we use the standard Meta-World PPO hyperparameters, specified in McLean et al. (2025). We found that the baseline PPO performance degrades significantly otherwise.

While many definitions for the effective rank exist in the literature, we consider the one commonly used in RL (Kumar et al., 2021; Obando-Ceron et al., 2024; Todorov et al., 2025). Namely, given a batch of feature vectors collected during rollouts, we form a feature matrix $X \in \mathbb{R}^{N\times d}$ where each row is one feature vector. We center features across the batch, $\tilde{X} = X - \mu$, where μ denotes the batch mean activation. Let $\sigma_1 \geq \cdots \geq \sigma_l$ be the singular values of $\tilde{X}$ (with $l = \min(N, d)$). We define normalized singular values

$$p_i = \frac{\sigma_i}{\sum_{j=1}^{l} \sigma_j},$$

and the mean effective rank as the smallest k_{eff} such that the cumulative mass exceeds $1 - \delta$:

$$k_{\text{eff}} = \min\Big\{k \in \{1, \ldots, l\} : \sum_{i=1}^{k} p_i \geq 1 - \delta\Big\}.$$

We use the standard value $\delta = 0.01$ for all experiments and report the normalized effective rank $k_{\text{norm}} = \frac{k_{\text{eff}}}{k}$, so that $k_{\text{norm}} \in [0, 1]$ measures the fraction of available bottleneck dimensions that are effectively used. In the absence of a bottleneck, k corresponds to the layer width. The full code is available at https://github.com/atodorov284/ortho-bottlenecks.

Table 1: Classic Control (CartPole-v1, Acrobot-v1) DQN hyperparameters. The same configuration is used for both environments.

Hyperparameter	**Value**
Total timesteps	5×10^5
Training environments	10
Discount γ	0.99
Learning rate	2.5×10^{-4}
Linear LR decay	False
Replay buffer size	10,000
Batch size	128
Learning starts	10,000
Training interval	10
Target update interval	500
Polyak τ	1.0
ε-greedy start	1.0
ε-greedy final	0.05
ε anneal timesteps	2.5×10^5
Encoder type	Linear
Encoder feature dim (D)	128
Encoder last dim	128
Q-head hidden layers	0
Q-head feature dim	64
Activation	ReLU
Testing environments	128
Test rollout horizon	Episode length
Number of metric evaluations	100
Test ε	0.0

Table 2: MinAtar PPO hyperparameters and evaluation settings.

Hyperparameter	**Value**
Environment	Breakout-MinAtar
Total timesteps	1×10^7
Training environments	64
Rollout length (steps)	128
Update epochs	4
Minibatches per epoch	8
Discount factor γ	0.99
GAE λ	0.95
PPO clip ϵ	0.2
Entropy coefficient	0.01
Value coefficient	0.5
Max grad norm	0.5
Learning rate	5×10^{-3}
Learning rate annealing	True
Activation	ReLU
Seed	0
Encoder type	CNN
Encoder feature dim	256
Encoder last dim	256
Actor feature dim	64
Critic feature dim	64
Actor hidden layers	0
Critic hidden layers	0
Testing environments	128
Test rollout horizon	Episode length
Number of metric evaluations	100

Table 3: Atari (Battle Zone, Double Dunk, Name This Game, Phoenix, Q*bert) PQN hyperparameters and environment settings. Total timesteps correspond to 200M frames with frame skip 4.

Hyperparameter	**Value**
Total timesteps	5×10^7
Training environments	128
Steps per env per update	32
Epochs per update	2
Minibatches per epoch	32
Discount γ	0.99
λ (trace / advantage)	0.65
Learning rate	2.5×10^{-4}
Linear LR decay	False
Max grad norm	10
Normalization	LayerNorm
Encoder last dim	512
ε start	1.0
ε final	0.001
ε decay ratio	0.1
Episodic life	True
Reward clip	True
Sticky actions prob.	0.0
Frame skip	4
No-op max	30
Testing environments	8
Test ε	0.0

Table 4: Brax MuJoCo (PPO) hyperparameters for Reacher, Pusher, HalfCheetah, and Humanoid.

