Posterior Coreset Construction with Kernelized Stein Discrepancy for Model-Based Reinforcement Learning

Souradip Chakraborty[1], **Amrit Bedi**[1], **Alec Koppel**[2], **Pratap Tokekar**[1], **Furong Huang**[1], **Dinesh Manocha**[1]
[1]University of Maryland, College Park, [2]JP Morgan AI Research, NY, USA,

Abstract

Model-based reinforcement learning (MBRL) exhibit favorable performance in practice, but their theoretical guarantees are mostly restricted to the setting when transition model is Gaussian or Lipschitz, and demands a posterior estimate whose representational complexity grows unbounded with time. In this work, we develop a novel MBRL method (i) which relaxes the assumptions on the target transition model to belong to a generic family of mixture models; (ii) is applicable to large-scale training by incorporating a compression step such that the posterior estimate consists of a *Bayesian coreset* of only statistically significant past state-action pairs; and (iii) exhibits a Bayesian regret of $\mathcal{O}(dH^{1+(\alpha/2)}T^{1-(\alpha/2)})$ with coreset size of $\Omega(\sqrt{T^{1+\alpha}})$, where d is the aggregate dimension of state action space, H is the episode length, T is the total number of time steps experienced, and $\alpha \in (0, 1]$ is the tuning parameter which is a novel introduction into the analysis of MBRL in this work. To achieve these results, we adopt an approach based upon Stein's method, which allows distributional distance to be evaluated in closed form as the kernelized Stein discrepancy (KSD). Experimentally, we observe that this approach is competitive with several state of the art RL methodologies, and can achieve up-to 50% reduction in wall clock time in some continuous control environments.

1 Problem Formulation

We consider the problem of modelling an episodic finite-horizon Markov Decision Process (MDP) where the true unknown MDP is defined as $M^* := \{\mathcal{S}, \mathcal{A}, R^*, P^*, H, R_{\max}, \rho\}$, where $\mathcal{S} \subset \mathbb{R}^{d_s}$ and $\mathcal{A} \subset \mathbb{R}^{d_a}$ denote continuous state and action spaces, respectively. Here, P^* represents the true underlying generating process for the state action transitions and R^* is the true rewards distribution. After every episode of length H, the state will reset according to the initial state distribution ρ. At time step $i \in [1, H]$ within an episode, the agent observe $s_i \in \mathcal{S}$, selects $a_i \in \mathcal{A}$ according to a policy μ, receives a reward $r_i \sim R^*(s_i, a_i)$ and transitions to a new state $s_{i+1} \sim P^*(\cdot|s_i, a_i)$. We consider M^* itself as a random process, as is the often the case in Bayesian Reinforcement Learning, which helps us to distinguish between the true and fitted transition/reward model. Next, we define policy μ as a mapping from state $s \in \mathcal{S}$ to action $a \in \mathcal{A}$ over an episode of length H. For a given MDP M, the value for time step i is the reward accumulation during the episode $V^M_{\mu,i}(s) = \mathbb{E}[\Sigma^H_{j=i}[\bar{r}^M(s_j, \mu(s_j, j))|s_i = s]$where actions are under policy $\mu(s_j, j)$ (j denotes the timestep within the episode) and $\bar{r}^M(s, a) = \mathbb{E}_{r\sim R^M(s,a)}[r]$. Without loss of generality, we assume the expected reward an agent receives at a single step is bounded $|\bar{r}^M(s, a)| \leq R_{\max}$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$. This further implies that $|V(s)| \leq HR_{\max}$, $\forall s$. For a given MDP M, the optimal policy μ^M is defined as $\mu^M = \arg\max_\mu V^M_{\mu,i}(s)$ for all s and $i = 1, \ldots, H$. Next, we also define future value function $U^M_i(P)$ to be the expected value of the value function over all initializations and trajectories $U^M_i(P) = \mathbb{E}_{s'\sim P(s'|s,a),a=\mu^M(s,i)}[V^M_{\mu^M,i}(s')|s_i = s]$ where P is the transition distribution under

MDP M. According to these definitions, one would like to find the optimal policy (**??**) for the true model $M = M^*$. Next, we review the PSRL algorithm, which is an adaption of Thompson sampling to RL [41] (see Algorithm 1 in Appendix B). In PSRL, we start with a prior distribution over MDP given by ϕ. Then at each episode, we take sample M^k from the posterior given by $\phi(\cdot|\mathcal{D}_k)$, where $\mathcal{D}_k := \{s_{1,1}, a_{1,1}, r_{1,1}, \cdots , s_{k-1,H}, a_{k-1,H}, r_{k-1,H}\}$ is a data set containing past trajectory data, i.e., state-action-reward triples, which we call a *dictionary*. That is, where $s_{k,i}, a_{k,i}$ and $r_{k,i}$ indicate the state, action, and reward at time step zi in episode k. Then, we evaluate the optimal policy $\mu^k := \mu^{M^k}$. Thereafter, information from the latest episode is appended to the dictionary as $\mathcal{D}_{k+1} = \mathcal{D}_k \cup \{s_{k,1}, a_{k,1}, r_{k,1}, \cdots , s_{k,H}, a_{k,H}, r_{k,H}\}$.

Bayes Regret and Limitations of PSRL: The Bayes regret of PSRL (cf. Algorithm 1) is established to be $\tilde{\mathcal{O}}(\sqrt{d_K(R)d_E(R)T} + \mathbb{E}[L^*]\sqrt{d_K(P)d_E(P)})$ where d_K and d_E are Kolmogorov and Eluder dimensions, R and P refer to function classes of rewards and transitions, and L^* is a global Lipschitz constant for the future value function. Although it is mentioned that system noise smooths the future value functions in [41], an explicit connection between H and L is absent, which leads to an exponential dependence on horizon length H in the regret [41, Corollary 2] for LQR.

A crucial assumption in deriving the best known regret bound for PSRL in continuous control is of target distribution belonging to Gaussian/symmetric class, which is often violated. For instance, if we consider a variant of inverted pendulum with an articulated arm, the transition model has at least as many modes as there are minor-joints in the arm. Another major challenge is related to *posterior's parameterization complexity* $M(T) := |\mathcal{D}_k|$, which we subsequently call the dictionary size (step 10 in Algorithm 1) which is used to parameterize the posterior distribution. We note that $M(T) = \Omega(T)$ for the PSRL [41] and MPC-PSRL [19] algorithms which are state of the art approaches.

To alleviate the Gaussian restriction, we consider an alternative metric of evaluating the distributional estimation error, namely, the kerneralized Stein discrepancy (KSD). Additionally, that KSD is easy to evaluate under appropriate conditions on the target distribution, i.e., the target distribution is smooth, one can compare its relative quality as a function of which data is included in the posterior. Doing so allows us to judiciously choose which points to retain during the learning process in order to ensure small Bayesian regret. To our knowledge, this work is the first to deal with the compression of posterior estimate in model-based RL settings along with provable guarantees. These aspects are derived in detail in the following section.

2 Proposed Approach

2.1 Posterior Coreset Construction via KSD

The core of our algorithmic development is based upon the computation of Stein kernels and KSD to evaluate the merit of a given transition model estimate, and determine which past samples to retain. Doing so is based upon the consideration of Stein based integral probability metrics (IPM) rather than total variation (TV) distance which allows us to employ Stein's identity, which under a hypothesis that the score function of the target is computable. This approach is well-known to yield methods to improve the sample complexity of Markov Chain Monte Carlo (MCMC) methods [13]. This turns out to be the case in model-based RL as well is a testament to its power [49, 47]. This method to approximate a target density P consists of defining an IPM [48] based on a set $\mathcal{G}$ consisting of test functions on $\mathcal{X} \subset \mathbb{R}^{2d_s+d_a}$, and is defined as $D_{\mathcal{G},P}(\{x_i\}_{i=1}^n) := \sup_{g\in\mathcal{G}} \left|\frac{1}{n}\sum_{i=1}^n g(x_i) - \int_{\mathcal{X}} g\mathrm{d}P\right|$

Although IPMs efficiently quantify the discrepancy between an estimate and target, requires P to evaluate the integral, which may be unavailable. To alleviate this issue, Stein's method restricts the class of test functions g to be those such that $\mathbb{E}_P[g(z)] = 0$. In this case, IPM only depends on the Dirac-delta measure (δ) from the stream of samples, removing dependency on the exact integration in terms of P. Under certain smoothness assumptions on the posterior, the IPM can be evaluated in closed form through the kernelized Stein discrepancy (KSD). The key technical upshot of employing Stein's method is that we can now evaluate the integral probability metric of posterior $\phi_{\mathcal{D}_k} := \phi(\cdot|\mathcal{D}_k)$ parameterized by dictionary $\mathcal{D}_k$ through the KSD, which is efficiently computable: $\text{KSD}(\phi_{\mathcal{D}_k}) := \sqrt{\frac{1}{|\mathcal{D}_k|^2}\sum_{h_i,h_j}\kappa_0(h_i,h_j)}$ which depends solely on score function of the estimated posterior and we no longer require access to the true unknown target transition model of the MDP.

