REWARD-AGNOSTIC DATA REUSE FOR SHARED-STRUCTURE POLICY EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

How should stored transitions be prioritized when the same experience supports predictions of multiple signals? We study reward-agnostic replay for linear temporal-difference (TD) policy evaluation with a replay buffer. We exploit the decomposition of the expected TD update into a matrix shared across reward functions and a reward-dependent vector. By formulating estimation of the shared matrix as an approximate matrix multiplication problem, we derive a replay rule that prioritizes transitions using the product of the current-state feature norm and the norm of its discounted one-step feature difference. For the corresponding unbiased matrix estimator, this sampling rule minimizes expected squared Frobenius error within the estimator family we consider. The priorities are reward-independent, so they can be reused across multiple reward functions, including reward functions introduced later, without any reward-specific recomputation. In multi-reward evaluation with the reward-dependent vectors computed from the full buffer, a single norm-sampled matrix estimate improves value prediction over uniform replay and eventually outperforms reward-dependent PER as the number of tasks grows. The same priorities also transfer to held-out tasks and to tasks introduced later. Finally, we apply the same sampling rule to DQN on four MinAtar games, where we observe game-dependent gains in return and lower auxiliary multi-signal Bellman residuals in several settings. Together, these results help characterize when shared, reward-independent structure is useful for replay prioritization.

1 INTRODUCTION

In this paper, we study reward-agnostic data reuse for linear temporal-difference (TD) policy evaluation with a replay buffer (Sutton, 1988; Sutton & Barto, 2018; Tsitsiklis & Van Roy, 1997). Policy evaluation aims to estimate the expected accumulation of future rewards, or more generally other signals over time, from a given state or state-action pair under a fixed policy. A common way to reuse past experience is a replay buffer (Lin, 1992; Mnih et al., 2015), which stores past experiences so they can be sampled later for additional updates. Uniform replay samples all stored experiences with equal probability, while prioritized experience replay (PER) samples experiences more frequently when they appear more informative for the current prediction problem (Schaul et al., 2016). Once replay is introduced, a natural question arises: under a limited update budget, which past experiences should be replayed to make these predictions as accurate as possible?

This question becomes more subtle when the same stored experience is used to predict multiple signals. An agent may need to predict several long-term signals under the same policy, such as the accumulated rewards for different tasks, costs associated with different events, or other measurements accumulated over time (Sutton et al., 2011; Modayil et al., 2014; White, 2015). Different predictions may depend on different reward signals, but they can share the same environment dynamics and feature representation. A replay rule based on the reward or prediction error of one signal may therefore work well for that prediction, but may not prioritize the information that is shared across predictions. This motivates the central question of this paper: *can replay be guided by information that is shared across different predictions?*

The key observation in this paper is that, in linear TD policy evaluation, the expected update separates into a reward-dependent vector and a matrix that does not depend on the reward (Bradtke & Barto, 1996; Tsitsiklis & Van Roy, 1997). The vector depends on the particular reward or long-term

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.

signal being predicted, while the matrix depends on the features of the current and next states and on the environment dynamics under the fixed policy. When many predictions share the same policy, dynamics, and features, the matrix is shared across them. If all stored transitions could be processed whenever policy evaluation is performed, the natural choice would be to use the entire replay buffer. The sampling problem studied here arises when the update budget allows only a limited number of stored transitions to be replayed. The goal is therefore not to improve over using the full buffer, but to use a limited number of sampled transitions more effectively than uniform replay when estimating the component shared across predictions.

To answer this question, we propose a reward-agnostic replay rule. Instead of prioritizing transitions using the prediction error for a particular signal, the proposed rule uses only the feature representation of the current state and the discounted difference between the current- and next-state features. Intuitively, transitions with large feature norms and large discounted differences between current and next-state features contribute more to the shared component of the TD update. We formalize this intuition by formulating estimation of the shared TD matrix as an approximate matrix multiplication (AMM) problem, which gives the *norm-based* sampler (Drineas et al., 2006a). For the corresponding unbiased matrix estimator, this sampling rule minimizes expected squared Frobenius error within this estimator family. The priorities are reward-independent, so they can be reused across multiple reward functions, including ones introduced later, without reward-specific recomputation.

Improving the shared matrix estimate does not always improve value prediction. For single-signal prediction, errors in the reward-dependent vector can instead dominate, and reward-aware replay can be more suitable when this task-specific component is the main source of error. Norm-based replay is most useful when the shared matrix is an important source of error and can be reused across many predictions. A second factor is how unevenly individual transitions contribute to this matrix: when the norm-based scores vary substantially across transitions, targeted sampling has more room to improve over uniform replay. We use the heterogeneity of these scores as a diagnostic for when norm-based sampling is likely to improve matrix estimation. Finally, we test whether the same ideas extend beyond fixed linear features to DQN, where the feature representation used for prioritization is learned during training.

The contributions of the paper are as follows:

1. We show that estimating the shared TD matrix can be formulated as an approximate matrix multiplication problem, yielding a reward-agnostic norm-based sampling rule with minimum expected squared Frobenius error within the family of unbiased matrix estimators we consider.
2. We empirically characterize when this rule is useful. Across linear and replay-buffer settings, gains in matrix estimation increase with the heterogeneity of the norm-based scores, giving a simple diagnostic for when targeted sampling can help. When many reward-dependent vectors exist and are known, one norm-sampled matrix estimate can be reused across tasks: it improves prediction over uniform replay, and eventually outperforms reward-dependent PER as the number of tasks grows. Its reward-independent priorities also transfer to held-out and newly introduced tasks, while reward-dependent replay can perform better for single-reward learning.
3. We apply the same priority to DQN on four MinAtar games, using features learned by the network during training. We observe game-dependent gains in return and lower auxiliary multi-signal Bellman residuals in several settings, with the return pattern qualitatively tracking the heterogeneity of the feature-based priority scores.

The rest of the paper is organized as follows: Section 2 introduces linear TD, general value functions (GVFs), and replay. Section 3 presents the sampling rule, with implementation details in Appendix D. Section 4 evaluates when norm-based replay improves estimation of the shared matrix and value prediction. Section 5 evaluates the same sampler with learned features in DQN, followed by the conclusion in Section 6.

2 Background and Notation

Linear TD Policy Evaluation. We consider a discounted Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$ (Puterman, 1994), where $\mathcal{S}$ and $\mathcal{A}$ are the state and action sets, p is the transition kernel, r is the expected one-step reward, and $\gamma \in [0, 1)$ is the discount factor. At time

$t \in \{0, 1, 2, \ldots\}$, the agent occupies a state $S_t \in \mathcal{S}$, selects an action $A_t \in \mathcal{A}$, receives a reward $R_t \in \mathbb{R}$, and transitions to S_{t+1}. The expected reward is $r(s, a) \coloneqq \mathbb{E}[R_t \mid S_t{=}s, A_t{=}a]$, and the transition kernel is $p(s, a, s') \coloneqq \Pr(S_{t+1}{=}s' \mid S_t{=}s, A_t{=}a)$. A stationary policy π assigns $\pi(s, a) \coloneqq \Pr(A_t{=}a \mid S_t{=}s)$. Let $G_t \coloneqq \sum_{k=0}^{\infty} \gamma^k R_{t+k}$ denote the discounted return. Policy evaluation aims to estimate the value function $v^\pi(s) \coloneqq \mathbb{E}[G_t \mid S_t{=}s; \pi]$ (Sutton & Barto, 2018).

When the state space is large, storing and estimating a separate value for every state can be impractical. We therefore represent each state s by a feature vector $\phi(s) \in \mathbb{R}^d$ and approximate its value using a linear function of these features. The resulting approximation is $v_{\boldsymbol{\theta}}(s) = \boldsymbol{\theta}^\top \phi(s)$, where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the parameter vector we seek to estimate. We write $\boldsymbol{\phi}_t \coloneqq \phi(S_t)$. One linear TD update is $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \delta_t \boldsymbol{\phi}_t$, where $\alpha > 0$ is the step size and $\delta_t = R_t + \gamma \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t$ is the TD error. Holding $\boldsymbol{\theta}_t$ fixed and taking the expectation under the stationary state-transition distribution induced by π gives the expected update $\mathbb{E}[\boldsymbol{\theta}_{t+1} \mid \boldsymbol{\theta}_t] = \boldsymbol{\theta}_t + \alpha(\boldsymbol{b} - \mathbf{A}\boldsymbol{\theta}_t)$, where

$$\mathbf{A} \coloneqq \mathbb{E}\big[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma \boldsymbol{\phi}_{t+1})^\top\big] \in \mathbb{R}^{d \times d}, \qquad \boldsymbol{b} \coloneqq \mathbb{E}[R_t \boldsymbol{\phi}_t] \in \mathbb{R}^d. \tag{1}$$

The TD fixed point (Tsitsiklis & Van Roy, 1997) solves $\boldsymbol{b} = \mathbf{A}\boldsymbol{\theta}_{\text{TD}}$. When $\mathbf{A}$ is nonsingular,

$$\boldsymbol{\theta}_{\text{TD}} = \mathbf{A}^{-1}\boldsymbol{b}. \tag{2}$$

Estimating $\mathbf{A}$ and $\boldsymbol{b}$ from transitions and solving this system is least-squares TD (LSTD) (Bradtke & Barto, 1996; Boyan, 2002). To reduce sensitivity to ill-conditioning, we consider the regularized TD solution

$$\boldsymbol{\theta}_{\text{TD}}^{\lambda} = (\mathbf{A} + \lambda \mathbf{I})^{-1}\boldsymbol{b}, \tag{3}$$

with regularization parameter $\lambda > 0$ (Kolter & Ng, 2009; Farahmand et al., 2008). For a finite state set with $n{=}|\mathcal{S}|$, let Φ denote the feature matrix whose i^{th} row is $\phi(s_i)^\top$. The transition matrix under policy π has entries $\mathbf{P}^\pi(s, s') \coloneqq \sum_a \pi(s, a) p(s, a, s')$ and the corresponding expected reward is $r^\pi(s) \coloneqq \sum_a \pi(s, a) r(s, a)$. Assuming the Markov chain under π admits a stationary distribution d^π, let $\mathbf{D}^\pi = \operatorname{diag}(d^\pi)$ and let $\boldsymbol{r}^\pi$ denote the vector with entries $r^\pi(s_i)$. Then Eq. (1) can be written as $\mathbf{A} = \Phi^\top \mathbf{D}^\pi (\mathbf{I} - \gamma \mathbf{P}^\pi)\Phi$, and $\boldsymbol{b} = \Phi^\top \mathbf{D}^\pi \boldsymbol{r}^\pi$.

General Value Functions. A *general value function* (GVF) (Sutton et al., 2011; White, 2017) replaces the reward by another per-step signal, called a cumulant, $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and predicts $v^{\pi,c}(s) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t c(S_t, A_t) \mid S_0 = s]$. With K cumulants $\{c_k\}_{k=1}^K$ under a shared policy and shared features, each prediction has its own $\boldsymbol{b}_k = \mathbb{E}[c_k(S_t, A_t)\boldsymbol{\phi}_t]$, while all K predictions share the same matrix $\mathbf{A}$.

Replay Buffers and PER. A replay buffer is $\mathcal{B} = \{(\boldsymbol{\phi}_i, R_i, \boldsymbol{\phi}'_i, \tau_i)\}_{i=1}^N$, where $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}'_i$ are the current- and next-state feature vectors, and $\tau_i \in \{0, 1\}$ indicates whether the transition terminates an episode. Uniform replay draws from $\mathcal{B}$ with equal probability. Prioritized experience replay (PER) (Schaul et al., 2016) instead uses

$$p_i \propto |\delta_i(\boldsymbol{\theta})|^{\alpha_{\text{PER}}}, \tag{4}$$

where $\delta_i(\boldsymbol{\theta}) = R_i + \gamma(1 - \tau_i)\boldsymbol{\theta}^\top \boldsymbol{\phi}'_i - \boldsymbol{\theta}^\top \boldsymbol{\phi}_i$ and $\alpha_{\text{PER}} \geq 0$ is the priority exponent ($\alpha_{\text{PER}} = 0$ recovers uniform sampling). PER priorities depend on the reward and value prediction for the signal being learned, so changing or adding signals generally requires recomputing or aggregating reward-dependent priorities. The sampler in Section 3 targets the shared matrix $\mathbf{A}$ rather than a single TD error.

3 METHOD

The sampling problem studied here is to estimate the shared TD matrix $\mathbf{A}$ from a limited number of stored transitions or, in the model-based case, sampled states. We seek an estimator $\hat{\mathbf{A}}$ with small expected squared Frobenius error. Writing $\mathbf{A}$ as a product of two feature matrices makes this an approximate matrix multiplication (AMM) problem (Drineas et al., 2006a). For the unbiased estimator that samples rank-1 contributions with inverse-probability weights, the expected squared Frobenius error is minimized by sampling each contribution in proportion to the product of the norms of its two factors (Appendix C, Lemma 2). We use that rule to sample transitions when estimating $\mathbf{A}$, and then ask when a more accurate matrix estimate leads to more accurate value prediction.

3.1 ESTIMATING $\mathbf{A}$ AS AN AMM PROBLEM

We now pose $\mathbf{A}$-estimation as an instance of approximate matrix multiplication. Using the stationary distribution and transition matrix defined above, $\mathbf{A} = \sum_s d^\pi(s)\phi(s)\left(\phi(s) - \gamma \sum_{s'} \mathbf{P}^\pi(s,s')\phi(s')\right)^\top$, so each state contributes a rank-1 matrix to $\mathbf{A}$. With a limited sampling budget, we would like to sample states with larger contributions to $\mathbf{A}$ more often, rather than sample all states uniformly. This state-level decomposition can be written as the matrix product

$$\mathbf{A} = \Phi^\top \mathbf{D}^\pi (\mathbf{I} - \gamma \mathbf{P}^\pi)\Phi = (\Phi^\top \mathbf{D}^\pi)((\mathbf{I} - \gamma \mathbf{P}^\pi)\Phi) = (\Phi^\pi)^\top \Psi^\pi.$$

Here, $\Phi^\pi \coloneqq \mathbf{D}^\pi \Phi \in \mathbb{R}^{n \times d}$ is the feature matrix weighted row-weighted by the stationary distribution, with $\Phi^\pi_{i,:} = \phi^\pi(s_i) \coloneqq d^\pi(s_i)\phi(s_i)$. Similarly, $\Psi^\pi \coloneqq (\mathbf{I} - \gamma \mathbf{P}^\pi)\Phi \in \mathbb{R}^{n \times d}$, with $\Psi^\pi_{i,:} = \psi^\pi(s_i) \coloneqq \phi(s_i) - \gamma \sum_{s'} \mathbf{P}^\pi(s_i, s')\phi(s')$. Thus, $\psi^\pi(s)$ is the difference between the current-state feature vector and the discounted expected next-state feature vector. Therefore, $\mathbf{A}$ is the matrix product of $(\Phi^\pi)^\top$ and Ψ^π, i.e., $\mathbf{A} = (\Phi^\pi)^\top \Psi^\pi$. The rows of Φ^π are the stationary-weighted current-state features, and the rows of Ψ^π are the differences between the current-state features and the discounted expected next-state features. A simple way to estimate $\mathbf{A}$ is to sample indices uniformly or according to d^π, with inverse-probability weights; both estimators are unbiased (Lemma 1; Appendix D). Applying the AMM result gives the sampling distribution that minimizes the sum of the elementwise variances of $\hat{\mathbf{A}}$, using the norms of the rows of Φ^π and Ψ^π, while remaining unbiased. For a budget of T sampled states,

$$\hat{\mathbf{A}} = \frac{1}{T} \sum_{j=1}^{T} \frac{1}{p_{i_j}} \phi^\pi(s_{i_j}) \psi^\pi(s_{i_j})^\top, \quad \text{where } \; p_i \propto \|\phi^\pi(s_i)\|_2 \|\psi^\pi(s_i)\|_2. \tag{5}$$

Within this unbiased AMM estimator family, this sampling distribution minimizes the expected squared Frobenius error of $\hat{\mathbf{A}}$. Since each state contributes an outer product of the form $\phi^\pi(s_i)\psi^\pi(s_i)^\top$ to $\mathbf{A}$, and since $\|\phi^\pi(s_i)\psi^\pi(s_i)^\top\|_{\mathrm{F}} = \|\phi^\pi(s_i)\|_2 \|\psi^\pi(s_i)\|_2$, the rule samples states in proportion to the Frobenius norm of their individual contribution to the TD matrix $\mathbf{A}$. The model-based procedure for (5), and the uniform and d^π baselines, are in Appendix D.

3.2 WHEN A BETTER MATRIX ESTIMATE LEADS TO BETTER VALUE ESTIMATES

Norm-based sampling, with the probabilities in Eq. (5), minimizes the expected squared Frobenius error of $\hat{\mathbf{A}}$ within the unbiased AMM estimator family considered above. Whether this reduction in matrix-estimation error translates into more accurate value function estimates is not always immediately obvious: errors in $\hat{\mathbf{A}}$ are propagated through a matrix inverse, and an inverse can amplify small errors when $\hat{\mathbf{A}}$ is ill-conditioned. The next proposition shows that, for the regularized form of LSTD introduced in Eq. (3), the impact of such propagation is bounded in terms of $|\hat{\mathbf{A}} - \mathbf{A}|_{\mathrm{F}}$; the following corollary gives a linear bound when this error is sufficiently small. The argument is standard in matrix perturbation theory (Stewart & Sun, 1990; Horn & Johnson, 2013).