Hyperparameter	**Reacher**	**Pusher**	**HalfCheetah**	**Humanoid**
Total timesteps	5×10^7	5×10^7	5×10^7	5×10^7
Learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Learning rate annealing	True	True	True	True
Parallel environments	2048	2048	2048	2048
Unroll length (steps)	50	30	20	10
Update epochs	8	8	8	8
Num. minibatches	32	16	32	32
Discount γ	0.95	0.95	0.95	0.97
GAE λ	0.95	0.95	0.95	0.95
PPO clip ϵ	0.3	0.3	0.3	0.3
Entropy coefficient	1×10^{-3}	1×10^{-2}	1×10^{-3}	1×10^{-3}
Value loss coefficient	0.5	0.5	0.5	0.5
Max grad norm	0.5	0.5	0.5	0.5
Activation	Tanh	Tanh	Tanh	Tanh
Action repeat	4	1	1	1
Episode length	1000	1000	1000	1000
Reward scaling	5.0	5.0	1.0	0.1
Normalize observations	True	True	True	True
Encoder type	Linear	Linear	Linear	Linear
Encoder feature dim	256	256	256	256
Encoder last dim	256	256	256	256
Actor feature dim	128	128	128	128
Num. actor layers	2	2	2	2
Critic feature dim	128	128	128	128
Num. critic layers	2	2	2	2
Log-std init	0.0	0.0	0.0	0.0

Table 5: Meta-World MT10 training and architecture hyperparameters. The baseline agent uses the same hyperparameters but without a bottleneck.

Hyperparameter	**Value**
Terminate on success	False
Total environment steps	2×10^7
Rollout steps per epoch	10,000
Evaluation frequency	2000 steps
Discount factor γ	0.99
GAE λ	0.97
Number of epochs	16
Gradient steps per epoch	32
Normalize advantages	False
Baseline type	MLP
Policy network	$[400, 400, 400]$
Value network	$[400, 400, 400]$
Architecture	Vanilla MLP
Activation	`tanh`
Policy squashing	`squash_tanh=False`
Optimizer	Adam
Learning rate	3×10^{-4}
Projector network width	256
Projector network depth	3
Bottleneck dimension	24

B Expressivity Analysis

This appendix section provides a self-contained proof of Proposition 3.2, which formalizes two basic properties of fixed orthogonal bottlenecks under a linear realizability assumption.

We consider a discounted Markov decision process (MDP) $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel $\mathcal{P}(\cdot \mid s, a)$, reward function $\mathcal{R}(s, a)$, and discount factor $\gamma \in [0, 1)$. A policy π induces trajectories $(s_n, a_n, r_{n+1})_{n\geq 0}$ with $a_n \sim \pi(\cdot \mid s_n)$, $s_{n+1} \sim \mathcal{P}(\cdot \mid s_n, a_n)$, and $r_{n+1} = \mathcal{R}(s_n, a_n)$. The objective is to maximize the expected discounted return $J(\pi) = \mathbb{E}_\pi[\sum_{n\geq 0} \gamma^n r_{n+1}]$, which is commonly approached by estimating value functions. The state-value and action-value functions are

$$V^\pi(s) = \mathbb{E}_\pi\left[\sum_{n\geq 0} \gamma^n r_{n+1} \,\middle|\, s_0 = s\right], \qquad V^\star(s) = \sup_\pi V^\pi(s),$$

$$Q^\pi(s, a) = \mathbb{E}_\pi\left[\sum_{n\geq 0} \gamma^n r_{n+1} \,\middle|\, s_0 = s,\ a_0 = a\right], \qquad Q^\star(s, a) = \sup_\pi Q^\pi(s, a).$$

Under linear realizability, we first show that if $V^\star$ can be represented in a learned feature space by a matrix $\Theta^\star$ of rank r, then any orthogonal bottleneck dimension $k \geq r$ is representationally sufficient. In particular, inserting a fixed orthonormal projector $B^\top$ after the encoder does not reduce expressivity relative to the given features. We then prove a trainability equivalence: if $W_t \in \mathbb{R}^{D\times D}$ denotes the encoder's final layer parameters at iteration t, gradient descent on the projected parameterization induces the same gradient dynamics on the composite map $A_t = B^\top W_t$ as gradient descent on an explicit k-dimensional map C_t when initialized identically. This reduction allows standard results for deep linear networks and matrix factorization to be applied to the projected architecture. For completeness, we restate Assumption 3.1 and Proposition 3.2. We state the analysis in terms of $V^\star$, but the same definitions and arguments apply to $Q^\star$ by replacing s with (s, a) and using action-dependent features.