This is a major merit of utilizing Stein's method in MBRL, and allows us to improve the regret of model-based RL methods based on posterior sampling. This the previous point is distinct from the computational burden of storing dictionary $\mathcal{D}_k$ that parameterizes $\phi_{\mathcal{D}_k}$ and evaluating the optimal value function according to the current belief model. After this novel change (see Lemma **??** in Sec. 3), we can utilize the machinery of KSD to derive the regret rate for the proposed algorithm in this work (cf. Algorithm 2) in lieu of concentration inequalities, as in [19]. We shift to the computational storage requirements of the posterior in continuous space next.

KSD Thinning: We develop a principled way to avoid the requirement that the dictionary $\mathcal{D}_k$ retains all information from past episodes, and is instead parameterized by a coreset of statistically significant samples. The issue in PSRL (or MBRL in general) is as the number of episodes experienced becomes large, the posterior representational complexity grows linearly and unbounded with episode index k. On top of that, the posterior update in PSRL (cf. Algorithm 1) is also parameterized by data collected in $\mathcal{D}_{k+1}$ (ex : Gaussian processes.[46]). To deal with this bottleneck, we propose to sequentially remove those particles from $\mathcal{D}_{k+1}$ that contribute least in terms of KSD. It can be interpreted as projecting posterior estimates onto "subspaces" spanned by only statistically representative past state-action-state triples. Such nonparametric posterior representation has been shown to exhibit theoretical and numerical advantages in probability density estimation [11, 12], Gaussian Processes [32], and Monte Carlo methods [18]. Here we introduce it for the first time in model-based RL, which allows us to control the growth of the posterior complexity, which in turn permits us to obtain computationally efficient updates.

α	$\mathbb{E}[\text{Regret}_T]$	$M(T)$
0	$\mathcal{O}(dHT)$	$\tilde{\Omega}(\sqrt{T})$
0.5	$\mathcal{O}(dH^{\frac{7}{4}}T^{\frac{1}{4}})$	$\tilde{\Omega}(T^{3/4})$
1	$\mathcal{O}(dH^{\frac{3}{2}}T^{\frac{1}{2}})$	$\tilde{\Omega}(T)$

Table 1: Tradeoff for different values of α.

To be more specific, suppose we are at episode k with dictionary $\mathcal{D}_k$ associated with posterior $\phi_{\mathcal{D}_k}$, and we denote the dictionary after update as $\widetilde{\mathcal{D}}_{k+1} = \mathcal{D}_k + H$ and corresponding posterior as $\phi_{\widetilde{\mathcal{D}}_{k+1}}$. For a given dictionary $\mathcal{D}_k$, we can calculate the KSD of posterior $\phi_{\mathcal{D}_k}$ to target. We note that the KSD estimate goes to zero as $k \to \infty$ due to the posterior consistency conditions [22]. At each episode k, after performing the dictionary update to obtain $\widetilde{\mathcal{D}}_{k+1}$ (step 10 in Algorithm 2), we propose to thin dictionary $\widetilde{\mathcal{D}}_{k+1}$ such that $\text{KSD}(\phi_{\mathcal{D}_{k+1}})^2 < \text{KSD}(\phi_{\widetilde{\mathcal{D}}_{k+1}})^2 + \epsilon_{k+1}$ where $\mathcal{D}_{k+1}$ is the dictionary following thinning and $\epsilon_k > 0$ is a scalar parameter we call the thinning budget proposed. This means the posterior defined by compressed dictionary $\phi_{\widetilde{\mathcal{D}}_{k+1}}$ is at most ϵ_{k+1} in KSD from its uncompressed counterpart. We will see how ϵ_k permits us to trade off regret and dictionary size in practice. KSD may be succinctly stated as

$$(\phi_{\mathcal{D}_{k+1}}, \mathcal{D}_{k+1}) = \text{KSD-Thinning}(\phi_{\widetilde{\mathcal{D}}_{k+1}}, \widetilde{\mathcal{D}}_{k+1}, \epsilon_k). \tag{1}$$

We summarize the proposed algorithm in Algorithm 2 with compression subroutine in Algorithm 3 in Appendix, where `KSRL` is an abbreviation for Kernelized Stein Discrepancy Thinning for Model-Based Reinforcement Learning. Please refer to discussion Appendix B for MPC based action selection.

3 Bayesian Regret Analysis

In this section, we develop the Bayesian Regret Analysis for `KSRL`using score-based IPM that exploit's salient structural properties of Stein's method, and additionally provides a basis for establishing tradeoffs between regret and posterior representational complexity which is novel in this work. In the first step, we upper bound the future value function estimation error of P^k with respect to P^* as $U_i^k(P^k(\hat{h}_i)) - U_i^k(P^*(\hat{h}_i)) \leq HR_{\max}\text{KSD}(P^k(h_i))$ for all i and k. The salient point to note here is that we do not require access to P^* to evaluate the RHS of the equation. Then, we prove that $\mathbb{E}\left[\text{KSD}(\phi_{\mathcal{D}_k})\right] = \mathcal{O}\left(\frac{\sqrt{k \log(k)}}{f(k)}\right)$ where $\epsilon_k = \frac{\log(k)}{f(k)^2}$ is the thinning budget. For the statistical consistency of the posterior estimate, we note that it is sufficient to show that $\mathbb{E}\left[\text{KSD}(\phi_{\mathcal{D}_k})\right] \to 0$ as $k \to \infty$, which imposes a lower bound on the dictionary size growth rate $f(k) > \sqrt{k \log(k)}$ required for convergence. This result communicates that it is possible to achieve statistical consistency without having a linear growth in the dictionary size that is a drawback of prior art [41, 19]. Finally, with $\epsilon_k = \frac{\log(k)}{f(k)^2}$ and coreset size growth condition $f(k) = \sqrt{k^{\alpha+1}\log(k)}$ where $\alpha \in (0, 1]$, we prove total Bayes regret for `KSRL`is given by $\mathbb{E}[\text{Regret}_T] = \mathcal{O}\left(dT^{1-\frac{\alpha}{2}}H^{1+\frac{\alpha}{2}}\right)$ with coreset size $M(T) = \tilde{\Omega}\left(\sqrt{T^{1+\alpha}}\right)$. Observe that we match the best known prior results of PSRL with $\alpha = 1$ as shown in [19] in-terms of $O(dH^{\frac{3}{2}}T^{\frac{1}{2}})$, but with relaxed conditions on the posterior, allowing the approach to apply to a much broader class of problems. Moreover, for $\alpha < 1$, we obtain a superior tradeoff in model complexity and regret.

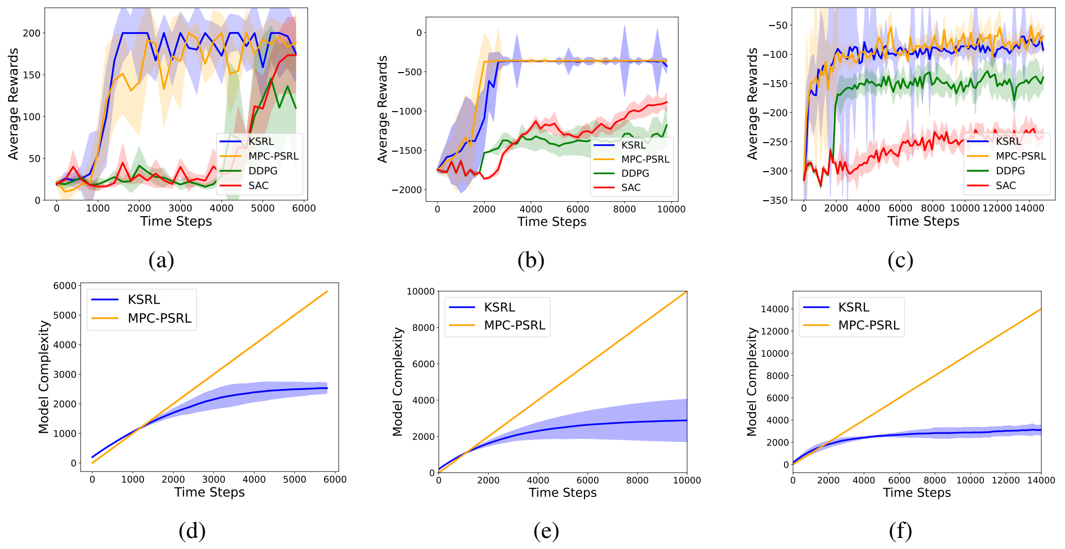


Figure 1: **(a)-(c)** compares the average cumulative reward return achieved by the proposed `KSRL` (shown in blue) algorithm with MPC-PSRL [19], SAC [23], and DDPG [5] for **Cartpole**, **Pendulum**, and **Pusher** without rewards. Fig. 4 in the Appendix F shows with rewards. **(d)-(f)** compares the model-complexity. We note that `KSRL` is able to achieve the maximum average reward at-par with the current SOTA MPC-PSRL with drastically reduced model complexity. Solid curves represent the average across five trials (seeds), shaded areas correspond to the standard deviation amongst the trials.

4 Experiments

We present a detailed experimental analysis of `KSRL` comparing to state of the art model-based and model-free RL methods on several continuous control tasks in terms of training rewards, model complexity and KSD convergence. For model-based we compare to MPC-PSRL method proposed in [19] since MPC-PSRL is already shown to outperform MBPO [27] and PETS [15] in this environmental settings. For comparison to model-free approaches, we compare with Soft Actor-Critic (SAC) from [23] and Deep Deterministic Policy Gradient (DDPG) [5]. We consider continuous control environments Stochastic Pendulum , Continuous Cartpole, Reacher and Pusher with and without rewards of modified OpenAI Gym [8] & MuJoCo environments [55]. See Appendix F for additional specific details of the environments and architecture.