Proposition 1 (Perturbation bound for regularized LSTD)**.** *Let $\hat{\boldsymbol{\theta}}_\lambda = (\hat{\mathbf{A}} + \lambda \mathbf{I})^{-1}\boldsymbol{b}$ and $\boldsymbol{\theta}^*_\lambda = (\mathbf{A} + \lambda \mathbf{I})^{-1}\boldsymbol{b}$ for a fixed reward vector $\boldsymbol{b}$, and let $\sigma_\lambda \coloneqq \sigma_{\min}(\mathbf{A} + \lambda \mathbf{I})$ denote the smallest singular value of the regularized matrix. If $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} < \sigma_\lambda$, then*

$$\|\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda\|_2 \leq \frac{\|\boldsymbol{b}\|_2 \cdot \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}}{\sigma_\lambda\big(\sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}\big)}.$$

Proof. Proof in Appendix E. □

Corollary 2. *Under the assumptions of Proposition 1, if $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} \leq \frac{1}{2}\sigma_\lambda$, then*

$$\|\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda\|_2 \leq \frac{2\|\boldsymbol{b}\|_2}{\sigma_\lambda^2} \cdot \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}.$$

The corollary makes the following takeaway: when $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} \leq \frac{1}{2}\sigma\lambda$, the bound on the parameter error scales linearly with the estimation error of $\mathbf{A}$, with constant $2\|\boldsymbol{b}\|_2/\sigma_\lambda^2$. Two consequences of this corollary are worth noting. First, the bound $\|\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda\|_2$ depends on the particular reward function of interest only through $\|\boldsymbol{b}\|_2$; the σ_λ-driven amplification depends on the conditioning of $\mathbf{A} + \lambda\mathbf{I}$, and therefore on the policy-induced dynamics, the feature representation, the discount factor, and the regularization. Second, the same bound applies to *any* reward vector sharing the same matrix $\mathbf{A}$: in a multi-task setting where an agent may have to evaluate several reward functions that share dynamics and features, any reduction in $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}$ tightens the value-error bound for *all* tasks at once. This observation will be central when we contrast norm-based sampling, such as the one we introduced in this paper, with existing reward-dependent sampling alternatives.

3.3 Model-Free Norm Sampling via Replay Buffers

The model-based estimator of Eq. (5) requires knowledge of d^π and $\mathbf{P}^\pi$, which are typically unavailable in model-free RL. We now adapt that norm-sampling rule to the replay-buffer setting introduced in Section 2. Given a buffer $\mathcal{B} = \{(\boldsymbol{\phi}_i, R_i, \boldsymbol{\phi}'_i, \tau_i)\}_{i=1}^N$ collected under policy π, define the empirical TD matrix as follows:

$$\mathbf{A}_\mathcal{B} \coloneqq \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\phi}_i \boldsymbol{\psi}_i^\top, \qquad \boldsymbol{\psi}_i \coloneqq \boldsymbol{\phi}_i - \gamma(1-\tau_i)\,\boldsymbol{\phi}'_i, \tag{6}$$

where the terminal indicator τ_i zeroes out the bootstrap term at the end of episodes. This form of TD matrix, $\mathbf{A}_\mathcal{B}$, is the replay-buffer analogue of $\mathbf{A} = (\Phi^\pi)^\top \Psi^\pi$. The ith transition in the replay buffer $\mathcal{B}$ contributes a rank-1 outer product whose current-state factor is $\boldsymbol{\phi}_i$ and whose TD-feature factor is $\boldsymbol{\psi}_i$.

Applying the same rule to the rank-1 contributions in Eq. (6), the norm-based priority over transitions becomes

$$\tilde{p}_i \propto \|\boldsymbol{\phi}_i\|_2 \|\boldsymbol{\psi}_i\|_2. \tag{7}$$

Let $p_i = \tilde{p}_i / \sum_{k=1}^N \tilde{p}_k$ denote the corresponding sampling probability. Sampling from p without importance correction is biased. The unbiased AMM estimator of $\mathbf{A}_\mathcal{B}$ weights each sampled transition by $1/(Np_i)$. In the experiments we use the same PER sampling procedure, including the priority exponent, optional importance weights, and max-normalization, but replace the TD-error priority $|\delta_i|$ with the product $\|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2$ (Appendix D).

4 Experiments

In this section, we evaluate norm-based sampling for estimating $\mathbf{A}$, and study when its reward-independent priorities are useful. The goal of these experiments is not to show that norm sampling dominates reward-aware replay in every setting. Instead, we investigate *when* improving the estimate of $\mathbf{A}$ helps value prediction, and when error in other quantities, such as the reward-dependent vector $\boldsymbol{b}$, becomes the main limitation.

We organize the experiments around three questions. (1) Does norm-based sampling reduce the error in estimating $\mathbf{A}$ under a fixed sampling budget, and can the heterogeneity of the norm scores serve as a diagnostic for when the improvement is larger? (2) In what settings does a better estimate of $\mathbf{A}$ help value learning, and when do other sources of error limit its benefit? (3) Does the same norm-based priority remain useful when applied to control with nonlinear function approximation?

4.1 Model-Based Matrix Estimation and Heterogeneity Diagnostic

We first consider a model-based version of the problem. In this setting, we assume that the stationary distribution d^π and the transition matrix $\mathbf{P}^\pi$ are available. This lets us compute Φ^π, Ψ^π, and the target matrix $\mathbf{A} = (\Phi^\pi)^\top \Psi^\pi$ exactly from the chain dynamics. The purpose of this experiment is not to propose a practical model-free algorithm, but to isolate the question that motivated the paper: with a fixed sampling budget, does norm-based sampling improve the estimate of $\mathbf{A}$, and does that improvement translate into lower value error?

Table 1: Model-based estimation on the random chain with the piecewise-linear shared map $\boldsymbol{\phi}_{\texttt{shared}}$ (stride a=100) at budget T=1024 (n=1001, d=11, γ=0.99). Entries are means over 1,000 seeds, with 99% CI half-widths in parentheses. For each policy, Frob. is $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}$ and Val. is $\|\Phi\hat{\mathbf{A}}^{-1}\boldsymbol{b} - \Phi\mathbf{A}^{-1}\boldsymbol{b}\|_{d^\pi}$ with population $\boldsymbol{b}$; lower is better, and the best method on each metric is in bold. $\mathrm{CV}(u)$ is the coefficient of variation of $u_i = \|\boldsymbol{\phi}^\pi(s_i)\|_2\|\boldsymbol{\psi}^\pi(s_i)\|_2$. Full sample-budget curves, and the tabular and random representations, are in Figures 4 and 5.

		Uniform		d^π		Norm	
Policy	$\mathrm{CV}(u)$	Frob.	Val.	Frob.	Val.	Frob.	Val.
π_{r}	0.44	$2.16(4)\times10^{-4}$	$\mathbf{1.87(7)\times10^{-3}}$	$2.33(10)\times10^{-4}$	$3.10(16)\times10^{-3}$	$\mathbf{1.93(3)\times10^{-4}}$	$2.18(10)\times10^{-3}$
$\pi_{\rightarrow}$	3.65	$9.80(52)\times10^{-4}$	$2.74(26)\times10^{-1}$	$8.90(52)\times10^{-4}$	$2.65(25)\times10^{-1}$	$\mathbf{2.72(5)\times10^{-4}}$	$\mathbf{1.21(7)\times10^{-2}}$
$\pi_{\Rightarrow}$	5.43	$1.80(11)\times10^{-3}$	$9.28(98)\times10^{-1}$	$1.40(8)\times10^{-3}$	$5.13(73)\times10^{-1}$	$\mathbf{3.75(8)\times10^{-4}}$	$\mathbf{1.50(9)\times10^{-2}}$

For this experiment, we use the random-chain domain described in Appendix F. We consider three feature representations—tabular one-hot ($\boldsymbol{\phi}_{\texttt{tabular}}$), normalized random features ($\boldsymbol{\phi}_{\texttt{random}}$), and a piecewise-linear shared map ($\boldsymbol{\phi}_{\texttt{shared}}$)—and three policies of increasing rightward skew, moving right with probability 0.5, 0.75, and 0.9 ($\pi_{\mathsf{r}}, \pi_{\rightarrow}, \pi_{\Rightarrow}$). For each pairing of feature representation and policy, we compare three ways of sampling states from $\mathcal{S}$ to estimate $\mathbf{A}$: (i) *uniform sampling* with the importance correction in (12); (ii) d^π*-sampling*, where states are sampled according to the stationary distribution as in (13); and (iii) *norm-based sampling*, with $p_i \propto \|\boldsymbol{\phi}^\pi(s_i)\|_2\|\boldsymbol{\psi}^\pi(s_i)\|_2$ as in Eq. (5).

We measure two errors: the first is the direct matrix-estimation error, $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}$. The second is the error in the value function estimated by solving the LSTD system using $\hat{\mathbf{A}}$, while keeping $\boldsymbol{b}$ fixed, i.e., $\|\Phi\hat{\mathbf{A}}^{-1}\boldsymbol{b} - \Phi\boldsymbol{\theta}^*\|_{d^\pi}$, where $\boldsymbol{\theta}^* = \mathbf{A}^{-1}\boldsymbol{b}$. In this experiment, we keep $\boldsymbol{b}$ fixed. This allows us to attribute any changes in value error to the quality of the matrix estimate alone. Later, in Appendix G, we return to the more realistic case where both $\mathbf{A}$ and $\boldsymbol{b}$ are estimated from finite data.

Table 1 shows the results for the shared feature representation $\boldsymbol{\phi}_{\texttt{shared}}$ at a fixed sample budget. We observe two main trends. First, under the skewed policies, norm-based sampling reduces the Frobenius error of $\hat{\mathbf{A}}$ compared to the two baselines; under π_{r} the three estimators are close. This is consistent with the AMM result: the advantage of norm-based sampling is larger when the state contributions to $\mathbf{A}$ are more uneven. Second, under the more skewed policies, the improvements in $\hat{\mathbf{A}}$ are also reflected in the value function estimation error when $\boldsymbol{b}$ is fixed, although this translation is not uniform across all policies. This shows that, in these settings, the reduction in matrix-estimation error can also translate into substantially lower value error.

The amount of improvement, however, is not the same in every setting. As the policy becomes more skewed, first under $\pi_{\rightarrow}$ and then under $\pi_{\Rightarrow}$, the gap between norm-based sampling and the baselines becomes more pronounced. Appendix F shows that the same qualitative pattern holds for the other feature representations: norm-based sampling helps most when the feature representation and the policy make the contributions to $\mathbf{A}$ highly non-uniform. Norm-based sampling samples in proportion to $u_i = \|\boldsymbol{\phi}^\pi(s_i)\|_2\,\|\boldsymbol{\psi}^\pi(s_i)\|_2$, for each state s_i. If the scores $\{u_i\}$ are nearly constant across states, then norm-based sampling approaches uniform sampling. If these scores vary substantially, uniform sampling spends much of its budget on states whose rank-1 contributions have small norm. Let $\mathcal{I}$ index states on the chain and stored transitions in the replay buffer. We measure that variation by the coefficient of variation, i.e., $\mathrm{CV}_{\mathcal{I}}(u) = \frac{\mathrm{std}_{i\in\mathcal{I}}(u_i)}{\mathrm{mean}_{i\in\mathcal{I}}(u_i)}$, a standard heterogeneity diagnostic for importance weights (Owen, 2013; Kong, 1992). In the random-chain experiments, the largest gains occur where $\mathrm{CV}_{\mathcal{I}}(u)$ is largest, as with shared features under skewed policies.

The same diagnostic can be computed in the replay-buffer setting using $\boldsymbol{\psi}_i = \boldsymbol{\phi}_i - \gamma(1-\tau_i)\boldsymbol{\phi}'_i$ (Eq. (6)). On Four Rooms we vary only the feature map, so that $\mathrm{CV}_{\mathcal{I}}(u)$ ranges from nearly homogeneous to highly heterogeneous, and estimate $\mathbf{A}$ from a batch of B=1000 transitions (Appendix F.7, Figure 6). The Frobenius gain of norm-based sampling over uniform replay, and over PER with task-averaged oracle TD-error priorities, grows with $\mathrm{CV}_{\mathcal{I}}(u)$.

4.2 Multi-Reward GVFs and Reusing Reward-Agnostic Priorities

A difficult setting for reward-agnostic sampling is a single sparse reward, where reward-specific information can be especially useful. We now turn to the setting that motivated the method. In a GVF

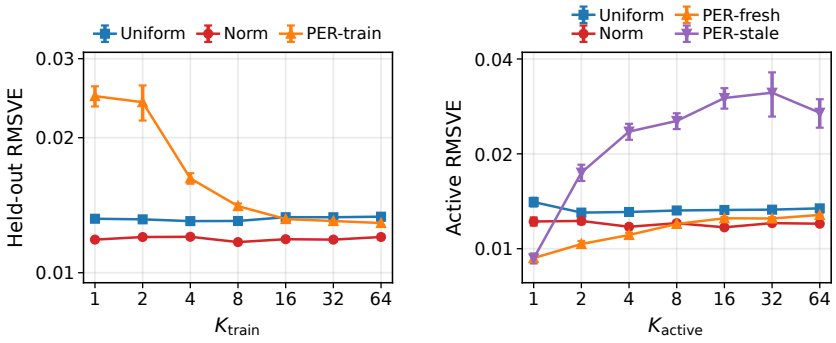


Figure 1: Reusing priorities across GVF tasks at $B = 500$ (300 seeds). Left: held-out generalization to 50 unseen goal-reaching GVFs after priorities are built from K_{train} seen tasks. Right: task arrival under a doubling schedule, where new tasks are added over time. Norm uses no reward information; PER-train and PER-fresh use reward-dependent TD errors from the available tasks.

setting, the agent predicts *many* signals under the same policy and feature representation (Sutton et al., 2011; McLeod et al., 2021). Each signal has its own reward-dependent vector $\boldsymbol{b}_k$, but all of the predictions share $\mathbf{A}$. If a single sampled estimate $\hat{\mathbf{A}}$ is reused across many rewards or cumulants, a more accurate estimate of $\mathbf{A}$can improve many value functions at once.

The domain is the continuing Four Rooms gridworld. Appendix F.7 uses the same environment while varying the feature map; here the representation is held fixed and the number of rewards varies. The behavior policy is uniform random, the replay buffer has 5,000 transitions per seed, and each state is encoded with RBF features ($\sigma = 0.1$, $d = 25$). We form K goal-reaching GVFs by selecting random goal states and defining the cumulant for task k as $c_k(s') = \mathbb{1}s' = g_k$, with $K \in \{1, 2, 4, 8, 16, 32, 64\}$. We compare uniform replay, norm-based replay, and three reward-dependent PER variants. PER-avg averages the magnitude of the oracle TD errors across all K tasks before applying the standard exponent 0.6. PER-worst uses the maximum TD error across tasks. PER-avg-budget5 averages TD-error magnitudes over the first $\min(K, 5)$ tasks, so the priority uses a fixed task budget as K grows. The held-out-task panel uses PER-train, whose priorities are built once from the training tasks, while the task-arrival panel compares PER-fresh, which recomputes priorities after each task round, with PER-stale, which keeps the priorities from the initial task set.

Norm-based priorities depend only on $\boldsymbol{\phi}_i$ and $\boldsymbol{\psi}_i$, so they can be built before the downstream cumulants are known. A PER distribution is tied to the rewards used to build it. If new GVFs must be estimated later, or if the replay distribution must be chosen before those cumulants are known, a reward-dependent priority either remains tied to the tasks used to construct it or has to be recomputed.

Figure 1 (left) reports held-out RMSVE on 50 unseen goal-reaching GVFs when PER-train priorities are computed from K_{train} training tasks. At $K_{\text{train}} = 1$, the PER-train priorities transfer poorly to the held-out tasks: held-out RMSVE is 0.0248, compared with 0.0119 for norm-based sampling. As K_{train} grows, PER-train moves closer to norm-based sampling, but at $K_{\text{train}} = 64$ it is still slightly worse (0.0129 versus 0.0120).

The right panel is a task-arrival schedule. Tasks arrive in rounds with cumulative $K_{\text{active}} \in \{1, 2, 4, 8, 16, 32, 64\}$. PER-fresh recomputes priorities every round from the tasks active so far, at a cost of $K_{\text{active}}N$ TD-error evaluations per round and 635,000 evaluations over the full schedule. PER-stale pays that cost only once, and its error grows as new tasks are added without updating its priorities, consistent with known limitations of stale PER priorities (Pan et al., 2022). Norm-based priorities have no reward-dependent cost. Norm has higher error than PER-fresh at the smallest active-task counts, becomes competitive as more tasks arrive, and has lower error at larger active-task counts in this experiment.

Norm-based replay is not a replacement for PER. For a single reward, or for per-step TD in which $\hat{\boldsymbol{b}}_k$ is a major source of error, reward-aware sampling is better aligned with the objective. When many GVFs share the same dynamics and features, an estimate of the shared matrix $\mathbf{A}$ can be reused

across all tasks. One reward-agnostic sampler can then support every task, including tasks that were unknown when the replay distribution was built.