Assumption B.1 (Linear realizability)**.** *There exists a matrix $\Theta^\star \in \mathbb{R}^{m\times D}$ such that for all $s \in \mathcal{S}$,*

$$V^\star(s) = \Theta^\star \phi(s).$$

Proposition B.2. *Assume $V^\star$ is linearly realizable in feature space with rank $r = \operatorname{rank}(\Theta^\star)$, and let $H(h; \theta)$ be any head expressive enough to realize at least a linear layer. For any $k \geq r$ and orthonormal $B \in \mathbb{R}^{D\times k}$:*

1. ***Representational sufficiency.*** *There exist encoder parameters and head parameters $\theta^\star$ such that the network*
$$s \;\mapsto\; H\big(B^\top z(s); \theta^\star\big)$$
exactly realizes $V^\star(s)$ for all $s \in \mathcal{S}$.

2. ***Trainability:*** *Let $W \in \mathbb{R}^{D\times D}$ be the encoder's final layer and $A_t = B^\top W_t$ the composite feature-to-bottleneck map. Training (θ, W) by gradient descent on loss $\mathcal{L}$ evolves A_t identically to training the direct parameterization $h = C\phi(s)$ on (θ, C), given $C_0 = A_0$.*

Proof. For general notation, fix a feature map $\phi : \mathcal{S} \to \mathbb{R}^D$ as in Assumption 3.1. We focus on the last linear layer of the encoder

$$z(s) = W\,\phi(s) \in \mathbb{R}^D, \qquad W \in \mathbb{R}^{D\times D}.$$

The fixed orthogonal bottleneck forms

$$h(s) = B^\top z(s) = B^\top W\,\phi(s) \in \mathbb{R}^k.$$

Define the composite feature-to-bottleneck map $A = B^\top W \in \mathbb{R}^{k\times D}$, so that throughout, $h(s) = A\,\phi(s)$. The network output is

$$\widehat{V}(s) \;=\; H\big(h(s);\theta\big) \;=\; H\big(A\phi(s);\theta\big).$$

1. Representational sufficiency

We will explicitly construct parameters $(W^\star, \theta^\star)$ such that for all s,

$$H\big(B^\top W^\star \phi(s);\theta^\star\big) = \Theta^\star\phi(s) = V^\star(s).$$

The key point is that $\Theta^\star$ has rank r and we assume $k \geq r$.

Since $\Theta^\star \in \mathbb{R}^{m\times D}$ has rank r, it admits a singular value decomposition

$$\Theta^\star = U_r\Sigma_r V_r^\top,$$

where

- $U_r \in \mathbb{R}^{m\times r}$ has orthonormal columns ($U_r^\top U_r = I_r$),
- $\Sigma_r \in \mathbb{R}^{r\times r}$ is diagonal with strictly positive singular values,
- $V_r \in \mathbb{R}^{D\times r}$ has orthonormal columns ($V_r^\top V_r = I_r$).

We can further factor Θ^* through an r-dimensional bottleneck by $\Theta^* = LR$, where

$$L = U_r\Sigma_r^{1/2} \in \mathbb{R}^{m\times r}, \qquad R = \Sigma_r^{1/2}V_r^\top \in \mathbb{R}^{r\times D}.$$

Then, indeed,

$$LR = U_r\Sigma_r^{1/2}\Sigma_r^{1/2}V_r^\top = U_r\Sigma_r V_r^\top = \Theta^\star.$$

Since the bottleneck dimension k satisfies $k \geq r$, we can embed this r-dimensional factorization into a k-dimensional one by padding with zeros. Define

$$U^\star = \begin{bmatrix} L & 0_{m\times(k-r)} \end{bmatrix} \in \mathbb{R}^{m\times k}, \qquad A^\star = \begin{bmatrix} R \\ 0_{(k-r)\times D} \end{bmatrix} \in \mathbb{R}^{k\times D}.$$

Then

$$U^\star A^\star = \begin{bmatrix} L & 0 \end{bmatrix}\begin{bmatrix} R \\ 0 \end{bmatrix} = LR = \Theta^\star.$$

Thus $\Theta^\star$ factors through a k-dimensional bottleneck.

We now must realize the composite map $A^\star = B^\top W^\star$ for some $W^\star \in \mathbb{R}^{D\times D}$. Since $B \in \mathbb{R}^{D\times k}$ has orthonormal columns, i.e. $B^\top B = I_k$, a canonical choice is

$$W^\star = BA^\star \in \mathbb{R}^{D\times D}.$$

Then

$$B^\top W^\star = B^\top(BA^\star) = (B^\top B)A^\star = I_k A^\star = A^\star.$$

By assumption, the head $H(h;\theta)$ contains at least a linear layer. Concretely, this means there exists a subset of parameters inside θ that can implement an affine map

$$h \mapsto Uh + b$$

for arbitrary $U \in \mathbb{R}^{m\times k}$ and $b \in \mathbb{R}^m$, possibly followed and/or preceded by additional transformations. We will choose $\theta^\star$ so that overall the head implements a pure linear map $h \mapsto U^\star h$. This is always possible for common heads used in RL (e.g., an MLP head) by setting all subsequent layers to

identity (or appropriate weights) and biases to zero; equivalently, one may take the output layer to be linear with weight $U^\star$ and zero bias, and set any intermediate layers to implement the identity on $\mathbb{R}^k$.