Discussion. Fig. 2 compares the average reward return (top row) and model complexity (bottom row) for Cartpole, Pendulum, and Pusher, respectively. We note that `KSRL` performs equally good or even better as compared to the state of the art MPC-PSRL algorithm with significant reduction in model complexity (bottom row in Fig. 2) consistently across different environments. From the model complexity plots, we remark that `KSRL` is capable of automatically selecting the data points and control the dictionary growth across different environments which helps to achieve same performance in terms of average reward with fewer dictionary points to parameterize posterior distributions. This also helps in achieving faster compute time in practice to perform the same task as detailed in Fig. 3 in the Appendix F. We also show improvements of our algorithm over MPC with fixed buffer size as shown in Fig. 8 Appendix F with both random and sequential removal.

References

[1] Shipra Agrawal and Randy Jia. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In *Advances in Neural Information Processing Systems*, pages 1184–1194,

2017.

[2] Philip Amortila, Doina Precup, Prakash Panangaden, and Marc G Bellemare. A distributional analysis of sampling-based reinforcement learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 4357–4366. PMLR, 2020.

[3] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1):5–43, 2003.

[4] Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin Yang. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pages 463–474. PMLR, 2020.

[5] Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

[6] Vivek S Borkar and Sean P Meyn. Risk-sensitive optimal control for markov decision processes with monotone cost. *Mathematics of Operations Research*, 27(1):192–209, 2002.

[7] Zdravko I Botev, Dirk P Kroese, Reuven Y Rubinstein, and Pierre L'Ecuyer. The cross-entropy method for optimization. In *Handbook of statistics*, volume 31, pages 35–59. Elsevier, 2013.

[8] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[9] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013.

[10] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[11] Trevor Campbell and Tamara Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. In *International Conference on Machine Learning*, pages 698–706. PMLR, 2018.

[12] Trevor Campbell and Tamara Broderick. Automated scalable bayesian inference via hilbert coresets. *The Journal of Machine Learning Research*, 20(1):551–588, 2019.

[13] Wilson Ye Chen, Alessandro Barp, François-Xavier Briol, Jackson Gorham, Mark Girolami, Lester Mackey, and Chris Oates. Stein point markov chain monte carlo. In *International Conference on Machine Learning*, pages 1011–1021. PMLR, 2019.

[14] Sayak Ray Chowdhury and Aditya Gopalan. Online learning in kernelized markov decision processes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3197–3205, 2019.

[15] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, pages 4754–4765, 2018.

[16] Christoph Dann, Tor Lattimore, and Emma Brunskill. Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[17] Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–423, 2013.

[18] Víctor Elvira, Joaquín Míguez, and Petar M Djurić. Adapting the number of particles in sequential monte carlo methods through an online scheme for convergence assessment. *IEEE Transactions on Signal Processing*, 65(7):1781–1794, 2016.

[19] Ying Fan and Yifei Ming. Model-based reinforcement learning for continuous control with posterior sampling. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3078–3087. PMLR, 18–24 Jul 2021.

[20] Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

[21] Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015.

[22] Jackson Gorham and Lester Mackey. Measuring sample quality with stein's method. *Advances in Neural Information Processing Systems*, 28, 2015.

[23] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[24] Cole Hawkins, Alec Koppel, and Zheng Zhang. Online, informative mcmc thinning with kernelized stein discrepancy. *arXiv preprint arXiv:2201.07130*.

[25] Cole Hawkins, Alec Koppel, and Zheng Zhang. Online, informative mcmc thinning with kernelized stein discrepancy, 2022.

[26] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(51):1563–1600, 2010.

[27] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

[28] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4863–4873, 2018.

[29] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143, 2020.

[30] Sanket Kamthe and Marc Deisenroth. Data-efficient reinforcement learning with probabilistic model predictive control. In *International conference on artificial intelligence and statistics*, pages 1701–1710. PMLR, 2018.

[31] Sudheesh Kumar Kattumannil. On stein's identity and its applications. *Statistics & Probability Letters*, 79(12):1444–1449, 2009.

[32] Alec Koppel, Hrusikesha Pradhan, and Ketan Rajawat. Consistent online gaussian process regression without the sample complexity bottleneck. *Statistics and Computing*, 31(6):1–18, 2021.

[33] Umit Kose and Andrzej Ruszczynski. Risk-averse learning by temporal difference methods with markov risk measures. *Journal of machine learning research*, 22, 2021.

[34] Jonathan Lee, Aldo Pacchiano, Vidya Muthukumar, Weihao Kong, and Emma Brunskill. Online model selection for reinforcement learning with function approximation. In *International Conference on Artificial Intelligence and Statistics*, pages 3340–3348. PMLR, 2021.

[35] Qiang Liu, Jason Lee, and Michael Jordan. A kernelized stein discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284. PMLR, 2016.

[36] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[37] Trung Nguyen, Zhuoru Li, Tomi Silander, and Tze Yun Leong. Online feature selection for model-based reinforcement learning. In *International Conference on Machine Learning*, pages 498–506. PMLR, 2013.

[38] Brendan O'Donoghue. Variational bayesian reinforcement learning with regret bounds. *Advances in Neural Information Processing Systems*, 34, 2021.

[39] Ian Osband, Van Roy Benjamin, and Russo Daniel. (More) efficient reinforcement learning via posterior sampling. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3003–3011, USA, 2013. Curran Associates Inc.

[40] Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems*, 26, 2013.

[41] Ian Osband and Benjamin Van Roy. Model-based reinforcement learning and the eluder dimension. In *Advances in Neural Information Processing Systems*, pages 1466–1474, 2014.

[42] Ian Osband and Benjamin Van Roy. Why is posterior sampling better than optimism for reinforcement learning? In *International conference on machine learning*, pages 2701–2710. PMLR, 2017.

[43] Ian Osband and Benjamin Van Roy. Why is posterior sampling better than optimism for reinforcement learning? In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, pages 2701–2710, International Convention Centre, Sydney, Australia, 2017. PMLR.

[44] Yi Ouyang, Mukul Gagrani, and Rahul Jain. Control of unknown linear systems with thompson sampling. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1198–1205. IEEE, 2017.

[45] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[46] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. Springer, 2004.

[47] Nathan Ross. Fundamentals of stein's method. *Probability Surveys*, 8:210–293, 2011.

[48] Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert RG Lanckriet. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599, 2012.

[49] Charles Stein et al. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley symposium on mathematical statistics and probability*, volume 1, pages 197–206, 1956.

[50] Alexander L Strehl, Lihong Li, and Michael L Littman. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 10(11), 2009.

[51] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[52] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[53] Georgios Theocharous, Zheng Wen, Yasin Abbasi Yadkori, and Nikos Vlassis. Scalar posterior sampling with applications. *Advances in Neural Information Processing Systems*, 31, 2018.

[54] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

[55] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[56] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[57] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[58] Ziping Xu and Ambuj Tewari. Reinforcement learning in factored mdps: Oracle-efficient algorithms and tighter regret bounds for the non-episodic setting. *Advances in Neural Information Processing Systems*, 33:18226–18236, 2020.

[59] Lin Yang and Mengdi Wang. Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound. In *International Conference on Machine Learning*, pages 10746–10756. PMLR, 2020.

[60] Andrea Zanette, Alessandro Lazaric, Mykel Kochenderfer, and Emma Brunskill. Learning near optimal policies with low inherent bellman error. In *International Conference on Machine Learning*, pages 10978–10989. PMLR, 2020.

[61] Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Conference on Learning Theory*, pages 4532–4576. PMLR, 2021.

Supplementary Material for
“Posterior Coreset Construction with Kernelized Stein Discrepancy for Model-Based Reinforcement Learning"

A Introduction

Reinforcement learning, mathematically characterized by a Markov Decision Process (MDP) [45], has gained traction for addressing sequential decision-making problems with long-term incentives and uncertainty in state transitions [52]. A persistent debate exists as to whether model-free (approximate dynamic programming [51] or policy search [57]), or model-based (model-predictive control, MPC [20, 30]) methods, are superior in principle and practice [56]. A major impediment to settling this debate is that performance certificates are presented in disparate ways, such as probably approximate correct (PAC) bounds [50, 16], frequentist regret [28, 29], Bayesian regret [1, 58, 38], and convergence in various distributional metrics [6, 2, 33]. In this work, we restrict focus to regret, as it imposes the fewest requirements on access to a generative model underlying state transitions.

In evaluating the landscape of frequentist regret bounds for RL methods, both model-based and model-free approaches have been extensively studied [28]. Value-based methods in episodic settings have been shown to achieve regret bounds $\tilde{\mathcal{O}}(d^p H^q \sqrt{T})$ [59] (with $p = 1$, $q = 2$), where H is the episode length, and d is the aggregate dimension of the state and action space. This result has been improved to $p = q = 3/2$ in [29], and further to $p = 3/2$ and $q = 1$ in [60]. Recently, model-based methods have gained traction for improving upon the best known regret model-free methods with $p = 1$ and $q = 1/3$ [4]. A separate line of works seek to improve the dependence on T to be logarithmic through instance dependence [26, 61]. These results typically impose that the underlying MDP has a transition model that is linearly factorizable, and exhibit regret depends on the input dimension d. This condition can be relaxed through introduction of a nonlinear feature map, whose appropriate selection is highly nontrivial and lead to large gaps between regret and practice [37], or meta-procedures [34]. Aside from the feature selection issue, these approaches require evaluation of confidence sets which is computationally costly and lead to statistical inefficiencies when approximated [42].