Figure 9 (Appendix H) separates these two cases. In closed-form transfer, $\hat{\mathbf{A}}$ comes from the sampled batch and each $\boldsymbol{b}_k$ comes from the full buffer; norm-based replay overtakes the PER variants as K grows and stays ahead of uniform replay. In iterative TD, every mini-batch estimates both sides of the update; uniform replay is more stable, and norm-based replay remains about a factor of two worse. A better $\hat{\mathbf{A}}$ is most useful when error in the shared matrix is an important source of value error. It does not by itself resolve the error in per-step TD, where the reward-dependent part of the update is also estimated from each mini-batch. The useful regime is the one in which a shared matrix estimate is estimated once or updated independently of any particular reward, and then reused across many reward vectors.

5 From Linear Prediction to Deep Control

The experiments above use fixed features and a fixed policy. We next ask whether the same norm-based sampling remains useful when those two assumptions are dropped: the features used for priorities are learned, and the agent optimizes a control objective rather than evaluating a fixed policy. The sampling rule is the same as in Section 3.3; only the representation f used to compute the priority changes.

Setup. We train a deep Q-network (DQN) on four MinAtar games (Young & Tian, 2019; Obando-Ceron & Castro, 2021): Asterix, Breakout, Freeway, and Space Invaders. Norm-based replay uses the same priority as in the linear setting, $p_i \propto \|f_i\|_2 \|f_i - \gamma(1-\tau_i) f_i'\|_2$, where f now denotes the representation used to compute the priority. We consider four choices of f: the raw observation, the target network's penultimate trunk activation $\phi \in \mathbb{R}^{128}$ (the ReLU feature before the Q-head), a last-layer gradient approximation, and the parameter-gradient norm, denoted fullgrad. The main-text comparisons use $f = \phi$; we call this sampler norm-ϕ. The observation, gradient, and fullgrad variants appear in Appendix J. Each run uses 10^6 environment steps, a replay buffer of 10^5 transitions, and 30 seeds. The network, the priority-refresh interval, and the PER exponent grid are also in that appendix. Two detached linear heads share the trunk: a successor-feature head (Dayan, 1993; Barreto et al., 2017) and a 32-cumulant GVF head with a frozen random projection (Jaderberg et al., 2017; Veeriah et al., 2019). Stop-gradient into the trunk keeps the Q-update unchanged across these auxiliary prediction heads (Alain & Bengio, 2016).

Return depends on the game. Figure 2 shows uniform replay, PER with $\alpha = 0.6$, and norm-ϕ ($f = \phi$), after selecting the learning rate for each game and batch size using return AUC (area under the learning curve) (Agarwal et al., 2021). Norm-ϕ has the highest or comparable return on Freeway at nearly every batch, on Space Invaders, and on mid-batch Asterix, and its return falls more slowly as the batch grows. PER is higher on the smallest Asterix batches. The methods are similar on Breakout. Smaller batches can improve deep RL return in isolation (Obando-Ceron et al., 2023); here we compare samplers across matched batch sizes. Figure 10 repeats the comparison for all eight samplers at one shared learning rate and three training horizons. The ordering is the same, and the large-batch gaps are larger in that sweep.

The linear experiments used the coefficient of variation of $u_i = \|\boldsymbol{\phi}_i\|_2 \|\boldsymbol{\psi}_i\|_2$ as a diagnostic for when norm sampling reduces the error in $\mathbf{A}$ (Appendix F.7). On MinAtar the analogous statistic is the Gini coefficient of $u_i = \|f_i\|_2 \|f_i - \gamma(1-\tau_i) f_i'\|_2$, where larger values indicate more heterogeneous priority scores, computed from a randomly initialized network on 10,000 transitions before training. For $f = \phi$ the initialization values are 0.510 (Asterix), 0.405 (Freeway), 0.300 (Space Invaders), and 0.106 (Breakout). We use these values only as an exploratory measure of initial priority heterogeneity, not as a general predictor of return. The score histograms for $f = \phi$, and for the observation- and gradient-based choices of f, are in Appendix J (Figures 11 and 12).

Detached prediction heads. The detached heads provide several auxiliary predictions that share the same dynamics and learned features, analogous to the multi-reward GVFs in Section 4.2. The bottom row of Figure 2 is the GVF Bellman residual on a fixed set of buffer indices, at the learning rates selected for return. The residual is the TD loss on the replay buffer, and the buffers differ

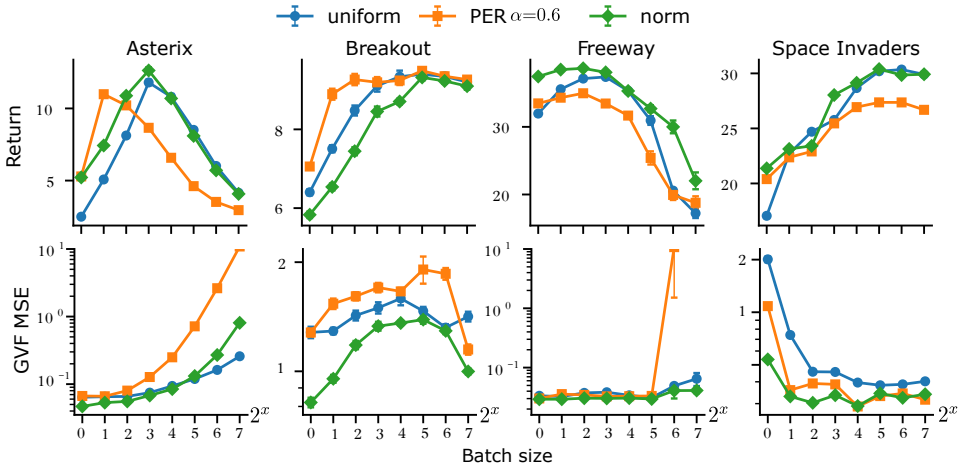


Figure 2: DQN return (top) and detached GVF Bellman residual (bottom) versus batch size, after selecting each cell's learning rate by return AUC. Thirty seeds. Curves show uniform, PER with $\alpha = 0.6$, and norm-ϕ ($f = \phi$, target penultimate trunk). The GVF axis is logarithmic. Norm-ϕ is best or tied on Freeway, Space Invaders, and mid-batch Asterix; Breakout is a tie and PER is higher on the smallest Asterix batches. On Asterix and Freeway, the PER residual grows sharply with batch size. The complete sweeps are Figures 10 and 13.

across samplers. On Asterix and Freeway, PER's residual grows sharply with batch size, while norm-ϕ stays at or below uniform. On Breakout and Space Invaders, norm-ϕ has the lowest or comparable residual at most batches. Appendix J contains the results for different sampler variants.

6 Conclusion

In this paper, we studied replay sampling for linear TD through the decomposition of the expected update into a reward-dependent vector $\boldsymbol{b}$ and a shared matrix $\mathbf{A}$. The vector $\boldsymbol{b}$ depends on the particular reward or cumulant, whereas $\mathbf{A}$ depends on the policy-induced dynamics, the feature representation, and the discount factor, but not on the reward being predicted. This separation suggests matching the sampler to the object that limits estimation. Formulating estimation of $\mathbf{A}$ as approximate matrix multiplication gives the norm-based rule $p_i \propto \|\boldsymbol{\phi}_i\|_2 \|\boldsymbol{\psi}_i\|_2$, which minimizes the expected squared Frobenius error within the unbiased AMM estimator family considered. In the random-chain and replay-buffer experiments, its advantage is largest when these per-transition contributions are heterogeneous; the coefficient of variation of their norms serves as a diagnostic for this heterogeneity. In the MinAtar experiments, we use the Gini coefficient of the corresponding learned-feature scores as an exploratory measure of the same property.

Our main empirical result concerns the multi-task setting. When many GVFs share the same dynamics and features and their reward-dependent vectors are computed from the full buffer, one estimate of $\mathbf{A}$ can be reused across tasks. Norm-based replay then improves value prediction over uniform replay and eventually outperforms reward-dependent PER as the number of tasks grows. Because its priorities do not depend on rewards, it also transfers to held-out tasks and to tasks introduced after the replay distribution was constructed. This does not mean that a better $\mathbf{A}$ estimate always improves value learning: for single-reward tasks, and for iterative TD where each mini-batch also carries reward-side noise, PER remains better aligned with the objective. Norm-based sampling targets error in the shared matrix; it is not designed to identify the transitions most informative for one particular reward.

The MinAtar experiments test whether norm-based sampling extends beyond fixed linear features. Norm-based replay gives game-dependent improvements with DQN, with larger gains occurring in some games with more heterogeneous learned-feature scores, but it is not a uniform replacement for PER. Overall, norm-based replay is most useful when error in the shared matrix is important, and the same estimate can be reused across predictions, whereas reward-aware replay is better suited to settings where task-specific error dominates.

AI USE STATEMENT

In this work, we used generative AI tools, to assist with drafting and revising the manuscript, improving clarity and readability, identifying undefined notation and inconsistencies, and suggesting alternative phrasing and organization. We also used generative AI to provide feedback on the framing and scope of mathematical claims and on the interpretation and presentation of experimental results, as well as for limited brainstorming and discussion of related work.

The core research ideas, theoretical results, experimental methodology, and reported experimental results were developed by the authors. Generative AI was not used to generate synthetic data, conduct experiments, or determine reported numerical results. It was also not used for translation, dataset cleaning or reformatting, or qualitative or thematic data analysis; these tasks were not applicable to this work.

All AI-assisted text, mathematical suggestions, and interpretations were reviewed and revised by the authors. Mathematical claims were checked against the corresponding derivations and proofs, experimental statements were checked against the underlying results, and literature-related suggestions were checked against the cited sources. AI-generated suggestions that were unsupported or inconsistent with the analysis or experiments were not included. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The sampling procedures are in Appendix D, and the proofs of the matrix-multiplication lemmas and of Proposition 1 are in Appendices C and E. Environment, feature, batch-size, and training details for each experiment are in Sections 4 and 5 and in the corresponding appendices. Anonymized code that regenerates these experiments is included as supplementary material, with run instructions in `code/README.md`.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2016.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in neural information processing systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/file/453fadbd8a1a3af50a9df4df899537b5-Paper.pdf.

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, David Silver, and Hado van Hasselt. Successor features for transfer in reinforcement learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 4055–4065, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/350db081a661525235354dd3e19b8c05-Abstract.html.

Jalaj Bhandari, Daniel Russo, and Raghav Singal. A finite time analysis of temporal difference learning with linear function approximation. In *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pp. 1691–1692. PMLR, 2018.

Justin A. Boyan. Technical update: Least-squares temporal difference learning. *Mach. Learn.*, 49(2-3):233–246, 2002. doi: 10.1023/A:1017936530646. URL https://doi.org/10.1023/A:1017936530646.

Steven J. Bradtke and Andrew G. Barto. Linear Least-Squares algorithms for temporal difference learning. *Machine Learning*, 22(1):33–57, March 1996. ISSN 1573-0565. doi: 10.1007/BF00114723.

Cheng Chen, Weinan Zhang, and Yong Yu. Efficient policy evaluation by matrix sketching. *Frontiers of Computer Science*, 16(5):165330, January 2022. ISSN 2095-2236. doi: 10.1007/s11704-021-0354-4.

Wesley Chung, Somjit Nath, Ajin Joseph, and Martha White. Two-timescale networks for nonlinear value function approximation. In *International Conference on Learning Representations*, 2019.

Michael B. Cohen, Jelani Nelson, and David P. Woodruff. Optimal approximate matrix product in terms of stable rank. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 11:1–11:14. Schloss Dagstuhl, 2016. doi: 10.4230/LIPIcs.ICALP.2016.11.

Christoph Dann, Gerhard Neumann, and Jan Peters. Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, 15:809–883, 2014.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Comput.*, 5(4):613–624, 1993. doi: 10.1162/NECO.1993.5.4.613. URL https://doi.org/10.1162/neco.1993.5.4.613.

Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast monte carlo algorithms for matrices i: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006a. doi: 10.1137/S0097539704442684. URL https://doi.org/10.1137/S0097539704442684.

Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Sampling algorithms for ℓ_2 regression and applications. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1127–1136. SIAM, 2006b.

Sylvester Eriksson-Bique, Mary Solbrig, Michael Stefanelli, Sarah Warkentin, Ralph Abbey, and Ilse C. F. Ipsen. Importance sampling for a Monte Carlo matrix multiplication algorithm, with application to information retrieval. *SIAM Journal on Scientific Computing*, 33(4):1689–1706, 2011. doi: 10.1137/10080659X.

Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesvári, and Shie Mannor. Regularized policy iteration. In *Advances in Neural Information Processing Systems*, volume 21, 2008.

William Fedus, Prajit Ramachandran, Rishabh Agarwal, Yoshua Bengio, Hugo Larochelle, Mark Rowland, and Will Dabney. Revisiting fundamentals of experience replay. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3061–3071. PMLR, 2020. URL http://proceedings.mlr.press/v119/fedus20a.html.

Scott Fujimoto, David Meger, and Doina Precup. An equivalence between loss functions and non-uniform sampling in experience replay. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Clement Gehring, Yangchen Pan, and Martha White. Incremental truncated lstd. In *International Joint Conference on Artificial Intelligence*, pp. 1505–1511, 2016.

Alborz Geramifard, Michael Bowling, and Richard S. Sutton. Incremental least-squares temporal difference learning. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pp. 356–361. AAAI Press, 2006.

Mohammad Ghavamzadeh, Alessandro Lazaric, Odalric Maillard, and Rémi Munos. LSTD with random projections. In *Advances in Neural Information Processing Systems*, volume 23, 2010.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11694.

Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11796.

John T. Holodnak and Ilse C. F. Ipsen. Randomized approximation of the Gram matrix: Exact computation and probabilistic bounds. *SIAM Journal on Matrix Analysis and Applications*, 36, 2015. doi: 10.1137/130940116.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. In *International Conference on Learning Representations*, 2018.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 2 edition, 2013.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

Scott Jordan, Yash Chandak, Daniel Cohen, Mengxue Zhang, and Philip Thomas. Evaluating the performance of reinforcement learning algorithms. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, 2020. URL https://proceedings.mlr.press/v119/jordan20a.html.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2525–2534. PMLR, 2018. URL https://proceedings.mlr.press/v80/katharopoulos18a.html.

J. Zico Kolter and Andrew Y. Ng. Regularization and feature selection in least-squares temporal difference learning. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 521–528. ACM, 2009.

Augustine Kong. A note on importance sampling using standardized weights. Technical Report 348, Department of Statistics, University of Chicago, July 1992.

George Konidaris, Sarah Osentoski, and Philip Thomas. Value function approximation in reinforcement learning using the fourier basis. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pp. 380–385. AAAI Press, 2011.

Aviral Kumar, Abhishek Gupta, and Sergey Levine. DisCor: Corrective feedback in reinforcement learning via distribution correction. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021.

Thibault Lahire, Matthieu Geist, and Emmanuel Rachelson. Large batch experience replay. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11790–11813. PMLR, 2022. URL https://proceedings.mlr.press/v162/lahire22a.html.

Alessandro Lazaric, Mohammad Ghavamzadeh, and Rémi Munos. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13:3041–3074, 2012.

Nir Levine, Tom Zahavy, Daniel J. Mankowitz, Aviv Tamar, and Shie Mannor. Shallow updates for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992.

Clare Lyle, Mark Rowland, and Will Dabney. Understanding and preventing capacity loss in reinforcement learning. In *International Conference on Learning Representations*, 2022.

Avner Magen and Anastasios Zouzias. Low rank matrix-valued Chernoff bounds and approximate matrix multiplication. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1422–1436. SIAM, 2011.

A. Rupam Mahmood, Hado van Hasselt, and Richard S. Sutton. Weighted importance sampling for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

Matthew McLeod, Chunlok Lo, Matthew Schlegel, Andrew Jacobsen, Raksha Kumaraswamy, Martha White, and Adam White. Continual auxiliary task learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021. arXiv comment: Neural Information Processing Systems 2021.

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, 2015.

Joseph Modayil, Adam White, and Richard S. Sutton. Multi-timescale nexting in a reinforcement learning robot. *Adapt. Behav.*, 22(2):146–160, 2014. doi: 10.1177/1059712313511648. URL https://doi.org/10.1177/1059712313511648.

Deanna Needell, Nathan Srebro, and Rachel Ward. Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Johan Obando-Ceron, Marc G. Bellemare, and Pablo Samuel Castro. Small batch deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Johan S. Obando-Ceron and Pablo Samuel Castro. Revisiting rainbow: Promoting more insightful and inclusive deep reinforcement learning research. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1373–1383. PMLR, 2021.

Art B. Owen. *Monte Carlo Theory, Methods and Examples*. Art B. Owen, 2013. URL https://artowen.su.domains/mc/.

Yangchen Pan, Erfan Sadeqi Azer, and Martha White. Effective sketching methods for value function approximation. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*, 2017. URL https://www.auai.org/uai2017/proceedings/papers/81.pdf.

Yangchen Pan, Jincheng Mei, Amir-massoud Farahmand, Martha White, Hengshuai Yao, Mohsen Rohani, and Jun Luo. Understanding and mitigating the limitations of prioritized experience replay. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pp. 1561–1571. PMLR, 2022. arXiv comment: Accepted to UAI 2022.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1312–1320. PMLR, 2015.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016.

G. W. Stewart and Ji-guang Sun. *Matrix Perturbation Theory*. Academic Press, 1990.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Mach. Learn.*, 3:9–44, 1988. doi: 10.1007/BF00115009. URL https://doi.org/10.1007/BF00115009.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2 edition, 2018.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999. ISSN 0004-3702.

Richard S. Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M. Pilarski, Adam White, and Doina Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*, pp. 761–768, 2011.

Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*, 17(73):1–29, 2016.