Thus we choose $\theta^\star$ such that

$$H(h;\theta^\star) = U^\star h \qquad \forall\, h \in \mathbb{R}^k.$$

With these choices, for any $s \in \mathcal{S}$ we have

$$H\big(B^\top z(s);\theta^\star\big) = H\big(B^\top W^\star \phi(s);\theta^\star\big) = U^\star(B^\top W^\star)\phi(s) = U^\star A^\star \phi(s) = \Theta^\star \phi(s) = V^\star(s).$$

This proves representational sufficiency.

2. Trainability

We now prove that training (θ, W) with the orthogonal bottleneck induces the same gradient descent dynamics on the composite map $A_t = B^\top W_t$ as training a direct bottleneck map C_t in the parameterization $h = C\phi(s)$, provided $C_0 = A_0$.

Consider an arbitrary (differentiable) training objective $\mathcal{L}$ computed from the head output. For example, $\mathcal{L}$ can be an empirical risk over a dataset or any differentiable surrogate used within an RL update (e.g., value regression loss, PPO value loss, etc.). The only property we use is that $\mathcal{L}$ depends on W only through the bottleneck features $h(s) = B^\top W\phi(s)$ (with B fixed).

Define the projected parameterization

$$\widehat{V}_{\text{proj}}(s;\theta,W) = H\big(B^\top W\phi(s);\theta\big) = H\big(A\phi(s);\theta\big), \qquad A = B^\top W,$$

with a corresponding loss $\mathcal{L}_{\text{proj}}(\theta, W)$. Similarly, define the direct parameterization

$$\widehat{V}_{\text{dir}}(s;\theta,C) = H\big(C\phi(s);\theta\big), \qquad C \in \mathbb{R}^{k\times D}.$$

with a corresponding loss $\mathcal{L}_{\text{dir}}(\theta, C)$.

By construction, if we identify $C = A = B^\top W$, then the two models produce identical outputs for all s and hence identical loss values with

$$\mathcal{L}_{\text{proj}}(\theta, W) = \mathcal{L}_{\text{dir}}(\theta, A) \quad \text{with } A = B^\top W.$$

For convenience, define the induced loss

$$\widetilde{\mathcal{L}}(\theta, A) = \mathcal{L}_{\text{dir}}(\theta, A),$$

so that

$$\mathcal{L}_{\text{proj}}(\theta, W) = \widetilde{\mathcal{L}}(\theta, B^\top W).$$

Fix θ and consider $W \mapsto \mathcal{L}_{\text{proj}}(\theta, W) = \widetilde{\mathcal{L}}(\theta, B^\top W)$. Let $dW \in \mathbb{R}^{D\times D}$ be an arbitrary perturbation. The corresponding perturbation of $A = B^\top W$ is

$$dA = B^\top dW.$$

Using the chain rule in differential form,

$$d\mathcal{L}_{\text{proj}} = d\widetilde{\mathcal{L}} = \langle \nabla_A \widetilde{\mathcal{L}}(\theta, A),\, dA\rangle_F.$$

Substituting $dA = B^\top dW$ results in

$$\begin{aligned} d\mathcal{L}_{\text{proj}} &= \langle \nabla_A \widetilde{\mathcal{L}}(\theta, A),\, B^\top dW\rangle_F \\ &= \mathrm{Tr}\Big((\nabla_A \widetilde{\mathcal{L}})^\top B^\top dW\Big) \\ &= \mathrm{Tr}\Big(B(\nabla_A \widetilde{\mathcal{L}})^\top dW\Big) \;=\; \mathrm{Tr}\Big((B\nabla_A \widetilde{\mathcal{L}})^\top dW\Big) \\ &= \langle B\nabla_A \widetilde{\mathcal{L}}(\theta, A),\, dW\rangle_F, \end{aligned}$$

where in the intermediate step we used the cyclicity of the trace. Since this holds for all perturbations dW, the gradient is

$$\nabla_W \mathcal{L}_{\text{proj}}(\theta, W) = B\, \nabla_A \widetilde{\mathcal{L}}(\theta, A), \qquad A = B^\top W.$$