Thus, we prioritize *Bayesian* approaches to RL [21, 30] popular in robotics [17]. While many heuristics exist, performance guarantees take the form of Bayesian regret [40], and predominantly build upon posterior (Thompson) sampling [54]. In particular, beyond the tabular setting, [41] establishes a $\tilde{O}(\sigma_R\sqrt{d_K(R)d_E(R)T} + \mathbb{E}[L^*]\sigma_p\sqrt{d_K(P)d_E(P)})$ Bayesian regret for posterior sampling RL (PSRL) combined with greedy action selections with respect to the estimated value. Here L^* is a global Lipschitz constant for the future value function, d_K and d_E are Kolmogorov and eluder dimensions, and R and P refers to function classes of rewards and transitions. The connection between H and L is left implicit; however, [19][Sec. 3.2] shows that L can depend exponentially on H. Similar drawbacks manifest in the Lipschitz parameter of the Bayesian regret bound of [14], which extends the former result to continuous spaces through kernelized feature maps. However, recently an augmentation of PSRL is proposed which employs feature embedding with Gaussian (symmetric distribution) dynamics to alleviate this issue [19], yielding the Bayesian regret of $\tilde{\mathcal{O}}(H^{\frac{3}{2}}d\sqrt{T})$ that is polynomial in H and has no dependence on Lipschitz constant L. These results are still restricted in the sense that it requires (i) the transition model target to be Gaussian, (ii) its representational complexity to grow unsustainably large with time. Therefore, in this work we as the following question:

Can we achieve a trade-off between the Bayesian regret and the posterior representational complexity (aka coreset size) without oracle access to a feature map at the outset of training, in possibly continuous state-action spaces?

We provide an affirmative answer by honing in on the total variation norm used to quantify the posterior estimation error that appears in the regret analysis of [19], and identify that it can be sharpened by instead employing an integral probability metric (IPM). Specifically, by shifting to an IPM, and then imposing structural assumptions on the target, that is, restricting it to a class of smooth densities, we can employ *Stein's identity* [49, 31] to evaluate the distributional distance in closed form using the kernelized Stein discrepancy (KSD) [22, 35]. This restriction is common in Markov Chain Monte Carlo (MCMC) [3], and imposes that the target, for instance, belongs to a family of mixture models.

This modification in the metric of convergence alone leads to improved regret because we no longer require the assumption that the posterior is Gaussian [19]. However, our goal is to translate the scalability of PSRL from tabular settings to continuous spaces which requires addressing the parameterization complexity of the posterior estimate, which grows linearly unbounded [19] . With the power to evaluate KSD in closed form, then, we sequentially remove those state-action pairs that contribute least (decided by a compression budget ϵ) in KSD after each episode (which is completely novel in the RL setting) from the posterior representation according to [25]. Therefore, the posterior estimate only retains statistically significant past samples from the trajectories, i.e., it is defined by a Bayesian coreset of the trajectory data [11, 12]. The budget parameter ϵ then is calibrated in terms of a rate determining factor α to yield both sublinear Bayesian regret and sublinear representational complexity of the learned posterior – see Table 2. The resultant procedure we call Kernelized Stein Discrepancy-based Posterior Sampling for RL (`KSRL`). Our main contributions are, then, to:

- ▷ introduce Stein's method in MBRL for the first time, and use it to develop a novel transition model estimate based upon it, which operates in tandem with a KSD compression step to remove statistically insignificant past state-action pairs, which we abbreviate as `KSRL`;
- ▷ establish Bayesian regret bounds of the resultant procedure that is sublinear in the number of episodes experienced, without any prior access to a feature map, alleviating difficult feature selection drawbacks of prior art. Notably, these results relax Gaussian and Lipschitz assumptions of prior related results;
- ▷ mathematically establish a tunable trade-off between Bayesian regret and posterior's parameterization complexity (or dictionary size) via introducing parameter $\alpha \in (0, 1]$ for the first time in this work;
- ▷ experimentally demonstrate that `KSRL` achieves favorable tradeoffs between sample and representational complexity relative to several strong benchmarks.

B Background

Let us write down the posterior sampling based reinforcement learning (PSRL) algorithm here in detail which is the basic building block of the research work in this paper [41].

Model Predictive Control : Model based RL planning with Model-predictive control (MPC) has achieved great success[9] in several continuous control problems especially due to its ability to efficiently incorporate uncertainty into the planning mechanism. MPC has been an extremely useful mechanism for solving multivariate control problems with constraints [10] where it solves a finite horizon optimal control problem in a receding horizon fashion. The integration of MPC in the Model-based RL is primarily motivated due to its implementation simplicity, where once the model is learnt, the subsequent optimization for a sequence of actions is done through MPC. At each timepoint, the MPC applies the first action from the optimal action sequence under the estimated dynamics and reward function by solving $\arg\max_{a_{i:i+\tau}} \sum_{t=i}^{i+\tau} \mathbb{E}[r(s_t, a_t)]$. However, computing the exact $\arg\max$ is non-trivial for non-convex problems and hence approximate methods like random sampling

Setting	**Refs**	**Bayes Regret**	**Coreset Size**
Tabular	PSRL [39]	$\tilde{\mathcal{O}}(HS\sqrt{AT})$	$\Omega(T)$
Tabular	PSRL2 [43]	$\tilde{\mathcal{O}}(H\sqrt{SAT})$	$\Omega(T)$
Tabular	TSDE [44]	$\tilde{\mathcal{O}}(HS\sqrt{AT})$	$\Omega(T)$
Tabular	General PSRL [1]	$\tilde{\mathcal{O}}(DS\sqrt{AT})$	$\Omega(T)$
Tabular	DS-PSRL [53]	$\tilde{\mathcal{O}}(CH\sqrt{C'T})$	$\Omega(T)$
Tabular	PSRL3 [41]	$\tilde{\mathcal{O}}(\sqrt{d_k d_E T})$	$\Omega(T)$
Continuous/Gaussian	MPC-PSRL [19]	$\tilde{\mathcal{O}}(H^{\frac{3}{2}}d\sqrt{T})$	$\Omega(T)$
Continuous/ Smooth	`KSRL` (**This work**)	$\tilde{\mathcal{O}}\left(dH^{1+(\alpha/2)}T^{1-(\alpha/2)}\right)$	$\Omega(\sqrt{T^{1+\alpha}})$

Table 2: A comparison of Bayes regret (cf. (**??**)) and Bayesian Coreset (the number of stored data points in dictionary $\mathcal{D}_k$ to represent posterior at k). We introduce KSD-based compression to model-based RL (KSRL), with tuning parameter α to obtain sublinear Bayesian regret *and* coreset size for any $\alpha \in (0, 1]$. For $\alpha = 1$, we recover the state of the art results of MPC-PSRL ($\tilde{\mathcal{O}}(dH^{3/2}\sqrt{T})$). But our results holds for general transitions which are smooth and not restricted to Gaussian assumption.

Algorithm 1 Posterior Sampling for Reinforcement Learning (PSRL) [41]

```
1: **Input** : Episode length $H$, Total timesteps $T$, Dictionary $\mathcal{D}_1$, prior distribution $\phi$ for $M^*$, i=1
2: **for** episodes $k = 1$ to $K$ **do**
3:     Sample $M^k \sim \phi(\cdot|\mathcal{D}_k)$
4:     Evaluate $\mu^k$ under $M^k$ via (**??**) and initialize empty $\mathcal{C} = []$
5:     **for** timesteps $i = 1$ to $H$ **do**
6:         Take action $a_i \sim \mu^k(s_i)$
7:         Observe $r_i$ and $s_{i+1}$ action $a_i \sim \mu^k(s_i)$
8:         Update $\mathcal{C} = \mathcal{C} \cup \{(s_i, a_i, r_i, s_{i+1})\}$
9:     **end for**
10:    Update $\mathcal{D}_{k+1} = \mathcal{D}_k \cup \mathcal{C}$
11:    Update posterior to obtain $\phi(\cdot \mid \mathcal{D}_{k+1})$
12: **end for**
```

shooting [36], cross-entropy methods [7] are commonly used. For our specific case we use the Cross-entropy method for its effectiveness in sampling and data efficiency.

C Proof of Lemma ??

Proof. Using the definition of the future value function with the Bellman Operator, we can write

$$U_i^k(P^k(\hat{h}_i)) - U_i^k(P^*(\hat{h}_i)) \leq \max_s |V_{\mu_k}^{M_k}(s)| \cdot \|P^k(\cdot|\hat{h}_i) - P^*(\cdot|\hat{h}_i)\| \tag{2}$$

$$\leq H R_{\max} \|P^k(\cdot|\hat{h}_i) - P^*(\cdot|\hat{h}_i)\|, \tag{3}$$

where we have utilized the absolute upper bound on the value function $|V(s)| \leq HR_{\max}$. The norm in (3) is the total variation distance between the probability measures. In [19, Lemma 2], the right hand side of (44) is further upper bounded by the distance between the mean functions after assuming that the underlying distributions are Gaussian. This is the point of departure in our analysis where we introduce the notion of Stein discrepancy to upper bound the total variation distance between the probability measures.