John N Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

Vivek Veeriah, Matteo Hessel, Zhongwen Xu, Janarthanan Rajendran, Richard L. Lewis, Junhyuk Oh, Hado P. van Hasselt, David Silver, and Satinder Singh. Discovery of useful questions as auxiliary tasks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Adam White. *Developing a predictive approach to knowledge*. PhD thesis, University of Alberta, 2015.

Martha White. Unifying task specification in reinforcement learning. In *International Conference on Machine Learning*, pp. 3742–3750. PMLR, 2017.

Kenny Young and Tian Tian. MinAtar: An Atari-inspired testbed for thorough and reproducible reinforcement learning experiments. arXiv:1903.03176, 2019. URL https://arxiv.org/abs/1903.03176.

Shangtong Zhang and Richard S. Sutton. A deeper look at experience replay. *CoRR*, abs/1712.01275, 2017. URL http://arxiv.org/abs/1712.01275.

Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1–9. PMLR, 2015. URL https://proceedings.mlr.press/v37/zhaoa15.html.

A Related Work

Prioritized and non-uniform replay. Prioritized experience replay (PER) samples transitions according to a power of their TD error and is widely used in deep RL (Schaul et al., 2016; Horgan et al., 2018; Hessel et al., 2018). Prior work has related non-uniform replay to reweighted losses (Fujimoto et al., 2020) and to importance sampling for gradient estimation in large-batch replay (Lahire et al., 2022). Other methods reweight transitions without using TD error directly (Kumar et al., 2020), while analyses of PER have identified limitations caused by stale priorities and incomplete coverage (Pan et al., 2022). Our approach differs in what the sampling distribution is designed to estimate: rather than prioritizing transitions using the TD error or gradient of a particular signal, we formulate estimation of the shared TD matrix $\mathbf{A}$ as an approximate matrix multiplication problem. This gives reward-independent priorities that can be reused across multiple predictions.

Multi-prediction and shared structure. General value functions (GVFs) and related frameworks use the same stream of experience to learn many long-term predictions (Sutton et al., 2011; Modayil et al., 2014; White, 2015; Schaul et al., 2015). Goal-conditioned methods similarly reuse transitions across multiple targets (Andrychowicz et al., 2017). Continual auxiliary-task learning also studies settings in which many predictions share dynamics while differing in their task-specific signals (McLeod et al., 2021). This is closely related to the decomposition used here, where the matrix $\mathbf{A}$ is shared across predictions while each reward or cumulant has its own vector $\boldsymbol{b}$. Auxiliary prediction heads in deep RL provide another setting in which several signals are learned from a shared representation and stream of experience (Jaderberg et al., 2017; Veeriah et al., 2019).

LSTD, sketching, and approximate matrix multiplication. Least-squares TD (LSTD) estimates the matrix $\mathbf{A}$ and vector $\boldsymbol{b}$ from transitions (Bradtke & Barto, 1996; Boyan, 2002; Dann et al., 2014). Maintaining that system costs $O(d^2)$ computation per update, and incremental approximations have been proposed to reduce the cost for sparse features (Geramifard et al., 2006). Finite-sample analyses of TD and LSTD study how estimation error depends on the amount of available data (Lazaric et al., 2012; Bhandari et al., 2018). Sketching and randomized projections have also been used to reduce the cost of LSTD and related estimators (Ghavamzadeh et al., 2010; Pan et al., 2017; Chen et al., 2022; Gehring et al., 2016). Our approach instead uses classical approximate matrix multiplication (AMM), where sampling rows or columns according to their norms minimizes Frobenius error for the corresponding unbiased product estimator (Drineas et al., 2006a; Mahoney, 2011; Eriksson-Bique et al., 2011; Holodnak & Ipsen, 2015). Related randomized linear-algebra methods use leverage scores to control linear-system solution error (Drineas et al., 2006b), and spectral-norm guarantees for AMM are also known (Magen & Zouzias, 2011; Cohen et al., 2016). We use the Frobenius AMM objective because it leads directly to an explicit, reward-independent sampling rule for estimating $\mathbf{A}$.

Importance sampling and iterative updates. For iterative SGD-style optimization, sampling examples in proportion to gradient norms can reduce the variance of stochastic updates (Needell et al., 2014; Zhao & Zhang, 2015; Katharopoulos & Fleuret, 2018). This helps explain why reward-aware prioritization can be preferable when each mini-batch must estimate both the shared matrix term and the reward-dependent part of the TD update. Importance weighting and partial corrections have also been used to stabilize learning under non-uniform or off-policy sampling (Mahmood et al., 2014; Sutton et al., 2016). Following PER, our model-free implementation therefore uses a priority exponent α and an importance-sampling exponent β to control the strength of prioritization and correction.

Deep features and evaluation. Least-squares and LSTD-style prediction heads have previously been applied on top of learned deep features (Levine et al., 2017; Chung et al., 2019). The norms and effective capacity of learned representations can also change during training (Kumar et al., 2021; Lyle et al., 2022), motivating our examination of how heterogeneous the resulting norm-based priorities are in MinAtar. We treat the corresponding Gini coefficient only as an exploratory measure of priority heterogeneity. For the deep-control experiments, we select learning rates using return AUC and report results over multiple seeds, following standard evaluation practices in deep RL (Agarwal et al., 2021; Henderson et al., 2018; Jordan et al., 2020).

B Extended Background

This appendix expands the background of Section 2. The main-text equations for $\mathbf{A}$, $\boldsymbol{b}$, LSTD, regularized LSTD, and PER are unchanged; the material here is for completeness.

B.1 Markov Decision Processes and Policy Evaluation

Reinforcement learning problems are typically formulated as a Markov decision process (MDP) (Puterman, 1994), defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma, d_0)$. Here, $\mathcal{S}$ is the set of states and $\mathcal{A}$ is the set of actions. We denote the random variables for state, action, and reward at time $t \in \{0, 1, 2, \ldots\}$ by S_t, A_t, and R_t.

We assume the MDP is stationary: the transition dynamics and the reward function do not depend on t. The expected per-step reward $r\colon \mathcal{S}\times\mathcal{A} \to [-R_{\max}, R_{\max}]$, for a finite $R_{\max} > 0$, is $r(s,a) \coloneqq \mathbb{E}[R_t \mid S_t{=}s, A_t{=}a]$. The one-step transition kernel is $p\colon \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0,1]$, with $p(s,a,s') \coloneqq \Pr(S_{t+1}{=}s' \mid S_t{=}s, A_t{=}a)$. The starting-state distribution $d_0\colon \mathcal{S} \to [0,1]$ is the probability of beginning an episode in a given state at $t = 0$.

A stationary policy is $\pi\colon \mathcal{S}\times\mathcal{A} \to [0,1]$, with $\pi(s,a) \coloneqq \Pr(A_t{=}a \mid S_t{=}s)$. The agent selects $A_t \sim \pi(S_t, \cdot)$, observes R_t, and the environment transitions to $S_{t+1} \sim p(S_t, A_t, \cdot)$. Let $G_t \coloneqq \sum_{k=0}^{\infty} \gamma^k R_{t+k}$ denote the discounted return for $\gamma \in [0,1)$.

Policy evaluation estimates the value function $v^\pi\colon \mathcal{S} \to \mathbb{R}$, $v^\pi(s) \coloneqq \mathbb{E}[G_t \mid S_t{=}s; \pi]$. This function satisfies the Bellman equation (Sutton & Barto, 2018), $v^\pi(s) = \sum_a \pi(s,a)\big(r(s,a) + \gamma \sum_{s'} p(s,a,s') v^\pi(s')\big)$. When $\mathcal{S}$ is large, a tabular representation of v^π is impractical, so we approximate it with a linear function of features.

B.2 Policy Evaluation in the Linear Setting

Let $\boldsymbol{\theta} \in \mathbb{R}^d$ and let $\phi(s) \in \mathbb{R}^d$ be the feature of state s. The linear approximation is $v_{\boldsymbol{\theta}}(s) = \boldsymbol{\theta}^\top \phi(s)$, and we write $\boldsymbol{\phi}_t \coloneqq \phi(S_t)$. The TD update is $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\delta_t\boldsymbol{\phi}_t$ with $\delta_t = R_t + \gamma\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t$, equivalently

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\Big(R_t\boldsymbol{\phi}_t - \boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\boldsymbol{\theta}_t\Big).$$

Holding $\boldsymbol{\theta}_t$ fixed and taking the expectation under π yields $\mathbb{E}[\boldsymbol{\theta}_{t+1} \mid \boldsymbol{\theta}_t] = \boldsymbol{\theta}_t + \alpha(\boldsymbol{b} - \mathbf{A}\boldsymbol{\theta}_t)$, with $\mathbf{A}$ and $\boldsymbol{b}$ as in Eq. (1). The TD fixed point (Tsitsiklis & Van Roy, 1997) is Eq. (2), and the corresponding least-squares TD (LSTD) method (Bradtke & Barto, 1996; Boyan, 2002) estimates $\mathbf{A}$ and $\boldsymbol{b}$ from observed transitions and solves the linear system. When $\hat{\mathbf{A}}$ is ill-conditioned, regularized LSTD uses the ridge form in Eq. (3), trading a small bias toward the origin for invertibility (Kolter & Ng, 2009; Farahmand et al., 2008).

B.3 Vector Notation for Policy Evaluation

Let $n = |\mathcal{S}|$ and let $\Phi \in \mathbb{R}^{n\times d}$ be the feature matrix whose i^{th} row is $\Phi_{i,:} \coloneqq \phi(s_i)$, where s_i denotes the ith state in a fixed ordering of $\mathcal{S}$ (distinct from the random state S_t). Let $\mathbf{P}^\pi \in \mathbb{R}^{n\times n}$ with $\mathbf{P}^\pi(s_i, s_j) \coloneqq \Pr(S_{t+1}{=}s_j \mid S_t{=}s_i; \pi) = \sum_a \pi(s_i,a)p(s_i,a,s_j)$. Let $d^\pi : \mathcal{S} \to [0,1]$ be the stationary distribution under π, let $\mathbf{d}^\pi \in [0,1]^n$ with $\mathbf{d}^\pi_i \coloneqq d^\pi(s_i)$, and let $\mathbf{D}^\pi = \mathtt{diag}(\mathbf{d}^\pi)$. Let $\boldsymbol{r}^\pi \in \mathbb{R}^n$ with $\boldsymbol{r}^\pi_i = \mathbb{E}[R_t \mid S_t{=}s_i; \pi] = \sum_a \pi(s_i,a)r(s_i,a)$. Under d^π, Eq. (1) has the compact form $\mathbf{A} = \Phi^\top\mathbf{D}^\pi(\mathbf{I} - \gamma\mathbf{P}^\pi)\Phi$ and $\boldsymbol{b} = \Phi^\top\mathbf{D}^\pi\boldsymbol{r}^\pi$.

B.4 General Value Functions and Multi-Task Prediction

The standard policy-evaluation setup estimates a single value function v^π for a single reward r. An agent may instead need multiple predictions under the same policy, such as discounted sums of sensory signals, time-to-event quantities, or task-specific bonuses. *General value functions* (GVFs) (Sutton et al., 2011; White, 2017) replace the reward by a cumulant $c : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$. For fixed π and c,

$$v^{\pi,c}(s) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t c(S_t, A_t) \;\middle|\; S_0 = s\right].$$

A GVF is therefore an ordinary value function for an alternative reward signal, so the same TD and LSTD machinery applies.

The multi-task structure used in this paper is a collection of K predictions $\{v^{\pi,c_k}\}_{k=1}^K$ that share the policy and transition dynamics but differ in the cumulants $\{c_k\}_{k=1}^K$. In the linear setting, each prediction has its own reward vector $\boldsymbol{b}_k = \mathbb{E}[c_k(S_t, A_t)\boldsymbol{\phi}_t]$, while all K problems share the same matrix $\mathbf{A}$ from Eq. (1).

B.5 Replay Buffers and Prioritized Experience Replay

A buffer is $\mathcal{B} = \{(\boldsymbol{\phi}_i, R_i, \boldsymbol{\phi}'_i, \tau_i)\}_{i=1}^N$, where $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}'_i$ are the features of the current and next state of the ith stored transition, R_i is the observed reward, and $\tau_i \in \{0, 1\}$ is a terminal indicator ($\tau_i = 1$ if the transition ends the episode). Uniform replay draws from $\mathcal{B}$ with equal probability.

Prioritized experience replay (PER) (Schaul et al., 2016) assigns each transition a priority proportional to a power of its TD error, as in Eq. (4), where $\delta_i(\boldsymbol{\theta}) = R_i + \gamma(1-\tau_i)\boldsymbol{\theta}^\top\boldsymbol{\phi}'_i - \boldsymbol{\theta}^\top\boldsymbol{\phi}_i$ and $\alpha_{\text{PER}} \geq 0$ ($\alpha_{\text{PER}} = 0$ recovers uniform sampling). Analyses of replay also show that replay ratio, reuse, and the sampling distribution can affect learning dynamics (Zhang & Sutton, 2017; Fedus et al., 2020). Because the sampling distribution changes, PER typically applies importance weights of the form $(Np_i)^{-\beta}$ to correct, or partially correct, the resulting bias.

Two properties of Eq. (4) motivate the sampler in Section 3. First, PER priorities are reward-dependent: δ_i couples each transition to a given observed reward, so a priority structure must be maintained separately for each signal. Second, PER targets the magnitude of a per-transition TD error rather than the variance of an estimator of the shared matrix $\mathbf{A}$. The objective in this paper is the latter: reward-agnostic structure shared across many predictions.

C Approximate Matrix Multiplication

Suppose we wish to approximate the product of two matrices, $\mathbf{B} \in \mathbb{R}^{l\times m}$ and $\mathbf{C} \in \mathbb{R}^{m\times n}$. The exact product can be written as a sum of outer products,[1]

$$\mathbf{BC} = \sum_{i=1}^{m} \mathbf{B}_{:,i}\mathbf{C}_{i,:}^\top. \tag{8}$$

Instead of summing over all m terms in Eq. (8), we can construct an unbiased stochastic estimator by sampling indices $i_1, \ldots, i_T$ from $\{1, \ldots, m\}$ and weighting each sampled outer product by the inverse of its sampling probability. Under uniform sampling, this gives

$$\widehat{\mathbf{BC}}_{\text{unif}} = \frac{m}{T}\sum_{t=1}^{T} \mathbf{B}_{:,i_t}\mathbf{C}_{i_t,:}, \qquad i_t \sim \text{Uniform}(\{1, \ldots, m\}).$$

The AMM result of Drineas et al. (2006a) shows that the sum of elementwise variances, equivalently the expected squared Frobenius error of the unbiased estimator of $\mathbf{BC}$, is minimized by sampling index i with probability proportional to the product of the two associated terms, $p_i \propto \|\mathbf{B}_{:,i}\|_2\|\mathbf{C}_{i,:}\|_2$.[2] We refer to sampling distributions of this form as *norm-based sampling* rules.

Lemma 1. *(Lemma 3, Drineas et al. (2006a)) Suppose $\mathbf{B} \in \mathbb{R}^{l\times m}, \mathbf{C} \in \mathbb{R}^{m\times n}, T \in \mathbb{Z}^+$, such that $1 \leq T \leq m$ and indices are sampled according to $p \in \mathbb{R}^m$, such that $\forall i, p_i \geq 0$ and $\sum_{i=1}^m p_i = 1$. When the approximation $\widehat{\mathbf{BC}}$ is constructed as*

$$\widehat{\mathbf{BC}} = \frac{1}{T}\sum_{t=1}^{T} \frac{1}{p_{i_t}}\mathbf{B}_{:,i_t}\mathbf{C}_{i_t,:}^\top, \quad \textit{where}, i_t \sim p, \tag{9}$$

then,

$$\mathbb{E}\Big[(\widehat{\mathbf{BC}})_{i,j}\Big] = (\mathbf{BC})_{i,j}, \quad \mathbf{Var}(\widehat{\mathbf{BC}})_{i,j} = \frac{1}{T}\sum_{k=1}^{m} \frac{\mathbf{B}_{i,k}^2\mathbf{C}_{k,j}^2}{p_k} - \frac{1}{T}(\mathbf{BC})_{i,j}^2. \tag{10}$$

[1] For a matrix $\mathbf{X}$, we use $\mathbf{X}_{:,i}$ to refer to its ith column and $\mathbf{X}_{i,:}$ to refer to its ith row.

[2] All vectors are column vectors, so $\mathbf{C}_{i,:}^\top$ denotes the row vector corresponding to the ith row of $\mathbf{C}$.