Now, consider gradient descent updates on (θ, W) with step size $\eta > 0$:

$$\theta_{t+1} = \theta_t - \eta\, \nabla_\theta \mathcal{L}_{\text{proj}}(\theta_t, W_t), \qquad W_{t+1} = W_t - \eta\, \nabla_W \mathcal{L}_{\text{proj}}(\theta_t, W_t).$$

Define $A_t = B^\top W_t$. Then

$$\begin{aligned} A_{t+1} &= B^\top W_{t+1} = B^\top (W_t - \eta\, \nabla_W \mathcal{L}_{\text{proj}}(\theta_t, W_t)) \\ &= B^\top W_t - \eta\, B^\top \nabla_W \mathcal{L}_{\text{proj}}(\theta_t, W_t) \\ &= A_t - \eta\, B^\top \left(B\, \nabla_A \widetilde{\mathcal{L}}(\theta_t, A_t) \right) \\ &= A_t - \eta\, (B^\top B)\, \nabla_A \widetilde{\mathcal{L}}(\theta_t, A_t). \end{aligned}$$

By orthonormality, we have $B^\top B = I_k$, so we obtain

$$A_{t+1} = A_t - \eta\, \nabla_A \widetilde{\mathcal{L}}(\theta_t, A_t).$$

But this is exactly the gradient descent update that would be obtained by directly training the parameter $C \in \mathbb{R}^{k \times D}$ in the direct parameterization with the same step size:

$$C_{t+1} = C_t - \eta\, \nabla_C \mathcal{L}_{\text{dir}}(\theta_t, C_t) = C_t - \eta\, \nabla_A \widetilde{\mathcal{L}}(\theta_t, C_t).$$

Thus, if $C_0 = A_0$, then by induction $C_t = A_t$ for all t.

Finally, we verify that the head-parameter updates match under the identification $C_t = A_t$. Since the two losses satisfy

$$\mathcal{L}_{\text{proj}}(\theta, W) = \widetilde{\mathcal{L}}(\theta, B^\top W), \qquad \mathcal{L}_{\text{dir}}(\theta, C) = \widetilde{\mathcal{L}}(\theta, C),$$

we have, for fixed $A = B^\top W$ and $C = A$,

$$\nabla_\theta \mathcal{L}_{\text{proj}}(\theta, W) = \nabla_\theta \widetilde{\mathcal{L}}(\theta, A) = \nabla_\theta \mathcal{L}_{\text{dir}}(\theta, A).$$

Hence the θ-iterates coincide as well when initialized identically.

We have shown that when $k \geq r$, there exist parameters realizing $V^\star$ exactly through the fixed orthogonal bottleneck, and that gradient descent on (θ, W) induces the same dynamics on the composite map $A_t = B^\top W_t$ (and on θ_t) as direct training of (θ, C) in the parameterization $h = C\phi(s)$ with $C_0 = A_0$. This completes the proof.

□

C Encoder Width Sweep

To disentangle encoder capacity from representation dimensionality, we vary the width of the encoder's final layer on Humanoid while fixing the bottleneck dimension to $k = 8$. Figure 8 reports the corresponding learning curves. Across a wide range of widths D, curves overlap closely, and final performance is similar, indicating weak sensitivity to encoder width once the bottleneck dimension is fixed. That said, very large widths (e.g., $D = 1024$) can be slightly worse than moderate widths, suggesting diminishing returns, and occasional mild degradation from increasing encoder capacity beyond what is needed in this setting.

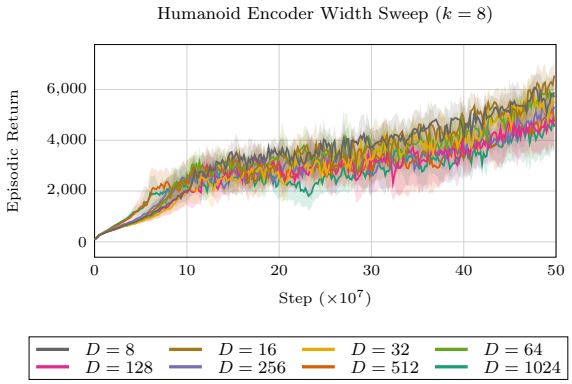


Figure 8: Humanoid encoder-width D sweep with fixed bottleneck dimension $k = 8$. Curves largely overlap across encoder widths, indicating that performance is only weakly sensitive to encoder width once k is fixed; very large widths (e.g., $D = 1024$) can exhibit slightly lower returns.