First, we build the analysis for $d = 1$ and later would extend it to multivariate scenarios. We start the analysis by showing that the total variation distance is upper bounded by the KSD for $d = 1$. Let us define the notion of an integral probability metric between two

Algorithm 2 **K**ernelized **S**tein Discrepancy-based Posterior Sampling for **RL** (KSRL)

```
1: Input : Episode length $H$, Total timesteps $T$, Dictionary $\mathcal{D}$, prior distribution $\phi = \{\mathcal{P}, \mathcal{R}\}$ for
   true MDP $M^*$, planning horizon $\tau$ for MPC Controller, thinning budget $\{\epsilon_k\}_{k=1}^K$
2: Initialization : Initialize dictionary $\mathcal{D}_1$ at with random actions from the controller as $\mathcal{D}_1 :=$
   $\{s_{1,1}, a_{1,1}, r_{1,1}, \cdots, s_{1,H}, a_{1,H}, r_{1,H}\}$, posterior $\phi_{\mathcal{D}_1} = \{\mathcal{P}_{\mathcal{D}_1}, \mathcal{R}_{\mathcal{D}_1}\}$
3: for Episodes $k = 1$ to $K$ do
4:     Sample a transition $P^k \sim \mathcal{P}_{\mathcal{D}_k}$ and reward model $r^k \sim \mathcal{R}_{\mathcal{D}_k}$ and initialize empty $\mathcal{C} = []$
5:     for timesteps $i = 1$ to $H$ do
6:         Evaluate optimal action sequence $a^*_{k,i:k,i+\tau} = \arg\max_{a_{k,i:k,i+\tau}} \sum_{t=i}^{i+\tau} \mathbb{E}[r(s_{k,t}, a_{k,t})]$
7:         Execute $a^*_{k,i}$ from the optimal sequence $a^*_{k,i:k,i+\tau}$
8:         Update $\mathcal{C} \leftarrow \mathcal{C} \cup \{(s_{k,i}, a_{k,i}, s_{k,i+1}, r_{k,i})\}$
9:     end for
10:    Update dictionary $\widetilde{\mathcal{D}}_{k+1} \leftarrow \mathcal{D}_k \cup \mathcal{C}$
11:    Perform thinning operation (cf. Algorithm 3)
           $(\phi_{\mathcal{D}_{k+1}}, \mathcal{D}_{k+1}) = \text{KSD-Thinning}(\phi_{\widetilde{\mathcal{D}}_{k+1}}, \widetilde{\mathcal{D}}_{k+1}, \epsilon_k)$
12: end for
```

Algorithm 3 Posterior Coreset with **KSD Thinning** for **R**einforcement **L**earning (KSD-Thinning)

```
1: Input: $(q_{\mathcal{W}}, \mathcal{W}, \epsilon)$
2: Require: Target score function
3: Compute the reference KSD as $\alpha := \text{KSD}(q_{\mathcal{W}})$ via (??)
4: while $\text{KSD}(q_{\mathcal{W}})^2 < \alpha^2 + \epsilon$ do
5:     Compute the least influential point $x_j$ as the minimial $h_i \in \widetilde{\mathcal{D}}_{k+1}$ (??)
6:     if $\text{KSD}(q_{\mathcal{W}\setminus\{x_j\}})^2 < \alpha^2 + \epsilon$ then
7:         Remove the least influential point, set $\mathcal{W} = \mathcal{W} \setminus \{x_j\}$
8:     else
9:         Break loop
10:    end if
11: end while
12: Output thinned dictionary $\mathcal{W}$ satisfying $\text{KSD}(q_{\mathcal{W}})^2 < \alpha^2 + \epsilon$
```

distributions p and q as

$$D(p,q) := \sup_{f\in\mathcal{F}} \left| \int f dp - \int f dq \right|,$$

where $\mathcal{F}$ is any function space. Now, if we restrict ourselves to a function space $\mathcal{F}' := \{f : \|f\|_\infty \leq 1\}$, then we boils down to the definition of total variation distance between p and q given by

$$TV(p,q) := \sup_{f\in\mathcal{F}'} \left| \int f dp - \int f dq \right|.$$

We can restrict our function class to be $RKHS$ given by $\mathcal{H}$ and still write

$$TV(p,q) := \sup_{f\in\mathcal{H}'} \left| \int f dp - \int f dq \right|,$$

where we define $\mathcal{H}' := \{f : \|f\|_\infty \leq 1\}$. We note that $\mathcal{H}'$ is uniquely determined by the kernel $\kappa(x,y)$ (e.g., RBF kernel) . Now, we know $\mathcal{H}'$ is a subset of RKHS $\mathcal{H}$. Now, we will try to upper bound the supremum over $\mathcal{H}'$ with supremum over a general class of functions which we call Stein class of functions as $\mathcal{H}_S$ [35]. Note that since $\mathcal{H}'$

$$TV(p,q) = \sup_{f\in\mathcal{H}'} \left| \int f dp - \int f dq \right| \leq \sup_{f\in\mathcal{H}_S} \left| \int f dp - \int f dq \right|.$$

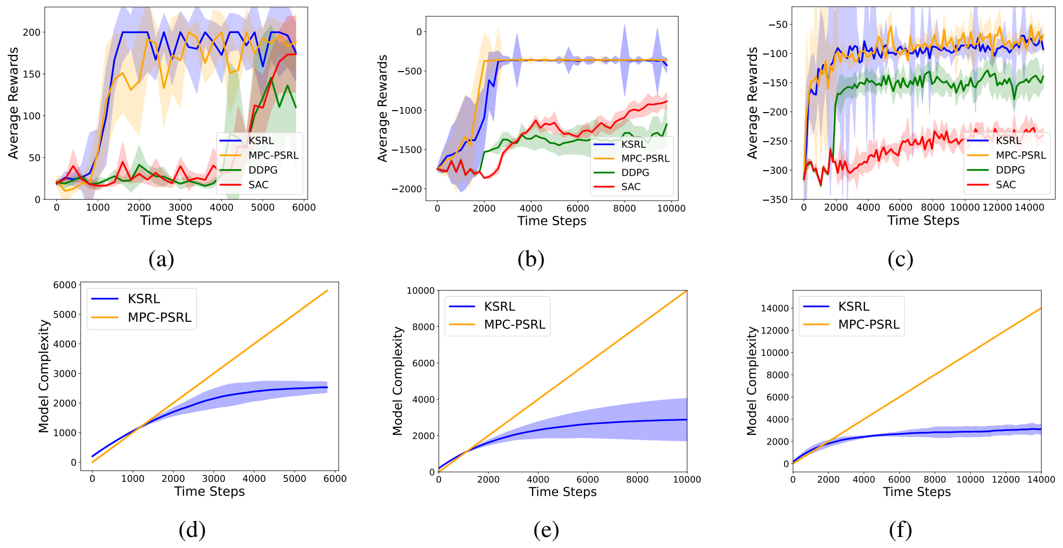


Figure 2: **(a)-(c)** compares the average cumulative reward return achieved by the proposed KSRL (shown in blue) algorithm with MPC-PSRL [19], SAC [23], and DDPG [5] for modified **Cartpole**, **Pendulum**, and **Pusher** without rewards. Fig. 4 in the Appendix F shows with rewards. **(d)-(f)** compares the model-complexity. We note that KSRL is able to achieve the maximum average reward at-par with the current SOTA MPC-PSRL with drastically reduced model complexity. Solid curves represent the average across five trials (seeds), shaded areas correspond to the standard deviation amongst the trials.

Now, for Stein class of functions $\mathcal{H}_S$, it holds that $\int f dp = 0$. Therefore, we can write

$$TV(p, q) = \sup_{f \in \mathcal{H}'} \left| \int f dp - \int f dq \right| \leq \sup_{f \in \mathcal{H}_S} \left| \int f dp - \int f dq \right| = \sup_{f \in \mathcal{H}_S} \left| \int f dq \right|$$

Hence, we can write

$$TV(p, q) \leq \sup_{f \in \mathcal{H}_S} \left| \int f dq \right| =: \text{KSD}(q).$$

Utilizing the above upper bound into the right hand side of (3), we can write

$$U_i^k(P^k(\hat{h}_i)) - U_i^k(P^*(\hat{h}_i)) \leq H R_{\max} \text{KSD}\left(P^k(h_i)\right). \tag{4}$$

The above result holds for $d = 1$ case, and assuming that the variables are independent of each other in all the dimensions, we can naively write that in d dimensions, we have

$$U_i^k(P^k(\hat{h}_i)) - U_i^k(P^*(\hat{h}_i)) \leq d H R_{\max} \text{KSD}\left(P^k(h_i)\right). \tag{5}$$

Hence proved. □

D Proof of Lemma ??

Before starting the proof, here we discuss what is unique about it as compared to the existing literature. The kernel Stein discrepancy based compression exists in the literature [24] but that is limited to the settings of estimating the distributions. This is the first time we are extending the analysis to model based RL settings. The samples in our setting are collected in the form of episodes following an optimal policy μ^k (cf. (**??**)) for each episode. The analysis here follows a similar structure to the one performed in the proof of [24, Theorem 1] with careful adjustments to for the RL setting we are dealing with in this work. Let us now begin the proof.