Proof. Let us define $\mathbf{X}_t$ as the tth sampled rank-one approximation matrix product $(\widehat{\mathbf{BC}})$, i.e.,

$$\mathbf{X}_t := \frac{(\mathbf{B}_{:,i_t}\mathbf{C}_{i_t,:}^\top)}{Tp_{i_t}}, \text{ hence, } \widehat{\mathbf{BC}} = \sum_{t=1}^{T}\mathbf{X}_t.$$

The expected value of $\mathbf{X}_t$ is

$$\mathbb{E}[\mathbf{X}_t] = \sum_{k=1}^{m}\frac{1}{T}\frac{(\mathbf{B}_{:,k}\mathbf{C}_{k,:}^\top)}{p_k} \times p_k = \frac{1}{T}\sum_{k=1}^{m}(\mathbf{B}_{:,k}\mathbf{C}_{k,:}^\top)$$

As $\mathbf{BC} = \sum_{k=1}^{m}(\mathbf{B}_{:,k}\mathbf{C}_{k,:}^\top)$ (sum of rank-1 matrices), we get,

$$\mathbb{E}[\mathbf{X}_t] = \frac{1}{T}(\mathbf{BC})$$

Hence, the expectation of $\mathbf{X}_t$ of $(i,j)^{\text{th}}$ element is

$$\mathbb{E}[(\mathbf{X}_t)_{i,j}] = \frac{1}{T}(\mathbf{BC})_{i,j}$$

Consider the $(i,j)^{\text{th}}$ element of $\mathbf{X}_t$,

$$(\mathbf{X}_t)_{i,j} = \frac{(\mathbf{B}_{i,i_t}\mathbf{C}_{i_t,j})}{Tp_{i_t}} \implies (\mathbf{X}_t)_{i,j}^2 = \frac{(\mathbf{B}_{i,i_t}\mathbf{C}_{i_t,j})^2}{T^2p_{i_t}^2}$$

The expected value of $\mathbf{X}_t^2$ for $(i,j)^{th}$ element is

$$\begin{aligned}\mathbb{E}\big[(\mathbf{X}_t^2)_{i,j}\big] &= \frac{1}{T^2}\sum_{k=1}^{m}\frac{1}{p_k^2}(\mathbf{B}_{i,k}\mathbf{C}_{k,j})^2 \times p_k \\ &= \frac{1}{T^2}\sum_{k=1}^{m}\frac{1}{p_k}(\mathbf{B}_{i,k})^2(\mathbf{C}_{k,j})^2\end{aligned}$$

Therefore, the variance of the tth one-rank approximation is given by

$$\mathbf{Var}(\mathbf{X}_t)_{i,j} = \mathbb{E}\big[(\mathbf{X}_t^2)_{i,j}\big] - (\mathbb{E}[(\mathbf{X}_t)_{i,j}])^2 = \frac{1}{T^2}\sum_{k=1}^{m}\frac{1}{p_k}(\mathbf{B}_{i,k})^2(\mathbf{C}_{k,j})^2 - \frac{1}{T^2}(\mathbf{BC})_{i,j}^2.$$

Applying linearity of expectation, we get,

$$\mathbb{E}\Big[(\widehat{\mathbf{BC}})_{i,j}\Big] = \mathbb{E}\left[\sum_{t=1}^{T}(\mathbf{X}_t)_{i,j}\right] = \sum_{t=1}^{T}\mathbb{E}[(\mathbf{X}_t)_{i,j}] = \sum_{t=1}^{T}\frac{1}{T}(\mathbf{BC})_{i,j} = (\mathbf{BC})_{i,j}.$$

Applying linearity of variance under i.i.d., sampling and with replacement, we get,

$$\mathbf{Var}(\widehat{\mathbf{BC}})_{i,j} = \sum_{t=1}^{T}\mathbf{Var}(\mathbf{X}_t)_{i,j} = \frac{1}{T}(\sum_{k=1}^{m}\frac{\mathbf{B}_{i,k}^2\mathbf{C}_{k,j}^2}{p_k}) - \frac{1}{T}(\mathbf{BC})_{i,j}^2.$$

Hence proved. □

Lemma 2. *(Lemma 4, Drineas et al. (2006a)) Suppose $\mathbf{B} \in \mathbb{R}^{l\times m}, \mathbf{C} \in \mathbb{R}^{m\times n}, T \in \mathbb{Z}^+$, such that $1 \le T \le m$ and indices are sampled according to $p \in \mathbb{R}^m$, such that $\forall i, p_i \ge 0$ and $\sum_i p_i = 1$. When $\widehat{\mathbf{BC}}$ is constructed as Eq.* (9) *and if*

$$p_i = \frac{\|\mathbf{B}_{:,i}\|_2\|\mathbf{C}_{i,:}\|_2}{\sum_{j=1}^{m}\|\mathbf{B}_{:,j}\|_2\|\mathbf{C}_{j,:}\|_2},$$

then $\sum_{i=1}^{l}\sum_{j=1}^{n}\mathbf{Var}(\widehat{\mathbf{BC}})_{i,j}$ is minimized and is equal to

$$\begin{aligned}&\sum_{i=1}^{l}\sum_{j=1}^{n}\mathbf{Var}(\widehat{\mathbf{BC}})_{i,j} \\ &= \frac{1}{T}\left(\sum_{k=1}^{m}\|\mathbf{B}_{:,k}\|_2\|\mathbf{C}_{k,:}\|_2\right)^2 - \frac{1}{T}\|\mathbf{BC}\|_{\text{F}}^2,\end{aligned}$$

where $\|\cdot\|_{\text{F}}$ is the Frobenius norm.

Proof. When $(\widehat{\mathbf{BC}})$ is constructed as given in Lemma 1, its variance is given by (10).

Let us define a function f with parameters $\{p_i\}_{i=1}^m$ as the sum of variance over all matrix elements, i.e.,

$$\begin{aligned} f(p_1, p_2, ..p_m) &= \sum_{i=1}^{l}\sum_{j=1}^{n} \mathbf{Var}(\widehat{\mathbf{BC}})_{i,j} \\ &= \frac{1}{T}(\sum_{k=1}^{m}\frac{1}{p_k}\sum_{i=1}^{l}\sum_{j=1}^{n}\mathbf{B}_{i,k}^2\mathbf{C}_{k,j}^2) - \frac{1}{T}\sum_{i=1}^{l}\sum_{j=1}^{n}(\mathbf{BC})_{i,j}^2 \\ &= \frac{1}{T}(\sum_{k=1}^{m}\frac{1}{p_k}\sum_{i=1}^{l}\mathbf{B}_{i,k}^2\|\mathbf{C}_{k,:}\|_2^2) - \frac{1}{T}\|\mathbf{BC}\|_{\mathrm{F}}^2 \\ &= \frac{1}{T}(\sum_{k=1}^{m}\frac{1}{p_k}\|\mathbf{B}_{:,k}\|_2^2\|\mathbf{C}_{k,:}\|_2^2) - \frac{1}{T}\|\mathbf{BC}\|_{\mathrm{F}}^2 \end{aligned}$$

$$\implies f(p_1, p_2, ..p_m) = \frac{1}{T}\sum_{k=1}^{m}\frac{\|\mathbf{B}_{:,k}\|_2^2\|\mathbf{C}_{k,:}\|_2^2}{p_k} - \frac{1}{T}\|\mathbf{BC}\|_{\mathrm{F}}^2 \tag{11}$$

Then, we reformulate the problem of minimizing (11) by using Lagrange multiplier (λ) to include the constraint $\sum_{i=1}^m p_i = 1$, we get,

$$g(p_1, p_2, ..p_m, \lambda) = f(p_1, p_2, ..p_m) + \lambda(\sum_{i=1}^{m} p_i - 1).$$

Equating the first-order differential to 0 to determine the optimal parameters, we get,

$$\frac{\partial g(p_1, p_2, ..p_m, \lambda)}{\partial p_i} = \frac{-1}{T}\frac{\|\mathbf{B}_{:,i}\|_2^2\|\mathbf{C}_{i,:}\|_2^2}{p_i^2} + \lambda = 0,\ \forall\, i \in \{1, 2, ..m\}$$

Solving for p_i, we get,

$$p_i^* = \frac{\|\mathbf{B}_{:,i}\|_2\|\mathbf{C}_{i,:}\|_2}{\sqrt{T\lambda}},\ \ \forall\, i \in \{1, 2, ..m\}$$

Determining the appropriate λ such that the constraint $\sum_{i=1}^m p_i = 1$ satisfies, we get p_i as,

$$p_i^* = \frac{\|\mathbf{B}_{:,i}\|_2\|\mathbf{C}_{i,:}\|_2}{\sum_{j=1}^m \|\mathbf{B}_{:,j}\|_2\|\mathbf{C}_{j,:}\|_2},\ \forall\, i \in \{1, 2, ..m\}.$$

Further,

$$\begin{aligned} \sum_{i=1}^{l}\sum_{j=1}^{n}\mathbf{Var}(\widehat{\mathbf{BC}})_{i,j} &= \frac{1}{T}\sum_{k=1}^{m}\frac{\|\mathbf{B}_{:,k}\|_2^2\|\mathbf{C}_{k,:}\|_2^2}{p_k^*} - \frac{1}{T}\|\mathbf{BC}\|_{\mathrm{F}}^2 \\ &= \frac{1}{T}\left(\sum_{k=1}^{m}\|\mathbf{B}_{:,k}\|_2\|\mathbf{C}_{k,:}\|_2\right)^2 - \frac{1}{T}\|\mathbf{BC}\|_{\mathrm{F}}^2. \end{aligned}$$

□

D SAMPLING PROCEDURES

Unbiased AMM estimators of $\mathbf{A} = (\Phi^\pi)^\top\Psi^\pi$ take the form $\hat{\mathbf{A}} = \frac{1}{T}\sum_{j=1}^T \frac{1}{p_{i_j}}\phi^\pi(s_{i_j})\psi^\pi(s_{i_j})^\top$ (Lemma 1). Uniform sampling and d^π-sampling are the same estimator with different proposals:

$$\hat{\mathbf{A}} = \frac{1}{T}\sum_{j=1}^{T} n\phi^\pi(s_{i_j})\psi^\pi(s_{i_j})^\top,\ \text{ where }\ i_j \sim \texttt{Uniform}(\{1, 2, \ldots, n\}),\ \text{ or} \tag{12}$$

$$\hat{\mathbf{A}} = \frac{1}{T}\sum_{j=1}^{T} \frac{1}{d^\pi(s_{i_j})}\phi^\pi(s_{i_j})\psi^\pi(s_{i_j})^\top, \quad \text{where } s_{i_j} \sim d^\pi. \tag{13}$$

The model-based procedure for the norm-based proposal in Eq. (5) is given in Algorithm 1. The uniform and d^π baselines are recovered by choosing the corresponding proposal and importance weight.

Input: Stationary distribution d^π, feature matrix Φ, transition matrix $\mathbf{P}^\pi$, discount factor γ, number of samples T
Output: Unbiased estimate $\hat{\mathbf{A}}$ of $\mathbf{A} = (\Phi^\pi)^\top \Psi^\pi$
Initialize: $\hat{\mathbf{A}} \leftarrow 0$, $\mathbf{D}^\pi \leftarrow \texttt{diag}(d^\pi)$, $n \leftarrow |\mathcal{S}|$
Compute $\Phi^\pi \leftarrow \mathbf{D}^\pi\Phi$ and $\Psi^\pi \leftarrow (\mathbf{I} - \gamma\mathbf{P}^\pi)\Phi$
Norm-based sampling probabilities, Eq. (5) **for** *each state index* $k \in \{1, 2, \ldots, n\}$ **do**
  $p_k \leftarrow \|\Phi^\pi_{k,:}\|_2 \|\Psi^\pi_{k,:}\|_2$;
end
for *each state index* $k \in \{1, 2, \ldots, n\}$ **do**
  $p_k \leftarrow p_k / \sum_j p_j$;
end
Accumulate inverse-probability weighted rows of Φ^π and Ψ^π **for** *each sample* $t \in \{1, 2, \ldots, T\}$ **do**
  Sample index $k \sim p$;
  $\hat{\mathbf{A}} \leftarrow \hat{\mathbf{A}} + \frac{1}{Tp_k}\Phi^\pi_{k,:}(\Psi^\pi_{k,:})^\top$;
end
return $\hat{\mathbf{A}}$;

Algorithm 1: Model-Based Norm Sampling for $\hat{\mathbf{A}}$

Algorithm 1 requires oracle access to d^π and $\mathbf{P}^\pi$. On a replay buffer the variance-minimizing priority is $\tilde{p}_i \propto \|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2$ as in Eq. (7). Sampling from $\tilde{p}$ without importance correction is biased. If we wanted the exact *unbiased* AMM estimator of $\mathbf{A}_\mathcal{B}$, each sampled transition would carry the factor $1/(Np_i)$. Under finite replay budgets, however, this exact correction can be dominated by rare low-probability transitions with large importance weights. We therefore use the same stabilizing machinery as prioritized experience replay (Schaul et al., 2016); namely a priority exponent $\alpha \geq 0$ that interpolates between uniform sampling ($\alpha = 0$) and the full-norm priority ($\alpha = 1$), and an importance-sampling exponent $\beta \in [0, 1]$ that controls how much of the correction is applied (Mahmood et al., 2014). In particular, given (α, β), we introduce the following prioritization mechanism, which explicitly allows bias-correction and stochasticity to be regulated:

$$p_i = \frac{\big(\|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2\big)^\alpha}{\sum_{k=1}^{N}\big(\|\boldsymbol{\phi}_k\|_2\|\boldsymbol{\psi}_k\|_2\big)^\alpha}.$$

Based on such priorities, we draw a batch $\{i_j\}_{j=1}^{B}$ of samples from the replay buffer $\mathcal{B}$ with replacement and assign each sample the max-normalized weight:

$$w_j = \frac{(N\,p_{i_j})^{-\beta}}{\max_{k\in\{1,\ldots,B\}}(N\,p_{i_k})^{-\beta}}.$$

The matrix estimate $\hat{\mathbf{A}}$, then, is computed as follows:

$$\hat{\mathbf{A}} = \frac{1}{B}\sum_{j=1}^{B} w_j\,\boldsymbol{\phi}_{i_j}\,\boldsymbol{\psi}_{i_j}^\top.$$

The fully corrected choice, that is, setting $(\alpha, \beta) = (1, 1)$ and not performing max-normalization, recovers the unbiased AMM estimator. For $\alpha < 1$, priorities are flattened toward uniform; for $\beta < 1$, the correction is only partially applied, trading a small amount of bias for lower variance; and the per-batch normalization clips the largest weight in the batch to one. Non-uniform sampling without full correction also changes the implied state weighting and can shift the TD fixed point (Sutton et al.,

2016). The linear experiments use the raw product ($\alpha = 1$). For the MinAtar experiments, norm-based replay omits importance-sampling correction ($w_j = 1$). PER still applies it, with β annealed as in Schaul et al. (2016). A fully corrected norm variant is reported in Appendix J. Algorithm 2 is PER's sampler with $|\delta_i|$ replaced by $\|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2$.

Input: Replay buffer $\mathcal{B} = \{(\boldsymbol{\phi}_i, R_i, \boldsymbol{\phi}'_i, \tau_i)\}_{i=1}^N$, discount factor γ, batch size B, priority exponent α, IS exponent β
Output: Estimated matrix $\hat{\mathbf{A}}$
Per-transition TD features and base priorities **for** *each transition* $i \in \{1, 2, \ldots, N\}$ **do**
  $\boldsymbol{\psi}_i \leftarrow \boldsymbol{\phi}_i - \gamma(1-\tau_i)\,\boldsymbol{\phi}'_i$;
  $\tilde{p}_i \leftarrow \big(\|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2\big)^{\alpha}$;
end
$p_i \leftarrow \tilde{p}_i \,/\, \sum_{k=1}^N \tilde{p}_k$;
# Sample batch and form max-normalized IS weights Sample $\{i_j\}_{j=1}^B$ with replacement, $i_j \sim p$;
for $j \in \{1, 2, \ldots, B\}$ **do**
  $\tilde{w}_j \leftarrow (N\, p_{i_j})^{-\beta}$;
end
$w_j \leftarrow \tilde{w}_j \,/\, \max_{k=1,\ldots,B} \tilde{w}_k$;
Weighted matrix estimate $\hat{\mathbf{A}} \leftarrow \dfrac{1}{B}\sum_{j=1}^B w_j\, \boldsymbol{\phi}_{i_j}\, \boldsymbol{\psi}_{i_j}^\top$;
return $\hat{\mathbf{A}}$;

Algorithm 2: Norm-Based Replay for $\hat{\mathbf{A}}$ (PER-style)

E Proof of Proposition 1

Proposition 1 (Perturbation bound for regularized LSTD). *Let* $\hat{\boldsymbol{\theta}}_\lambda = (\hat{\mathbf{A}} + \lambda\mathbf{I})^{-1}\boldsymbol{b}$ *and* $\boldsymbol{\theta}^*_\lambda = (\mathbf{A} + \lambda\mathbf{I})^{-1}\boldsymbol{b}$*, and let* $\sigma_\lambda = \sigma_{\min}(\mathbf{A} + \lambda\mathbf{I})$*. If* $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} < \sigma_\lambda$*, then*

$$\|\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda\|_2 \le \frac{\|\boldsymbol{b}\|_2 \cdot \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}}{\sigma_\lambda\big(\sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}\big)}.$$

Proof. The proof proceeds in three steps: a resolvent identity for the difference, a submultiplicative bound on the resulting product, and a Weyl-style bound on the perturbed inverse.