D Orthogonal Initialization Variants

To verify that our results are not sensitive to the orthogonalization method used to initialize the fixed projection matrix B, we compare three standard procedures on Humanoid while fixing the bottleneck dimension to $k = 18$, which is at or above the recovery threshold for this task. We consider QR factorization, SVD (using the left singular vectors), and a polar-style normalization, in which we sample a Gaussian matrix X and normalize it by the inverse square root of its Gram matrix, setting $B = X(X^\top X)^{-1/2}$, where $(X^\top X)^{-1/2}$ is computed via an eigen-decomposition with a small floor of 10^{-6} for numerical stability.

Figure 9 shows the corresponding learning curves. Across all three initializations, curves overlap closely, and final performance is similar, indicating that the findings are insensitive to the specific method used to construct an orthonormal basis for the bottleneck subspace in this setting.

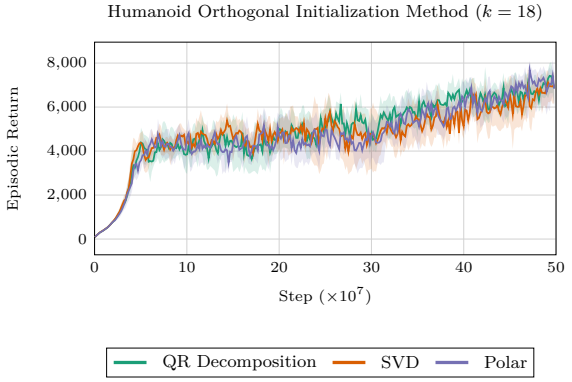


Figure 9: Humanoid learning curves with a fixed bottleneck dimension $k = 18$, comparing three orthogonal initialization methods for the fixed projection matrix B: QR decomposition, SVD, and a polar-style normalization. Performance is similar across initialization methods, suggesting that results are not driven by the particular orthogonalization procedure.

E Full Learning Curves

This appendix provides full learning curves for Classic Control and MinAtar, complementing the main-text results that focus on bottleneck sweeps and aggregate performance. For Atari and Brax

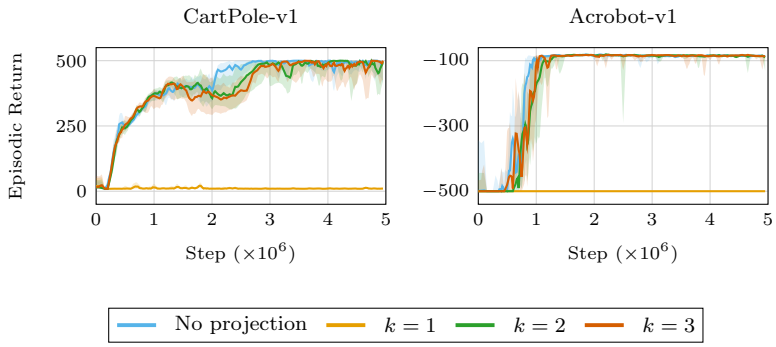


Figure 10: Classic Control learning curves for CartPole-v1 and Acrobot-v1 comparing the unconstrained DQN baseline to fixed orthogonal bottlenecks with varying dimension k. In both tasks, performance recovers once k reaches a small threshold, and further increases in k provide little additional benefit.

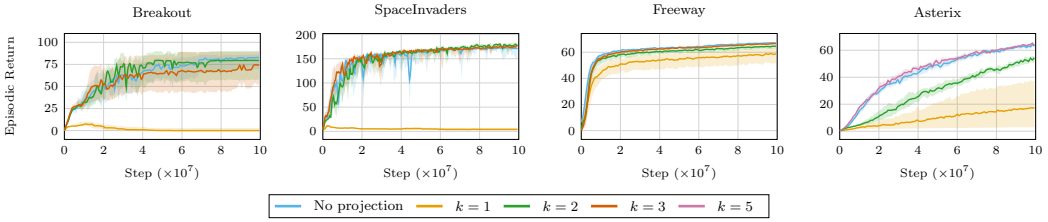


Figure 11: MinAtar learning curves comparing the unconstrained PPO baseline to fixed orthogonal bottlenecks across selected values of k. Similar to Classic Control, performance typically recovers once k exceeds a small task-dependent threshold, consistent with the main-text bottleneck sweeps. For Breakout, SpaceInvaders, and Freeway, $k = 1$, $k = 2$, and $k = 3$ are displayed. For Asterix, $k = 1$, $k = 3$, and $k = 5$ are displayed.

MuJoCo, plotting learning curves for every bottleneck dimension k would be visually cluttered and redundant with the summary in Figure 5. Instead, for each environment, we report representative learning curves for the unconstrained baseline, a bottleneck dimension below the recovery threshold, and a bottleneck dimension at or above the recovery threshold. This presentation makes the qualitative transition from failure to baseline-level learning explicit while keeping the figures readable.