Proof. Since we are learning for both reward and transition model denoted by $\mathcal{R}$ and $\mathcal{P}$ and they both are parameterized by the same dictionary $\mathcal{D}_k$, we present the KSD analysis for $\Lambda_{\mathcal{D}_k} := \{P^k, R^k\}$ without loss of generality. Further, for the proof in this section, we divide the H samples collected in k^{th} episode into $M > 1$ number of batches, and then select the KSD optimal point from each batch of H/M samples similar to SPMCMC procedure. We will start with the one step transition at $k-1$ where we have the dictionary $\mathcal{D}_{k-1}$ and we update it to obtain $\widetilde{\mathcal{D}}_k$ which is before the thinning operation. From the definition of KSD in (**??**), we can write

$$|\widetilde{\mathcal{D}}_k|^2 \mathrm{KSD}(\Lambda_{\widetilde{\mathcal{D}}_k})^2 = \sum_{h_i \in \mathcal{D}_k} \sum_{h_j \in \mathcal{D}_k} k_0(h_i, h_j) \tag{6}$$

$$= |\mathcal{D}_{k-1}|^2 \mathrm{KSD}(\Lambda_{\mathcal{D}_{k-1}})^2 + \sum_{m=1}^{H/M} \left[k_0(h_k^m, h_k^m) + 2 \sum_{h_i \in \mathcal{D}_{k-1}} k_0(h_i, h_k^m) \right]. \tag{7}$$

Next, for each m, since we select (SPMCMC based method in [13]) the sample h_k^m from $\mathcal{Y}_m := \{h_k^l\}_{l=1}^M$, and without loss of generality, we assume that H/M is an integer. Now, from the SPMCMC based selection, we can write

$$\begin{aligned} k_0(h_k^m, h_k^m) + 2 \sum_{h_i \in \mathcal{D}_{k-1}} k_0(h_i, h_k^m) &= \inf_{h_k^m \in \mathcal{Y}_m} k_0(h_k^m, h_k^m) + 2 \sum_{h_i \in \mathcal{D}_{k-1}} k_0(h_i, h_k^m) \\ &\leq S_k^2 + 2 \inf_{h_k^m \in \mathcal{Y}_m} \sum_{h_i \in \mathcal{D}_{k-1}} k_0(h_i, h_k^m). \end{aligned} \tag{8}$$

The inequality in (8) holds because we restrict our attention to regions for which it holds that $k_0(\mathbf{x}, \mathbf{x}) \leq S_k^2$ for all $\mathbf{x} \in \mathcal{Y}_k^m$ for all k and m. Utilizing the upper bound of (8) into the right hand side of (7), we get

$$|\widetilde{\mathcal{D}}_k|^2 \mathrm{KSD}(\Lambda_{\widetilde{\mathcal{D}}_k})^2 \leq |\mathcal{D}_{k-1}|^2 \mathrm{KSD}(\Lambda_{\mathcal{D}_{k-1}})^2 + \frac{H S_k^2}{M} + 2 \sum_{m=1}^{H/M} \inf_{h_k^m \in \mathcal{Y}_m} \sum_{h_i \in \mathcal{D}_{k-1}} k_0(h_i, h_k^m). \tag{9}$$

Eqn (6) clearly differentiates our method from [24] highlighting the novelty of our approach is deciphering the application of KSD to our model based RL problem. From the application of Theorem 5 [24] for our formulation with H new samples in the dictionary.

$$2 \inf_{h_k^m \in \mathcal{Y}_m} \sum_{h_i \in \widetilde{\mathcal{D}}_{k-1}} k_0(h_i, h_k^m) \leq r_k \|f_k\|_{\mathcal{K}_0}^2 + \frac{\mathrm{KSD}(\Lambda_{\mathcal{D}_{k-1}})^2}{r_k}, \tag{10}$$

for any arbitrary constant $r_k > 0$. We use the upper bound in (10) to the right hand side of (9), to obtain

$$|\widetilde{\mathcal{D}}_k|^2 \mathrm{KSD}(\Lambda_{\widetilde{\mathcal{D}}_k})^2 \leq |\mathcal{D}_{k-1}|^2 \left(1 + \frac{H}{r_k M}\right) \mathrm{KSD}(\Lambda_{\mathcal{D}_{k-1}})^2 + \frac{H S_k^2 + r_k H \|f_k\|_{\mathcal{K}_0}^2}{M}. \tag{11}$$

Next, we divide the both sides by $|\widetilde{\mathcal{D}}_k|^2 = (|\mathcal{D}_{k-1}| + H/M)^2$ to obtain

$$\mathrm{KSD}(\Lambda_{\widetilde{\mathcal{D}}_k})^2 \leq \frac{|\mathcal{D}_{k-1}|^2}{(|\mathcal{D}_{k-1}| + H/M)^2} \left(1 + \frac{H}{r_k M}\right) \mathrm{KSD}(q_{\mathcal{D}_{k-1}})^2 + \frac{H\left(S_k^2 + r_k \|f_k\|_{\mathcal{K}_0}^2\right)}{M\left(|\mathcal{D}_{k-1}| + H/M\right)^2} \tag{12}$$

Now, in this step we apply equation (**??**) and replace $\Lambda_{\widetilde{\mathcal{D}}_k}$ with the thinned dictionary one $\Lambda_{\mathcal{D}_k}$ and rewriting equation (12) as

$$\text{KSD}(\Lambda_{\mathcal{D}_k})^2 \leq \frac{|\mathcal{D}_{k-1}|^2}{(|\mathcal{D}_{k-1}| + H/M)^2}\left(1 + \frac{H}{r_k M}\right)\text{KSD}(\Lambda_{\mathcal{D}_{k-1}})^2 + \frac{H\left(S_k^2 + r_k\|f_k\|_{\mathcal{K}_0}^2\right)}{M\left(|\mathcal{D}_{k-1}| + H/M\right)^2} + \epsilon_k. \tag{13}$$

Note that we have established a recursive relationship for the KSD of the thinned distribution in (13). This is quite interesting because it would pave the way to establish the regret result which is the eventual goal. After unrolling the recursion in (13), we can write

$$\text{KSD}(\Lambda_{\mathcal{D}_k})^2 \leq \sum_{i=1}^{k}\left(\frac{H\left(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2\right)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2} + \epsilon_i\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\left(\prod_{j=i}^{k-1}\left(1 + \frac{H}{r_{j+1}M}\right)\right). \tag{14}$$

Taking expectation on both sides of (14) and applying the log-sum exponential bound we get

$$\mathbb{E}\left[\text{KSD}(\Lambda_{\mathcal{D}_k})^2\right] \leq \mathbb{E}\left[\exp\left(\frac{H}{M}\sum_{j=1}^{k}\frac{1}{r_j}\right)\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2} + \epsilon_i\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right], \tag{15}$$

where the log-sum exponential bound is used as

$$\prod_{j=i}^{k-1}\left(1 + \frac{H}{r_{j+1}M}\right) \leq \exp\left(\frac{H}{M}\sum_{j=1}^{n}\frac{1}{r_j}\right). \tag{16}$$

Next, we consider the inequality in (15), ignoring the exponential term, we further decompose the remaining summation term into two parts, respectively, as:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2} + \epsilon_i\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right] \qquad (17)\\
&= \underbrace{\mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2}\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\widetilde{D}_j| + H/M}\right)^2\right]}_{T_1} + \underbrace{\mathbb{E}\left[\sum_{i=1}^{k}\epsilon_i\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right]}_{T_2},
\end{aligned}$$

where T_1 corresponds to the sampling error and T_2 corresponds to the error due to the proposed thinning scheme. The term T_1 represents the bias incurred at each step of the un-thinned point selection scheme. The term T_2 is the bias incurred by the thinning operation carried out at each step. However, for our case in the model-based reinforcement learning setting, the bias term T_1 will be different in our case as the samples are generated in a sequential manner in each episode from a Transition kernel with a decaying budget. Let develop upper bounds on both T_1 and T_2 as follows.