Step 1: Resolvent identity. Define $\mathbf{M} \coloneqq \mathbf{A} + \lambda\mathbf{I}$ and $\hat{\mathbf{M}} \coloneqq \hat{\mathbf{A}} + \lambda\mathbf{I}$, so $\boldsymbol{\theta}^*_\lambda = \mathbf{M}^{-1}\boldsymbol{b}$ and $\hat{\boldsymbol{\theta}}_\lambda = \hat{\mathbf{M}}^{-1}\boldsymbol{b}$. Using the resolvent identity $\hat{\mathbf{M}}^{-1} - \mathbf{M}^{-1} = \hat{\mathbf{M}}^{-1}(\mathbf{M} - \hat{\mathbf{M}})\mathbf{M}^{-1}$ and noting that $\mathbf{M} - \hat{\mathbf{M}} = \mathbf{A} - \hat{\mathbf{A}}$,

$$\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda = (\hat{\mathbf{M}}^{-1} - \mathbf{M}^{-1})\boldsymbol{b} = \hat{\mathbf{M}}^{-1}(\mathbf{A} - \hat{\mathbf{A}})\mathbf{M}^{-1}\boldsymbol{b}.$$

Step 2: Submultiplicative bound. Taking the ℓ_2 norm and applying submultiplicativity,

$$\|\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda\|_2 \le \|\hat{\mathbf{M}}^{-1}\|_2 \cdot \|\hat{\mathbf{A}} - \mathbf{A}\|_2 \cdot \|\mathbf{M}^{-1}\|_2 \cdot \|\boldsymbol{b}\|_2. \tag{14}$$

Step 3: Bounding each factor. We bound each factor of (14) in turn:

(a) $\|\mathbf{M}^{-1}\|_2 = 1/\sigma_{\min}(\mathbf{M}) = 1/\sigma_\lambda$.

(b) $\|\hat{\mathbf{A}} - \mathbf{A}\|_2 \le \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}$, since the spectral norm is bounded by the Frobenius norm.

(c) By Weyl's singular-value perturbation inequality,

$$\sigma_{\min}(\hat{\mathbf{M}}) \ge \sigma_{\min}(\mathbf{M}) - \|\hat{\mathbf{M}} - \mathbf{M}\|_2 = \sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_2.$$

Since $\|\hat{\mathbf{A}} - \mathbf{A}\|_2 \le \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} < \sigma_\lambda$ by assumption, $\hat{\mathbf{M}}$ is invertible and

$$\|\hat{\mathbf{M}}^{-1}\|_2 = \frac{1}{\sigma_{\min}(\hat{\mathbf{M}})} \le \frac{1}{\sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}}.$$

Substituting (a)–(c) into (14) yields

$$\|\hat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*_\lambda\|_2 \le \frac{\|\boldsymbol{b}\|_2 \cdot \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}}{\sigma_\lambda\big(\sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}\big)}.$$

The corollary follows immediately: if $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} \le \frac{1}{2}\sigma_\lambda$, then $\sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}} \ge \frac{1}{2}\sigma_\lambda$, so $\sigma_\lambda(\sigma_\lambda - \|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}) \ge \frac{1}{2}\sigma_\lambda^2$, giving the constant $2\|\boldsymbol{b}\|_2/\sigma_\lambda^2$. □

F Experimental Setup and Additional Model-Based Results

F.1 Random Chain

The random-chain MDP (Sutton, 1988; Sutton & Barto, 2018), illustrated in Figure 3, is the shared evaluation domain throughout the model-based experiments of this paper. The state space consists of an odd number of states arranged in a line; the agent starts in the centre state, and at each step takes one of two actions ($a_{\texttt{left}}$ or $a_{\texttt{right}}$), deterministically transitioning to the corresponding neighbour. All transitions yield reward 0 except the rightmost-state-to-terminal transition, which yields $+1$, and the leftmost-state-to-terminal transition, which yields -1. We use the continuing variant: rather than terminating, the agent transitions back to the centre state.

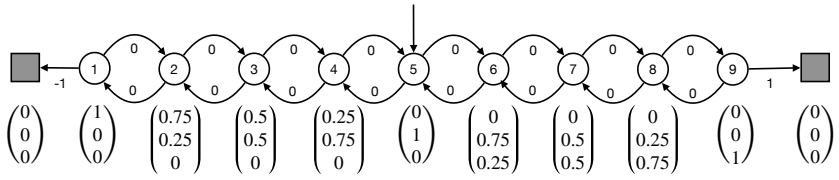


Figure 3: Random Chain for $n = 9$ states. Transition rewards are shown adjacent to the arrows; the vector below each state node is the *shared* feature representation $\boldsymbol{\phi}_{\texttt{shared}}$.

F.2 Feature Representations

We experiment with three feature maps:

- *Tabular* ($\boldsymbol{\phi}_{\texttt{tabular}}$): a one-hot vector for each state, $\boldsymbol{\phi}_{\texttt{tabular}}(\cdot) \in \mathbb{R}^n$, used with $n = 101$.
- *Random* ($\boldsymbol{\phi}_{\texttt{random}}$): a normalized random feature vector for each state, $\boldsymbol{\phi}_{\texttt{random}}(\cdot) \in \mathbb{R}^d$, used with $n = 1001$ and $d = 11$.
- *Shared* ($\boldsymbol{\phi}_{\texttt{shared}}$): the piecewise-linear interpolation illustrated in Figure 3: a one-hot vector at every a-th state, and a convex combination of the two adjacent basis vectors in between. We use $n = 1001$ and stride $a = 100$, so $\boldsymbol{\phi}_{\texttt{shared}}(\cdot) \in \mathbb{R}^{11}$.

F.3 Policies

We evaluate three on-policy distributions of increasing skewness:

- π_{r} (random): $\pi_{\mathsf{r}}(\cdot, a_{\texttt{left}}) = \pi_{\mathsf{r}}(\cdot, a_{\texttt{right}}) = 0.5$.
- $\pi_{\rightarrow}$ (right-skewed): $\pi_{\rightarrow}(\cdot, a_{\texttt{left}}) = 0.25$, $\pi_{\rightarrow}(\cdot, a_{\texttt{right}}) = 0.75$.
- $\pi_{\Rightarrow}$ (extreme right-skewed): $\pi_{\Rightarrow}(\cdot, a_{\texttt{left}}) = 0.1$, $\pi_{\Rightarrow}(\cdot, a_{\texttt{right}}) = 0.9$.

F.4 Population Objects, Sampling Strategies, and Metrics

Population objects. For each pairing of feature representation and policy, we compute the population objects d^π, Φ^π, Ψ^π, $\mathbf{A}$, and $\boldsymbol{b}$ exactly from the chain dynamics. The stationary distribution d^π is recovered as the dominant left eigenvector of $\mathbf{P}^\pi$; the matrix $\mathbf{A} = \Phi^\top \mathbf{D}^\pi(\mathbf{I} - \gamma\mathbf{P}^\pi)\Phi$ and the

reward-side vector $\boldsymbol{b} = \Phi^\top \mathbf{D}^\pi \boldsymbol{r}^\pi$ are then formed by direct multiplication, with $\gamma = 0.99$. Computing these in closed form means the only stochasticity in $\hat{\mathbf{A}}$ comes from the finite sample budget T and the choice of sampling distribution.

Sampling strategies. We compare three estimators of $\mathbf{A}$, each drawing T states with replacement from a distribution p and accumulating the rank-1 outer products $\phi^\pi(s_{i_j})\,\psi^\pi(s_{i_j})^\top$ with the appropriate importance correction:

- *uniform sampling* (12): $p_i = 1/n$, with the IS factor n in front of each contribution to keep the estimator unbiased;
- d^π*-sampling* (13): $p_i = d^\pi(s_i)$, the natural baseline that draws states proportional to the stationary distribution under π, with IS factor $1/d^\pi(s_i)$;
- *norm-based sampling* (5): $p_i \propto \|\phi^\pi(s_i)\|_2 \|\psi^\pi(s_i)\|_2$, the variance-minimizing AMM proposal of Lemma 2, with IS factor $1/p_i$.

Metrics. We report two quantities for each estimator. The first is the matrix error $\|\hat{\mathbf{A}} - \mathbf{A}\|_{\mathrm{F}}$, the Frobenius distance between the estimate and the population matrix. The second is the value error: we form $\hat{\boldsymbol{\theta}} = \hat{\mathbf{A}}^{-1}\boldsymbol{b}$ and the corresponding approximated value function $\hat{\mathbf{v}} = \Phi\hat{\boldsymbol{\theta}} \in \mathbb{R}^n$, and compare it against the population-solution value function $\hat{\mathbf{v}}^* = \Phi\boldsymbol{\theta}^*$ where $\boldsymbol{\theta}^* = \mathbf{A}^{-1}\boldsymbol{b}$, reporting

$$\|\hat{\mathbf{v}}^* - \hat{\mathbf{v}}\|_{d^\pi} = \sqrt{\sum_s d^\pi(s)\big(\hat{v}^*(s) - \hat{v}(s)\big)^2}.$$

We use the population $\boldsymbol{b}$ in both solves so that any change in value error is attributable to $\hat{\mathbf{A}}$ alone. Each curve in Figures 4 and 5 below is averaged over 1,000 independent seeds; shaded regions show 99% confidence intervals.

The same d^π-weighted value error is the RMSVE reported for the replay-buffer experiments in Sections G and 4.2. For an estimate $\boldsymbol{\theta}$ and a reference $\boldsymbol{\theta}_{\mathrm{ref}}$,

$$\mathrm{RMSVE}(\boldsymbol{\theta}, \boldsymbol{\theta}_{\mathrm{ref}}) \coloneqq \sqrt{\sum_s d^\pi(s)\,\big(\phi(s)^\top(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ref}})\big)^2}. \tag{15}$$

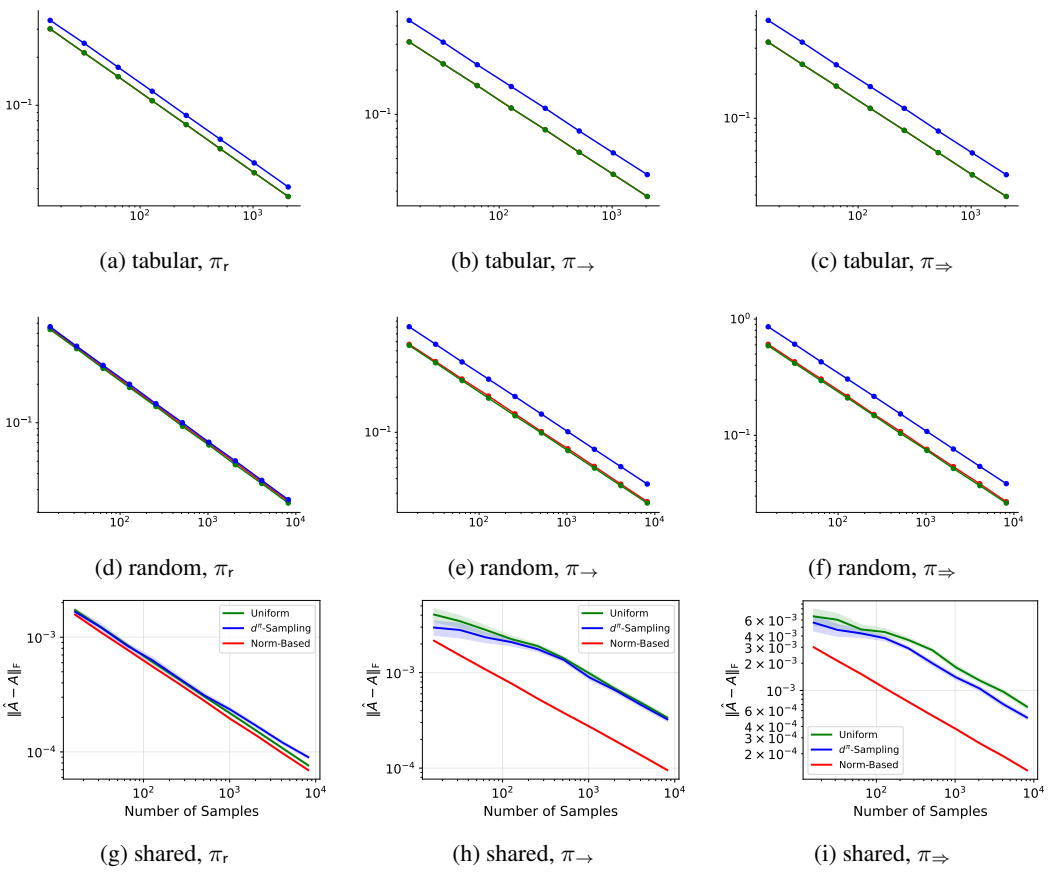
This is the d^π-weighted norm of the difference between the two induced value functions. The chain quantity $\|\hat{\mathbf{v}}^* - \hat{\mathbf{v}}\|_{d^\pi}$ is the special case with $\boldsymbol{\theta}_{\mathrm{ref}}$ the population TD solution. In Four Rooms we take $\boldsymbol{\theta}_{\mathrm{ref}}$ to be the regularized full-buffer LSTD solution.

F.5 Frobenius Error Across Feature Representations

Figure 4 extends the Frobenius comparison of Table 1 from the body to all three feature representations. Two factors visibly drive the gap between norm-based sampling and the alternatives. *Feature structure*: norm-based sampling has its strongest advantage on the shared representation, where the row-norm product $\|\phi^\pi(s)\|_2\|\psi^\pi(s)\|_2$ varies substantially across states; the gap is more modest on the tabular and random representations, where row norms are nearly uniform and the AMM-optimal proposal collapses toward the baselines. *Policy skewness*: as the policy moves from π_{r} through $\pi_{\rightarrow}$ to $\pi_{\Rightarrow}$, the stationary distribution becomes increasingly concentrated and the row-norm product becomes increasingly heterogeneous, so the variance-minimizing sampler diverges further from d^π- and uniform sampling. The two effects compound, with the largest gaps appearing in the bottom-right panel of Figure 4 (shared features, $\pi_{\Rightarrow}$). Norm-based sampling is also more sample-efficient under skewed policies—the sharp drop in error happens roughly one to two orders of magnitude earlier in the sample budget than for d^π- and uniform sampling.

F.6 Value Error Across Feature Representations

Figure 5 extends the value-error comparison of Table 1 similarly. The trend at the value-error level mirrors the matrix-error level: the same dependence on feature structure (largest gaps with $\boldsymbol{\phi}_{\texttt{shared}}$, smaller with $\boldsymbol{\phi}_{\texttt{tabular}}$ and $\boldsymbol{\phi}_{\texttt{random}}$), the same dependence on policy skewness (largest gaps under $\pi_{\Rightarrow}$), and the same sample-efficiency advantage in the skewed-policy panels. These curves solve the unregularized system $\hat{\boldsymbol{\theta}} = \hat{\mathbf{A}}^{-1}\boldsymbol{b}$ with the chain's single reward held fixed, so they show that



(a) tabular, π_r (b) tabular, $\pi_\rightarrow$ (c) tabular, $\pi_\Rightarrow$

(d) random, π_r (e) random, $\pi_\rightarrow$ (f) random, $\pi_\Rightarrow$

(g) shared, π_r (h) shared, $\pi_\rightarrow$ (i) shared, $\pi_\Rightarrow$

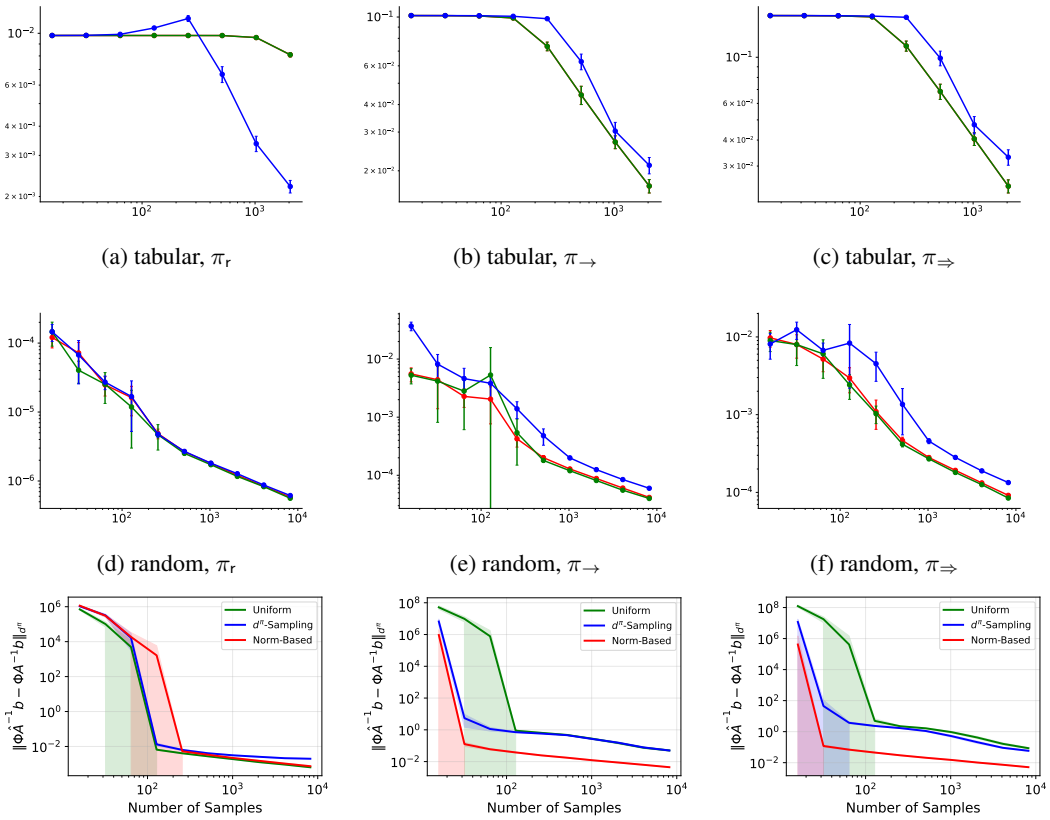
Figure 4: Matrix error $\|\hat{\mathbf{A}} - \mathbf{A}\|_F$ on the random chain across all three feature representations (rows: tabular, random, shared) and three policies (columns: π_r, $\pi_\rightarrow$, $\pi_\Rightarrow$). Each curve is averaged over 1,000 seeds with 99% confidence intervals; lower is better.

a smaller Frobenius error can reduce value error in this setting. They are not a direct check of Proposition 1, which bounds the regularized solution $(\hat{\mathbf{A}} + \lambda \mathbf{I})^{-1}\boldsymbol{b}$ for any reward vector that shares $\mathbf{A}$.