Figures 10 and 11 show full learning curves for Classic Control and MinAtar across several bottleneck dimensions. Figures 12 and 13 show representative curves for Atari and Brax MuJoCo, comparing the no-projection baseline to a suboptimal and a recovering bottleneck dimension, with the corresponding k values listed in Table 6. Shaded regions indicate 95% bootstrapped confidence intervals across the 10 seeds.

In addition to learning performance, for each learning-curve figure in this appendix, we include a matched figure that plots the normalized mean effective rank of the value representations over training for the same set of runs and bottleneck dimensions. This makes it possible to compare performance recovery and the evolution of representation rank side-by-side across benchmarks. Figure 14 provides the effective rank curves for Classic Control, Figure 15 for MinAtar, Figure 16 for Atari, Figure 17 for Brax MuJoCo, and Figure 18 shows a paired two-panel figure with the main-text learning curve and the corresponding effective rank curve for the same runs.

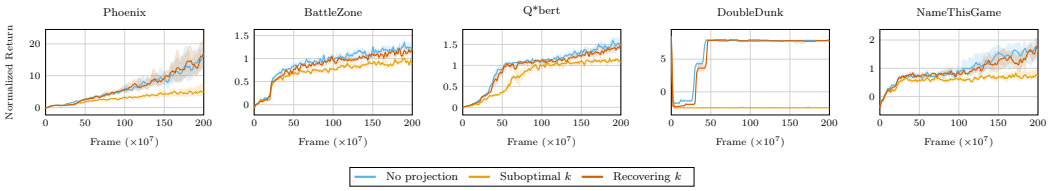


Figure 12: Representative Atari learning curves for each game, showing the unconstrained PQN baseline, a suboptimal bottleneck dimension (Suboptimal k), and a bottleneck dimension at/above the recovery threshold (Recovering k). Bottleneck dimensions differ by game; the specific k values used are listed in Table 6.

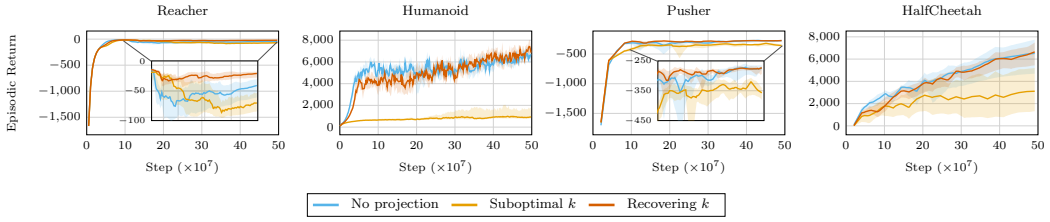


Figure 13: Representative Brax MuJoCo learning curves for each task, showing the unconstrained PPO baseline, a suboptimal bottleneck dimension (Suboptimal k), and a bottleneck dimension at/above the recovery threshold (Recovering k). Bottleneck dimensions differ by task; the specific k values used are listed in Table 6.

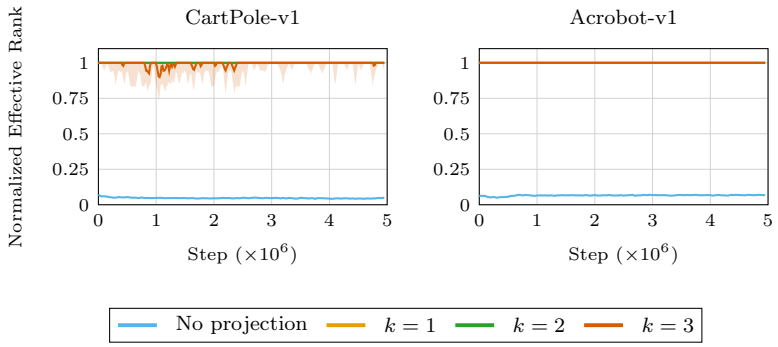


Figure 14: Classic Control normalized mean effective rank curves for CartPole-v1 and Acrobot-v1 for the same runs shown in Figure 10.