Bound on T_1: Let consider the first term on the right hand side of (17) as follow

$$\begin{aligned}
T_1 =&\mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2}\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right]\\
=&\mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{k-1}| + H/M\right)^2}\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_{j-1}| + H/M}\right)^2\right], \qquad (18)
\end{aligned}$$

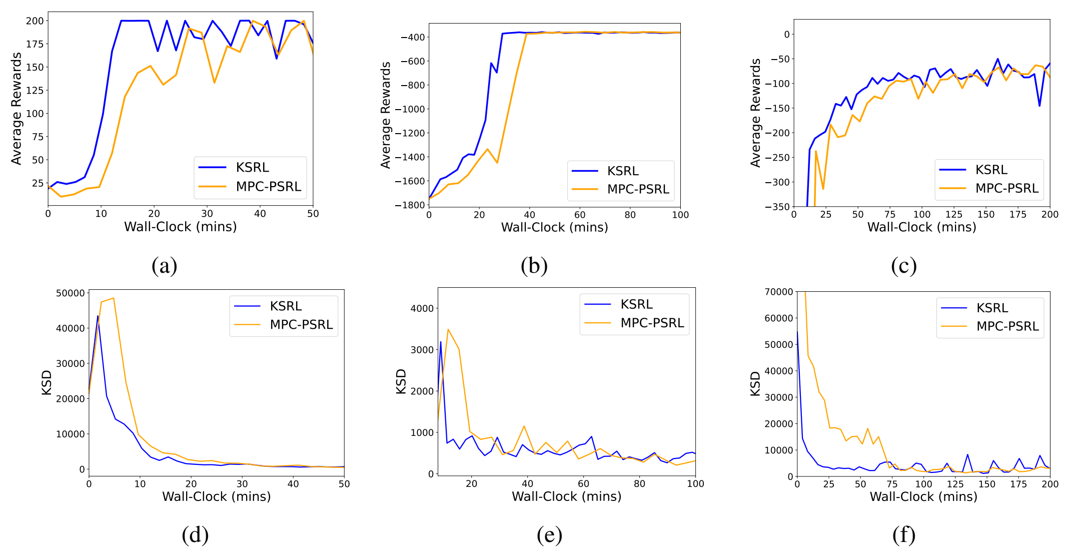


Figure 3: Performance against wall clock time and KSD Convergence results: **(a)-(c)** shows the average reward return against wall clock time (in minutes) for modified Cartpole, Pendulum, and Reacher (mean across 5 runs). **(d)-(f)** provides the evidence of KSD convergence which shows we are learning the target posterior effectively without any bias (even we are compressing the dictionary). From **(a)-(c)** it is evident that `KSRL`(blue) is able to achieve the similar performance even earlier than the existing dense counterparts with no thinning. Wall-clock time measured in minutes for runs in CPU.

where the equality in (18) holds by rearranging the denominators in the multiplication, and pulling $(|\mathcal{D}_{k-1}| + H/M)^2$ inside the first term. Next, from the fact that $|\mathcal{D}_j| \leq |\mathcal{D}_{j-1}| + H/M$ which implies that the product will be less that 1, we can upper bound the right hand side of (18) as follows

$$T_1 \leq \mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|^2_{\mathcal{K}_0})}{M\left(|\mathcal{D}_{k-1}| + H/M\right)^2}\right)\right] \tag{19}$$

$$= \sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\mathbb{E}\left[\|f_i\|^2_{\mathcal{K}_0}\right])}{M\left(|\mathcal{D}_{k-1}| + H/M\right)^2}\right), \tag{20}$$

where we took expectation inside the summation and apply it to the random variable in the numerator. From the model order growth condition, we note that $|\mathcal{D}_{k-1}| + H/M \geq |\mathcal{D}_k| \geq f(k)$, which implies that $1/\left(|\mathcal{D}_{k-1}| + H/M\right)^2 \leq 1/f(k)^2$, which we utilize in the right hand side of (20) to write

$$T_1 \leq \mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|^2_{\mathcal{K}_0})}{M\left(|\mathcal{D}_{k-1}| + H/M\right)^2}\right)\right] \tag{21}$$

$$= \sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\mathbb{E}\left[\|f_i\|^2_{\mathcal{K}_0}\right])}{Mf(k)^2}\right) \tag{22}$$

Following the similar logic mentioned in [13, Appendix A, Eqn. (17)], we can upper bound $\mathbb{E}\left[\|f_i\|^2_{\mathcal{K}_0}\right]$ as

$$\mathbb{E}\left[\|f_i\|^2_{\mathcal{K}_0}\right] \leq \frac{4b}{\gamma}\exp\left(-\frac{\gamma}{2}S_i^2\right) + \frac{4}{H/M}S_i^2, \tag{23}$$

which is based on the assumption that $\mathbb{E}_{\mathbf{y}\sim P}\left[\exp(\gamma\kappa_0(\mathbf{y},\mathbf{y}))\right] = b < \infty$. This implies that

$$T_1 \leq \frac{1}{f(k)^2}\sum_{i=1}^{k}\left((H/M + 4r_i)S_i^2 + \frac{4br_iH}{\gamma M}\exp\left(-\frac{\gamma}{2}S_i^2\right))\right), \tag{24}$$

which we obtain by applying the upper bound in (23) into (22). After simplification, we can write

$$T_1 \leq \mathcal{O}\left(\frac{k\log(k)}{f(k)^2}\right), \tag{25}$$

which provide the bound on T_1. Next, we derive the bound on T_2 as follows.

Bound on T_2 : Let us consider the term T_2 from (17) as follows

$$T_2 = \mathbb{E}\left[\sum_{i=1}^{k}\epsilon_i\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right]. \tag{26}$$

This term is extra in the analysis and appears due to the introduction of compression budget ϵ_k into the algorithm. We need control the growth of this term, and by properly designing ϵ_k, we need to make sure T_2 goes to zero at least as fast at T_1 to obtain a sublinear regret analysis. We start by observing that $|\mathcal{D}_j| \leq (H/M)j$ which holds trivially and hence implies

$$\frac{|\mathcal{D}_j|}{|\mathcal{D}_{j+1}|} \leq \frac{j}{j+1}. \tag{27}$$

From the algorithm construction, we know that $|\mathcal{D}_{j+1}| \leq |\mathcal{D}_j| + H/M$, applying this to (27), we obtain

$$\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M} \leq \frac{j}{j+1}. \tag{28}$$

Utilize the upper bound in (28) to the right hand side of (26), to write

$$T_2 \leq \sum_{i=1}^{k}\epsilon_i\left(\prod_{j=i}^{k-1}\frac{j}{j+1}\right)^2 = \sum_{i=1}^{k}\epsilon_i\frac{i^2}{k^2} = \frac{1}{k^2}\sum_{i=1}^{k}\epsilon_i i^2. \tag{29}$$

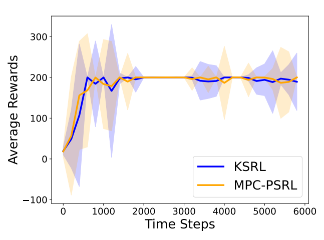
The above bound implies that we should choose the compression budget ϵ_i such that T_2 goes to zero at least as fast at T_1

$$\frac{1}{k^2}\sum_{i=1}^{k}\epsilon_i i^2 \leq \frac{k\log(k)}{f(k)^2} \tag{30}$$

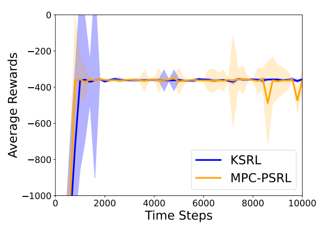
$$\sum_{i=1}^{k}\epsilon_i i^2 \leq \frac{k^3\log(k)}{f(k)^2}. \tag{31}$$

To satisfy the above condition, we choose $\epsilon_i = \frac{\log(i)}{f(i)^2}$, and obtain

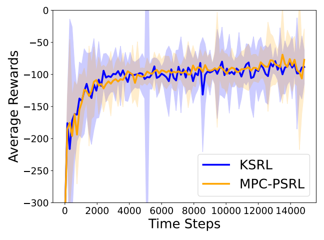
$$\begin{aligned}\sum_{i=1}^{k}\epsilon_i i^2 &= \sum_{i=1}^{k}\frac{i^2\log(i)}{f(i)^2}\\ &\leq \sum_{i=1}^{k}\frac{k^2\log(k)}{f(k)^2}\\ &\leq \frac{k^3\log(k)}{f(k)^2},\end{aligned} \tag{32}$$



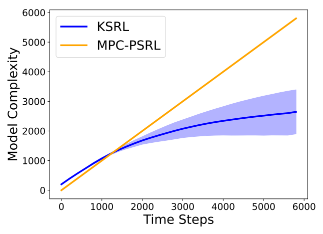
(a) .



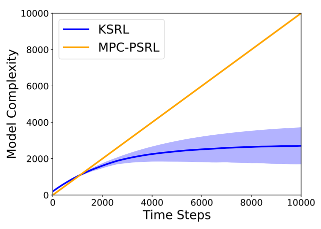
(b)



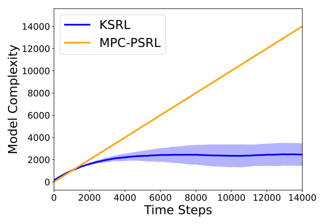
(c)



(d)



(e)



(f)

Figure 4: **(a)-(c)** compares the average cumulative reward return achieved by the proposed `KSRL` (shown in blue) algorithm with MPC-PSRL [19], SAC [23], and DDPG [5] for **Cartpole**, **Pendulum**, and **Pusher** with oracle rewards. **(d)-(f)** compares the model-complexity. We note that `KSRL` is able to achieve the maximum average reward at-par with the current state of the art MPC-PSRL with drastically reduced model complexity. Solid curves represent the average across five trials (seeds), shaded areas correspond to the standard deviation amongst the trials

.

which satisfy the upper bound in (31), which shows that $\epsilon_i = \frac{\log(i)}{f(i)^2}$ is a valid choice. Hence, $T_2 \leq \frac{k\log(k)}{f(k)^2}$.