F.7 Four Rooms Heterogeneity Diagnostic

This appendix gives the replay-buffer diagnostic summarized in Section 4.1. The domain is the continuing 11×11 Four Rooms gridworld with $\gamma = 0.99$, the same domain as Appendix G, but the question is different. Here the reward is held fixed and only the feature representation changes, so that the product scores $u_i = \|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2$ become more or less heterogeneous. For each seed we collect a replay buffer of 5,000 transitions under a uniform-random behavior policy. The representations are Fourier features of orders 3, 5, and 7 (Konidaris et al., 2011); RBF features with $\sigma \in \{0.1, 0.2, 0.3\}$; polynomial features of degrees 2 and 3; and scaled Fourier features in which the four leading Fourier coordinates are multiplied by 10. The environment and the behavior policy stay fixed.

Figure 6 plots one point per feature configuration, averaged over 300 seeds. In each case $\mathbf{A}$ is estimated from a batch of B=1000 transitions. The horizontal axis is $\mathrm{CV}_{\mathcal{I}}(u)$, and the vertical axis is the Frobenius improvement of norm-based sampling over the baseline. The left panel compares against uniform replay. The right panel compares against PER with priorities $|\delta|^{0.6}$ from task-averaged oracle TD errors over K=8 goal-reaching tasks. In both panels the gain grows with $\mathrm{CV}_{\mathcal{I}}(u)$. Ordinary Fourier features are relatively homogeneous, and norm-based sampling has little room to improve on uniform sampling. Narrow RBFs, polynomial features, and rescaled Fourier



(a) tabular, π_{r} (b) tabular, $\pi_{\rightarrow}$ (c) tabular, $\pi_{\Rightarrow}$

(d) random, π_{r} (e) random, $\pi_{\rightarrow}$ (f) random, $\pi_{\Rightarrow}$

(g) shared, π_{r} (h) shared, $\pi_{\rightarrow}$ (i) shared, $\pi_{\Rightarrow}$

Figure 5: Value error $\|\Phi\hat{\mathbf{A}}^{-1}\boldsymbol{b} - \Phi\mathbf{A}^{-1}\boldsymbol{b}\|_{d^\pi}$ on the random chain across all three feature representations (rows: tabular, random, shared) and three policies (columns: π_{r}, $\pi_{\rightarrow}$, $\pi_{\Rightarrow}$). Each curve is averaged over 1,000 seeds with 99% confidence intervals; lower is better.

features are heterogeneous, and the improvement in $\hat{\mathbf{A}}$ is much larger. This comparison is only about $\mathbf{A}$. For a single sparse reward, estimating $\boldsymbol{b}$ from the same batch reverses the ranking, because value error is then dominated by the reward-dependent direction (Section G, Figures 7 and 8).

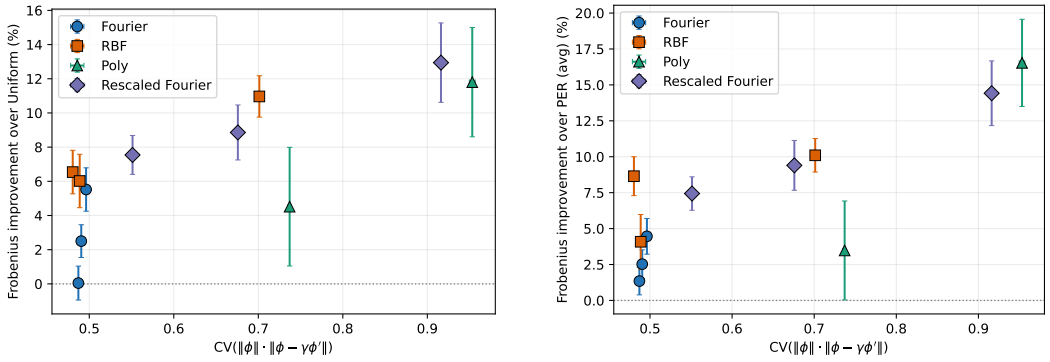


Figure 6: Heterogeneity diagnostic on Four Rooms. Each point is one feature configuration, and the horizontal axis is $\mathrm{CV}_{\mathcal{I}}(u)$ with $u_i = \|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2$. The vertical axis reports the Frobenius improvement of norm-based sampling over the baseline at batch size 1000, averaged over 300 seeds. Left: norm versus uniform. Right: norm versus PER with task-averaged oracle TD-error priorities over $K{=}8$ tasks.

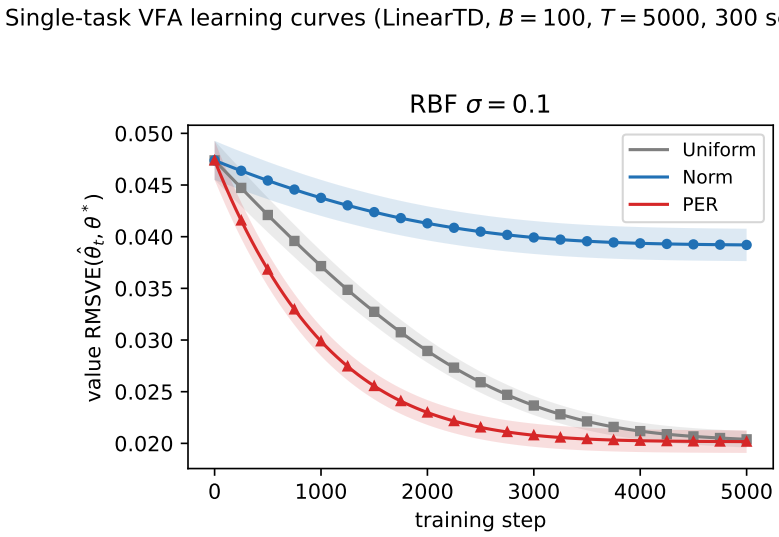


Figure 7: Single-task iterative TD on Four Rooms with RBF features. We run mini-batch linear TD with batch size $B = 100$ for 5,000 updates and report stationary-weighted RMSVE (15) over 300 seeds. Uniform and PER finish close to one another, while norm-based replay is substantially worse.

G Single-Task Setting: When Better $\hat{\mathbf{A}}$ Estimates no Longer Help

We now move our investigation from the controlled matrix-estimation setting to a standard replay-buffer policy-evaluation problem. The domain is Four Rooms (Sutton et al., 1999), an 11×11 grid-world with walls dividing the grid into four connected rooms. We use the continuing version of this task with discount $\gamma = 0.99$, and construct a replay buffer of 5,000 transitions collected by a uniform-random behavior policy. Each state is represented by its normalized two-dimensional location, encoded with radial basis function features; concretely a 5×5 grid of Gaussian functions with centers uniformly spread across $[0, 1]^2$ and with bandwidth $\sigma = 0.1$, resulting in a total of $d = 25$ features. We consider a single reward function, where the agent receives a reward of 1 when the next state is the bottom-right goal state, and 0 otherwise.

This experimental setting is useful since it introduces exactly the challenges not captured by the analyses in Section 4.1. In the model-based experiment, $\boldsymbol{b}$ was fixed, and $\hat{\mathbf{A}}$ was the only noisy quantity that needed to be estimated. In a replay-buffer value-estimation problem, on the other hand, both sides of the LSTD system must be estimated from a finite batch of data:

$$\hat{\mathbf{A}} \approx \mathbf{A}, \qquad \hat{\boldsymbol{b}} \approx \boldsymbol{b}.$$

First, recall that norm-based sampling is designed to minimize the variance of the estimator $\hat{\mathbf{A}}$. It is *not* designed to identify the reward-relevant transitions for a particular $\boldsymbol{b}$. In single-task settings, this distinction can be very important.

The comparison we perform now uses three samplers. The uniform replay strategy samples each transition with equal probability. Norm-based replay leverages priorities $p_i \propto \|\boldsymbol{\phi}_i\|_2\|\boldsymbol{\psi}_i\|_2$ and is reward-agnostic. Finally, PER uses priorities proportional to $|\delta_i|^{0.6}$, following the standard PER exponent proposed by Schaul et al. (2016). TD errors are computed using the oracle solution $\boldsymbol{\theta}^*$ in this single-reward setting, so the PER baseline is deliberately strong: it is reward-aligned and uses information that is not accessible to our norm sampling approach.

Figure 7 shows the result, for iterative TD. Value error is the stationary-weighted RMSVE of Eq. (15). Norm-based replay performs worse than both uniform replay and PER. This does not contradict the previous matrix-estimation results. Instead, it shows that reducing error in the estimate $\hat{\mathbf{A}}$ does not, by itself, guarantee that value function error will also be reduced when considering a *single* reward function. Intuitively, each TD also carries reward-side noise through the sampled targets. Since the reward function is sparse and localized near the goal, a reward-agnostic sampler

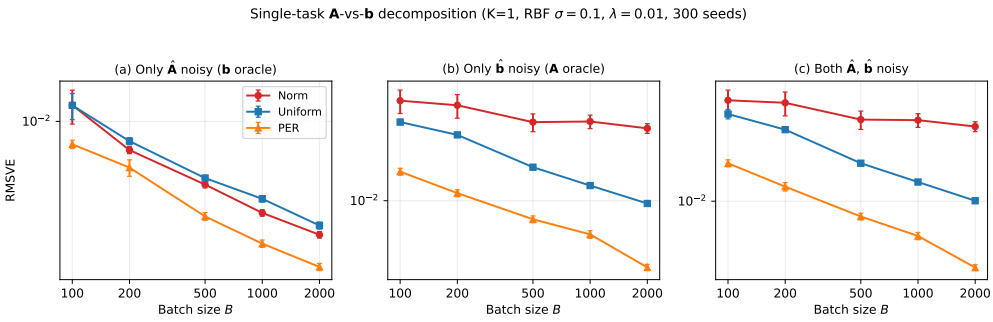


Figure 8: Single-task $\mathbf{A}$-versus-$\boldsymbol{b}$ decomposition on Four Rooms. Panel (a) keeps $\boldsymbol{b}_{\mathcal{B}}$ fixed and only varies $\hat{\mathbf{A}}$; panel (b) keeps $\mathbf{A}_{\mathcal{B}}$ fixed and only varies $\hat{\boldsymbol{b}}$; panel (c) estimates both from the same sampled batch. Curves report RMSVE (15) over 300 seeds with $\lambda = 10^{-2}$.

can spend a non-trivial part of its budget on transitions that are structurally important to estimate $\mathbf{A}$, but not particularly informative for estimating the (single particular) $\boldsymbol{b}$ vector of interest.

We next isolate the two possible sources of estimator error in a closed-form experiment. For each seed, let $\mathbf{A}_{\mathcal{B}}$ and $\boldsymbol{b}_{\mathcal{B}}$ be the estimates computed from all $5{,}000$ transitions in the buffer. We first draw a smaller batch of datapoints from each sampler, with B ranging from 100 to $2{,}000$, and use such information to compute $\hat{\mathbf{A}}$ and $\hat{\boldsymbol{b}}$. This results in three separate scenarios:

$$
\begin{aligned}
&(1) \qquad (\hat{\mathbf{A}} + \lambda \mathbf{I})\boldsymbol{\theta} = \boldsymbol{b}_{\mathcal{B}} && \text{(only } \hat{\mathbf{A}} \text{ is noisy)},\\
&(2) \qquad (\mathbf{A}_{\mathcal{B}} + \lambda \mathbf{I})\boldsymbol{\theta} = \hat{\boldsymbol{b}} && \text{(only } \hat{\boldsymbol{b}} \text{ is noisy)},\\
&(3) \qquad (\hat{\mathbf{A}} + \lambda \mathbf{I})\boldsymbol{\theta} = \hat{\boldsymbol{b}} && \text{(both } \hat{\mathbf{A}} \text{ and } \hat{\boldsymbol{b}} \text{ are noisy)},
\end{aligned}
$$

where $\lambda = 10^{-2}$. This ablation lets us investigate whether failures in single-task settings are caused primarily by noisy reward-vector estimates, or whether there is a deeper mismatch between the matrix estimation objective and the single-task value function estimation objective.

Figure 8 shows that there are two separate effects. First, when only $\hat{\mathbf{A}}$ is noisy, norm-based replay *does* improve the accuracy of the estimated value function, compared to uniform replay. This is due to the benefits of constructing a more accurate matrix from a lower-variance estimator, as discussed in Section 4.1, and the corresponding impact on value error. However, PER outperforms other samplers in the single-task setting across all the cases. Value error is not influenced equally by every direction of $\mathbf{A}$. It depends more on the directions that matter for this particular $\boldsymbol{b}$. PER is aligned with those reward-correlated directions, while norm-based sampling optimizes a task-agnostic Frobenius criterion.

Second, in cases where only $\hat{\boldsymbol{b}}$ is noisy, norm-based replay performs worse than uniform sampling. In this task, the reward signal is extremely sparse, so estimating $\boldsymbol{b}$ well relies heavily on sampling goal-adjacent transitions. Norm-based replay methods have no reason to prioritize those particular transitions, unless they happen to have large feature-transition norms. Once both $\hat{\mathbf{A}}$ and $\hat{\boldsymbol{b}}$ are estimated based on the batch of data, the resulting predicted value is dominated by this reward-vector channel, and the norm-based strategy underperforms compared to uniform samplers.

The conclusion of this experiment is that norm-based replay should not be expected to increase performance if we care *only* about learning the value function corresponding to a *single* reward function. Norm sampling by contrast, is a good sampler for estimating the shared structural object $\mathbf{A}$, while settings with a single reward function require high accuracy in a single reward-dependent direction. To deepen our investigation of such interactions, Section 4.2 studies samplers in the GVF setting; that is, in scenarios with many reward functions but where tasks share the same dynamics and features. In such cases, a single estimate of $\mathbf{A}$ can be reused across tasks, and the task-agnostic nature of norm-based sampling becomes an advantage rather than a liability.

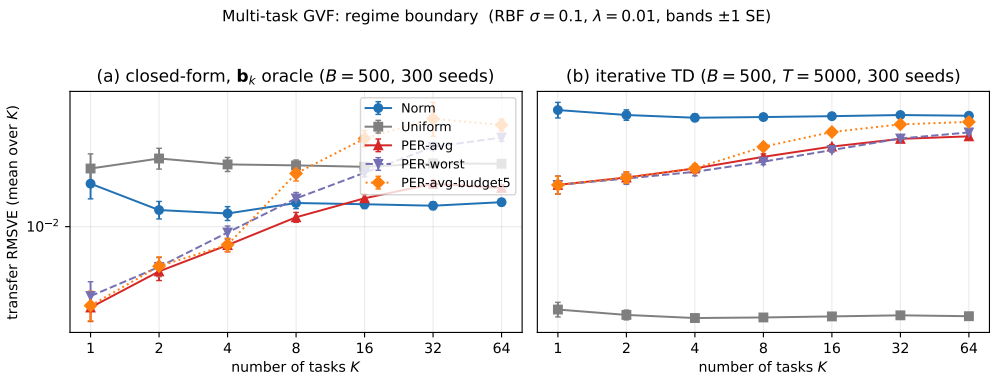


Figure 9: Multi-reward GVF experiment on Four Rooms. Left: closed-form transfer with $\boldsymbol{b}_k$ computed from the full buffer and $\hat{\mathbf{A}}$ estimated from a batch of $B = 500$ transitions. Right: iterative TD with $B = 500$ and 5,000 updates. In the closed-form regime, one shared matrix estimate is reused across tasks. The iterative regime carries sampled reward-side noise at every update. Curves report mean RMSVE over 300 seeds; lower is better.

H Multi-Reward GVF Curves

Figure 9 is the sweep summarized in Section 4.2. Both panels use the continuing Four Rooms domain with RBF features ($\sigma = 0.1$, $d = 25$), a replay buffer of 5,000 transitions, and a batch of B=500. The left panel is closed-form transfer: each $\boldsymbol{b}_k$ is computed from the full buffer, and only $\hat{\mathbf{A}}$ is estimated from the batch, with

$$(\hat{\mathbf{A}} + \lambda \mathbf{I})\hat{\boldsymbol{\theta}}_k = \boldsymbol{b}_k, \qquad \lambda = 10^{-2}.$$

The right panel is iterative TD for 5,000 updates. Each GVF has its own parameter vector, the same mini-batch indices are used for all K predictions, and each prediction uses its own cumulant. Curves are mean RMSVE over 300 seeds.

In closed-form transfer, $K = 1$ favors PER, consistent with the single-reward decomposition in Section G: the sampler is aligned with the one reward direction that matters. As K grows, that reward-specific signal is averaged over unrelated goals. Norm-based replay overtakes PER-avg at $K = 16$, and overtakes PER-worst and PER-avg-budget5 at $K = 8$. For $K \geq 16$, norm-based replay outperforms all three PER variants, and it remains better than uniform replay throughout.