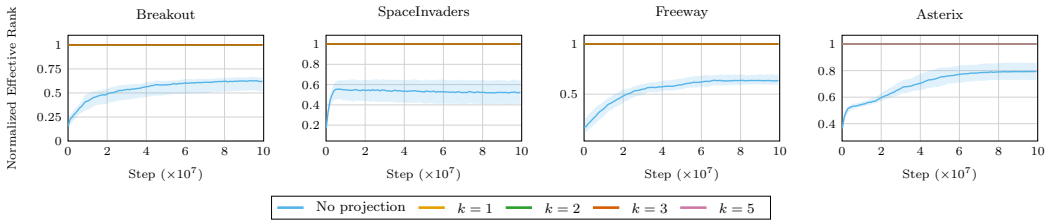


Figure 15: MinAtar normalized mean effective rank curves for the same runs shown in Figure 11.

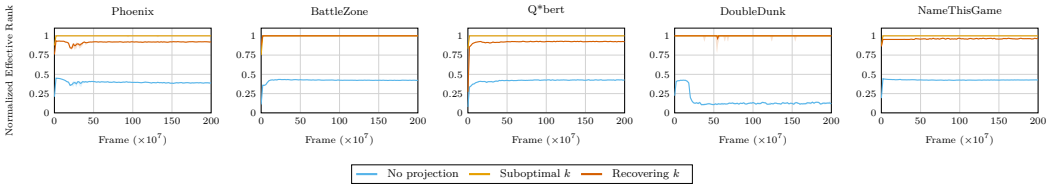


Figure 16: Atari normalized mean effective rank curves for the same runs shown in Figure 12.

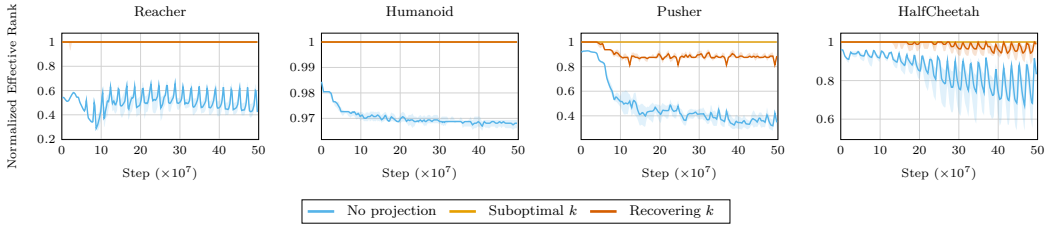
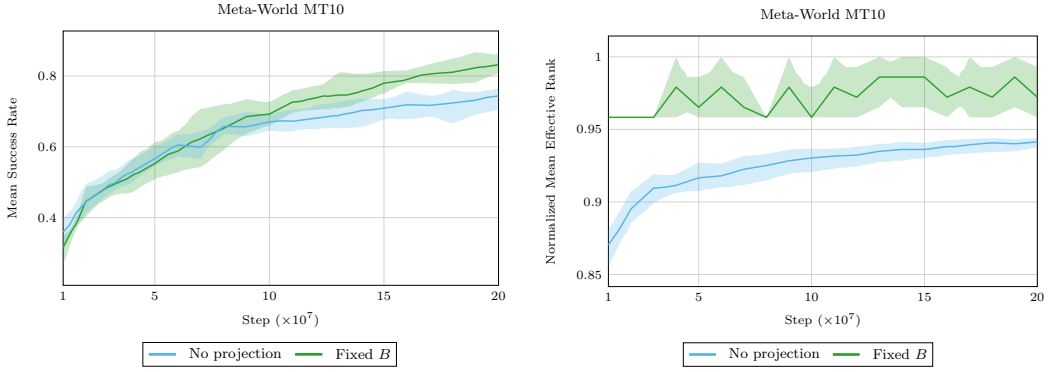


Figure 17: Brax MuJoCo normalized mean effective rank curves for the same runs shown in Figure 13.



(a) Learning curve.

(b) Normalized mean effective rank.

Figure 18: Meta-World MT10 performance and normalized mean effective rank for the same runs. *(Left)*: learning curves for the PPO baseline and the fixed orthogonal bottleneck at $k = 24$ (as in Figure 7 in the main text). *(Right)*: the corresponding normalized mean effective rank of the network representations over training.

Table 6: Bottleneck dimensions used for the representative learning curves in Figure 12 and Figure 13. For each environment, we plot the baseline with no projection, a suboptimal bottleneck dimension below the recovery threshold, and a recovering bottleneck dimension at/above the recovery threshold.

Benchmark	**Environment**	**Suboptimal** k	**Recovering** k
Atari-5	Phoenix	8	128
	Battle Zone	4	8
	Q*bert	8	128
	Double Dunk	1	2
	Name This Game	8	64
Brax MuJoCo	Reacher	2	6
	Humanoid	6	18
	Pusher	2	16
	HalfCheetah	4	14