Under iterative TD, a better $\hat{\mathbf{A}}$ does not by itself yield a better value function. Uniform replay is the most stable baseline across K, and norm-based replay remains about a factor of two worse. This matches the decomposition in Figure 8: when every update estimates task-specific rewards from the sampled batch, the $\hat{\boldsymbol{b}}_k$ channel can dominate. Norm-based sampling is not designed to identify rare rewarding transitions, so reward-side noise outweighs the gain from a more accurate shared matrix. Related analyses of importance sampling for SGD show that sampling proportional to gradient norms reduces variance for iterative updates (Needell et al., 2014; Zhao & Zhang, 2015; Katharopoulos & Fleuret, 2018; Lahire et al., 2022), which helps explain why TD-error prioritization can dominate when each mini-batch also carries the reward-dependent channel.

I Computational Details

The linear experiments were performed on a 24-core AMD Ryzen processor with a maximum frequency of 3.3 GHz and 32 GB RAM. The DQN control experiments were run on one node with eight H200 GPUs and took approximately 30 h of wall-clock time.

J Additional Control Figures

The DQN network in Section 5 is $\mathrm{Conv}(16, 3{\times}3) \to \mathrm{Dense}(128) \to \mathrm{Dense}(|\mathcal{A}|)$. The feature ϕ used by norm-ϕ is the 128-dimensional ReLU output of the penultimate layer under the target network; the observation and last-layer-gradient choices of f replace this representation in the same priority product. Priorities are refreshed every 10,000 environment steps. PER uses $\alpha \in \{0.4, 0.6, 0.8\}$, with β annealed from 0.4 to 1.0.

Importance weights and failure settings. PER applies max-normalized importance weights. The norm-ϕ curves in Figure 2 do not. Figure 15 applies full max-normalized importance correction ($\beta = 1$ from the first update) to every norm variant (30 seeds, 1M steps): the moderate-batch return advantage disappears on Asterix and Space Invaders, while the large-batch Freeway result remains. Concretely, Asterix norm-grad at batch 32 falls from 16.21 to 13.83, tying uniform; Space Invaders norm-obs at batch 32 falls from 38.15 to 35.85; at Freeway batch 64, corrected norm-ϕ is 25.6 and norm-grad is 36.9, against uniform 11.6. Successor-feature residuals are largely unchanged by the correction. Two further settings in which the sampler does not help are a replay buffer of 10^4 rather than 10^5, and auxiliary heads that are not detached from the trunk.

Figure 10 is the eight-sampler return sweep at a shared learning rate of $2.5{\times}10^{-4}$, for batches 1–64 and horizons 250k, 500k, and 1M. Selecting each family at its best cell after 1M steps gives Asterix $16.00{\pm}0.67$ (norm-grad, batch 32), against uniform $13.10{\pm}0.56$ and the best PER cell $12.59{\pm}0.60$; Freeway $46.72{\pm}0.22$ (norm-ϕ, batch 4), against uniform $44.95{\pm}0.22$; Space Invaders $38.25{\pm}1.28$, against uniform 36.07 ± 1.15; and a tie on Breakout. At batch 64, uniform return on Freeway is 12.06 ± 1.11 while norm-ϕ holds 39.38 ± 0.89, and PER with $\alpha = 0.8$ falls to 3.03 ± 0.21 on Asterix. An independent 30-seed re-run reproduced these cells within 3σ.

Figure 11 is the $f = \phi$ priority histogram at random initialization, the diagnostic summarized in Section 5. Asterix and Freeway are strongly skewed; Breakout is nearly uniform. The same ϕ-Gini at the end of training, measured over 30 seeds of 1M-step DQN with uniform replay, is 0.360 (Asterix), 0.519 (Freeway), 0.330 (Space Invaders), and 0.239 (Breakout). Figure 12 adds the observation and gradient choices of f.

Figure 13 is the full GVF-probe sweep, and Figure 14 is the successor-feature sweep. On the successor-feature probe, norm-ϕ is best on Breakout at every batch, for example 7.48 ± 0.80 at batch 1 against uniform 22.0 ± 2.9, while PER is lower on many Freeway cells. Reducing the replay buffer from 10^5 to 10^4 (10 seeds, 250k steps) removes most of that successor-feature gap. Closed-form LSTD heads on these deep features (Levine et al., 2017; Chung et al., 2019) are 5–50$\times$ worse than the SGD heads (30 seeds, 500k steps). Auxiliary heads that are not detached from the trunk reduce Freeway return from 28.48 to 0.45 (10 seeds, 250k steps), which is why Section 5 uses detached probes.

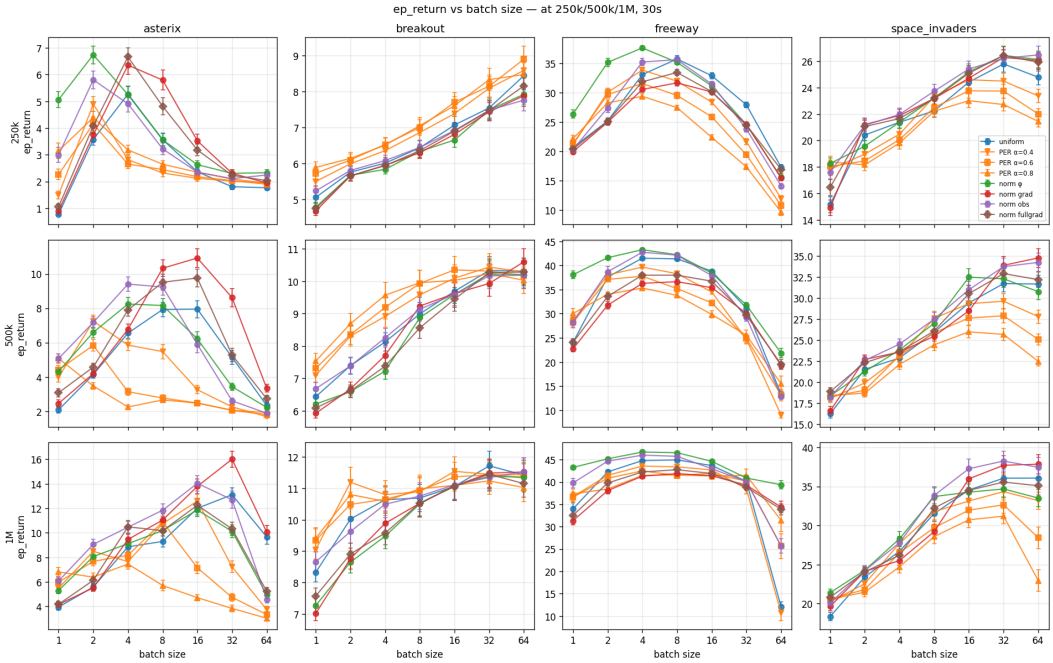


Figure 10: Final-quarter DQN return versus batch size for every game and sampler, at 250k, 500k, and 1M environment steps (30 seeds, shared learning rate 2.5×10^{-4}). This is the complete sweep summarized in Section 5.

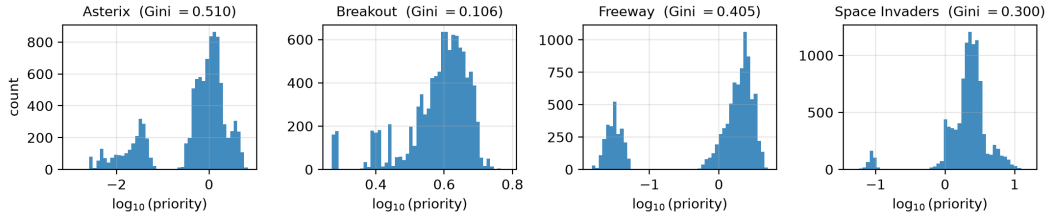


Figure 11: Histograms of norm-ϕ priorities ($f = \phi$), $p_i \propto \|f_i\|_2 \|f_i - \gamma(1-\tau_i) f_i'\|_2$, under a single randomly initialized network (10,000 random-policy transitions per game). Asterix and Freeway are strongly heterogeneous; Breakout is nearly uniform. The same ordering appears in the return comparison of Figure 2. Histograms for the observation and gradient choices of f are Figure 12.

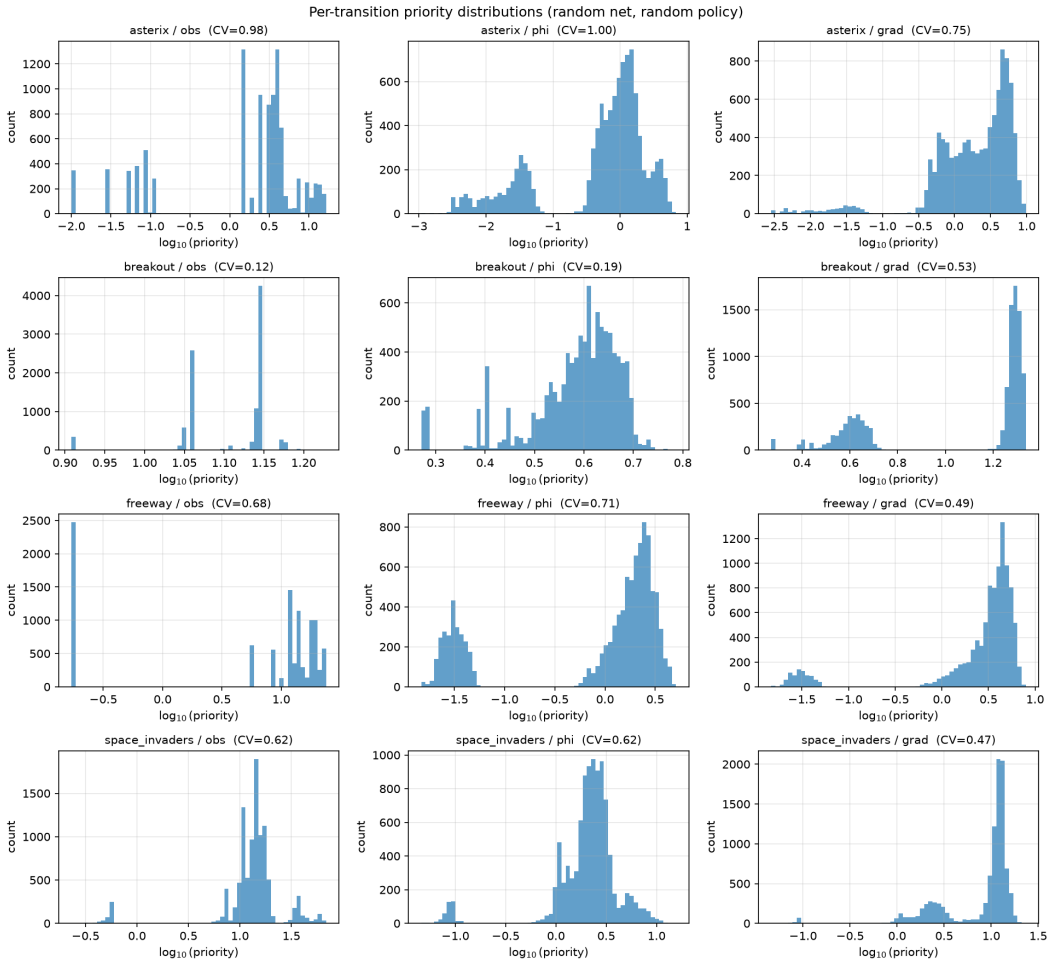


Figure 12: Per-transition norm-based priority histograms under a single randomly initialized network and a random policy (10,000 transitions per game), for four choices of f: observation, penultimate trunk ϕ, last-layer gradient, and parameter-gradient norm (fullgrad). Figure 11 is the $f = \phi$ column.

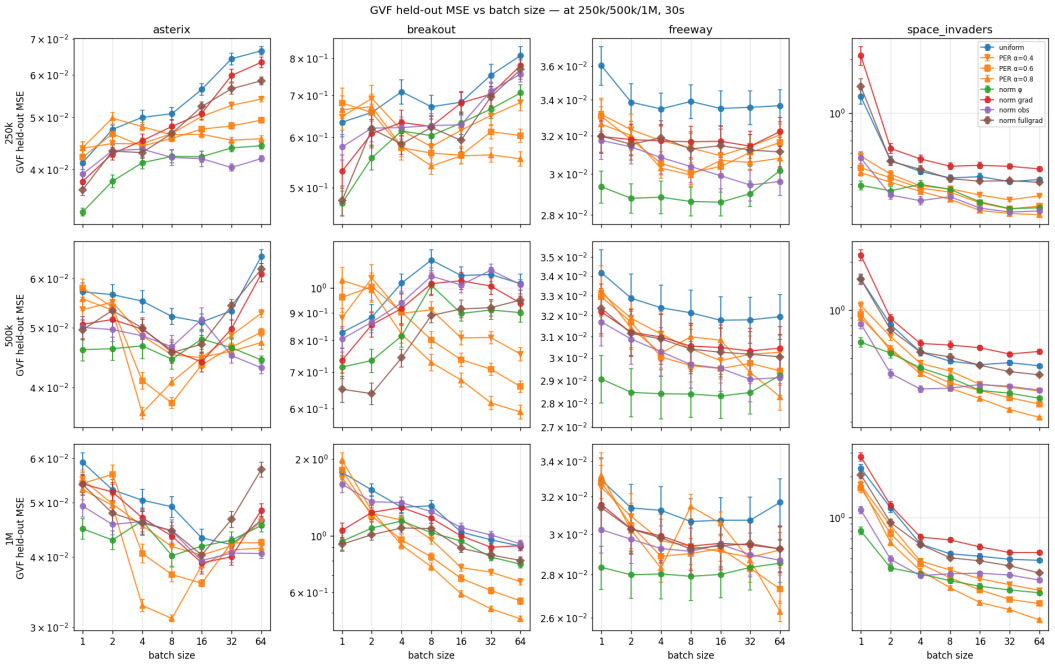


Figure 13: Detached GVF Bellman residual (final quarter, 1M steps, 30 seeds) versus batch size for every game and sampler. Figure 2 shows the same comparison for uniform, PER with $\alpha = 0.6$, and norm-ϕ.

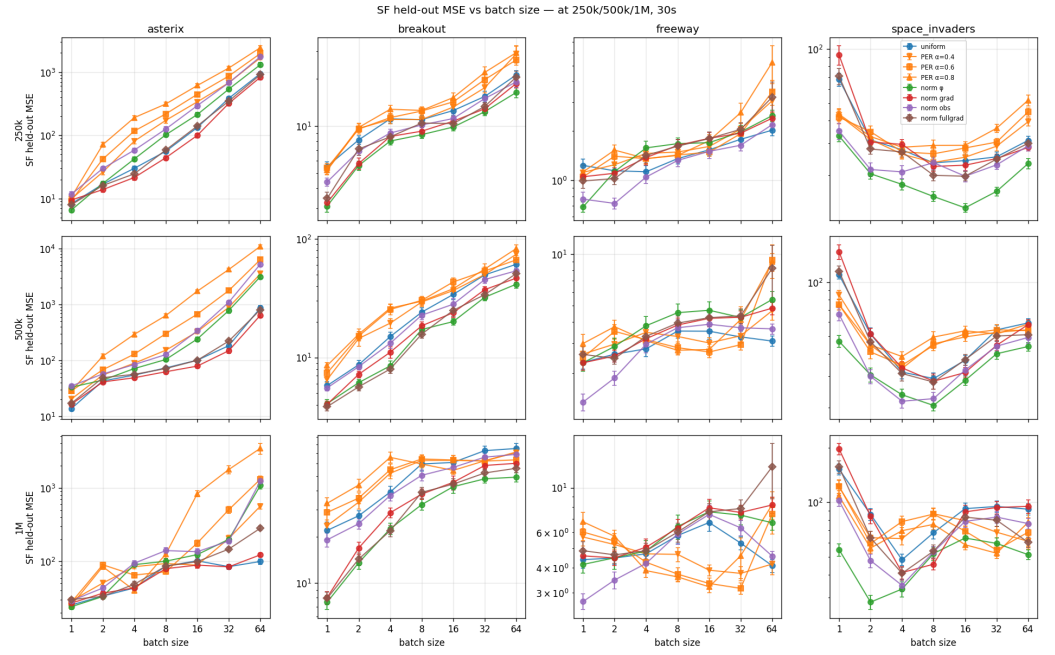


Figure 14: Detached successor-feature Bellman residual (final quarter, 1M steps, 30 seeds) versus batch size. This is a replay-buffer TD-loss diagnostic on fixed buffer indices, not an independent Monte Carlo evaluation. Norm-ϕ is best on Breakout at every batch; PER wins many Freeway cells; high-α PER degrades with batch size on Asterix.

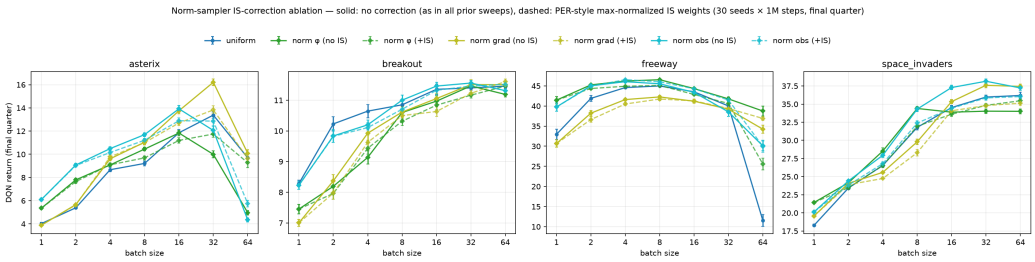


Figure 15: Return versus batch size for each norm variant with full importance correction (dashed) and without it (solid), 30 seeds and 1M steps. Uniform is shown as a reference. Correction removes the moderate-batch return edges on Asterix and Space Invaders; large-batch robustness on Freeway remains.