Finally, after substituting the upper bounds for T_1 and T_2 into (17), we obtain

$$\mathbb{E}\left[\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2} + \epsilon_i\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right] \leq \frac{2k\log(k)}{f(k)^2}. \tag{33}$$

Now revisiting the inequality in (14), we write

$$\begin{aligned}\mathbb{E}\left[\text{KSD}(\Lambda_{\mathcal{D}_k})^2\right] \leq \mathbb{E}&\left[\exp\left(\frac{H}{M}\sum_{j=1}^{k}\frac{1}{r_j}\right)\sum_{i=1}^{k}\left(\frac{H(S_i^2 + r_i\|f_i\|_{\mathcal{K}_0}^2)}{M\left(|\mathcal{D}_{i-1}| + H/M\right)^2} + \epsilon_i\right)\left(\prod_{j=i}^{k-1}\frac{|\mathcal{D}_j|}{|\mathcal{D}_j| + H/M}\right)^2\right]\\ &\leq \exp\left(\frac{H}{M}\sum_{j=1}^{k}\frac{1}{r_j}\right)\left(\frac{2k\log(k)}{f(k)^2}\right)\\ &\leq e\left(\frac{2k\log(k)}{f(k)^2}\right).\end{aligned} \tag{34}$$

where the last equality holds by the selection $r_j = \frac{Hk}{M}$ for all j. Finally, we collect all the constants in C to write

$$\mathbb{E}\left[\text{KSD}(\Lambda_{\mathcal{D}_k})^2\right] \leq \mathcal{O}\left(\frac{k\log(k)}{f(k)^2}\right). \tag{35}$$

Next, from applying Jensen's inequality on the left hand side in (35), and we can write

$$\mathbb{E}\left[\text{KSD}(\Lambda_{\widetilde{D}_k})\right] \leq \mathcal{O}\left(\frac{\sqrt{k\log(k)}}{f(k)}\right). \tag{36}$$

Hence proved. □

E Proof of Theorem ??

Proof. Based on the equality in (23), the first part of our regret in (**??**) becomes zero and we only need to derive and upper bound for Δ_k^{II}. Recall that, we have

$$\Delta_k^{II} = \int \rho(s_1)(V_{\tilde{\mu}^k}^{\widetilde{M}^k}(s_1) - V_{\tilde{\mu}^k}^{M^*}(s_1))ds_1. \tag{37}$$

Following the Bellman equations representation of value functions, We can factorize Δ_k^{II} in-terms of the immediate rewards r and the future value function U (cf. [41]) as follows

$$\mathbb{E}[\Delta_k^{II}|\mathcal{D}_k] = \mathbb{E}[\Delta_k(r) + \Delta_k(f)|\mathcal{D}_k], \tag{38}$$

where we define

$$\Delta_k(r) = \sum_{i=1}^{H}(r^k(\hat{h}_i) - r^*(\hat{h}_i)), \tag{39}$$

$$\Delta_k(P) = \sum_{i=1}^{H}(U_i^k(P^k(\hat{h}_i)) - U_i^k(P^*(\hat{h}_i))). \tag{40}$$

Now, for our Stein based thinning algorithm, we thin the updated dictionary $\widetilde{\mathcal{D}}_k$ after every episode (H steps) and obtain a compressed and efficient representation of dictionary given by $\mathcal{D}_k$. Let us derive the upper bound on $\mathbb{E}[\Delta_k(P)]$as follows. The Bayes regret at the k^{th} episode can be written as

$$\mathbb{E}[\Delta_k(P)] = \mathbb{E}\left[\sum_{i=1}^{H}(U_i^k(P^k(\hat{h}_i)) - U_i^k(P^*(\hat{h}_i)))\right]. \tag{41}$$

Utilize the upper bound from the statement of Lemma **??** to write

$$\begin{aligned}\mathbb{E}[\Delta_k(P)] &\leq \mathbb{E}\left[\sum_{i=1}^{H} dHR_{\max}\text{KSD}(P^k(\cdot|\hat{h}_i))\right] \\ &= dHR_{\max}\sum_{i=1}^{H}\mathbb{E}\left[\text{KSD}(\tilde{P}^k(h_i))\right]. \end{aligned}\tag{42}$$

This is a an important step and point of departure from the existing state of the art regret analysis for model based RL methods [19, 41]. Instead of utilizing the naive upper bound of Total Variation distance in the right hand side of (41), we follow a different approach and bound it via the Kernel Stein Discrepancy which is a novel connection explored for the first time in this work. We remark that the KSD upper bound in Lemma **??** is for the joint posterior where samples are $h_i = (s, a, s')$. We note here that since the score function if independent of the normalizing constant, we can write $\nabla \log \tilde{P}^k(\cdot|h_i') = \nabla \log \tilde{P}^k(h_i)$, therefore we utilize the KSD upper bound of joint posterior in Lemma **??** on the KSD term per episode in the right hand side of (42) to obtain

$$\mathbb{E}[\Delta_k(P)] \leq dHR_{\max}\sum_{i=1}^{H}\frac{\sqrt{k\log(k)}}{f(k)} = dH^2R_{\max}\frac{\sqrt{k\log(k)}}{f(k)}. \tag{43}$$

Next, we take summation over the number of episodes which are given by $[\frac{T}{H}]$ where T is the total number of time steps in the environments, and H is the episode length. So, after

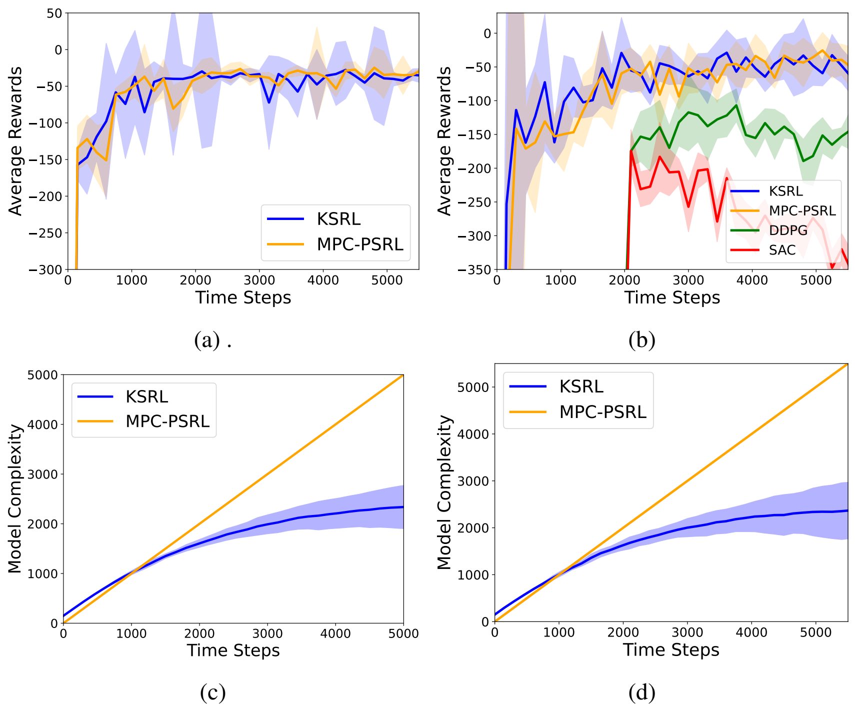


Figure 5: **(a)-(b)** compares the average cumulative reward return achieved by the proposed KSRL (shown in blue) algorithm with MPC-PSRL [19], SAC [23], and DDPG [5] for **Reacher** with and without oracle rewards respectively. **(c)-(d)** compares the model-complexity. We note that KSRL is able to achieve the maximum average reward at-par with the current state of the art MPC-PSRL with drastically reduced model complexity. Solid curves represent the average across five trials (seeds), shaded areas correspond to the standard deviation amongst the trials

.

summing over $k = 1$ to $[\frac{T}{H}]$, we obtain

$$\sum_{k=1}^{[\frac{T}{H}]} \mathbb{E}[\Delta_k(P)] \leq dH^2 R_{\max} \sum_{k=1}^{[\frac{T}{H}]} \frac{\sqrt{k\log(k)}}{f(k)}. \tag{44}$$

To derive the explicit regret rates, we assume our dictionary growth function $f(k) = \sqrt{k^{\alpha+1}\log(k)}$ with range of $\alpha \in [0, 1]$. Substituting this into (44), we can write

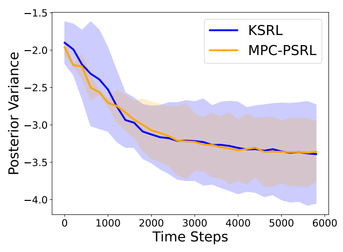
$$\sum_{k=1}^{\frac{T}{H}} \frac{\sqrt{k\log(k)}}{f(k)} = \sum_{k=1}^{\frac{T}{H}} \frac{\sqrt{k\log(k)}}{\sqrt{k^{\alpha+1}\log(k)}} = \sum_{k=1}^{\frac{T}{H}} k^{-\frac{\alpha}{2}} \leq \int_0^{\frac{T}{H}} x^{-\frac{\alpha}{2}}\,dx = \frac{2}{1-\alpha/2} T^{1-\frac{\alpha}{2}} H^{1+\frac{\alpha}{2}}. \tag{45}$$

Using (45) into (44), we get

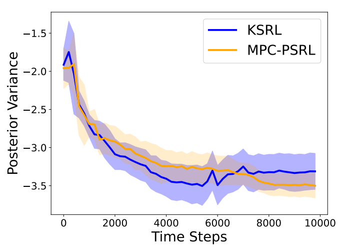
$$\sum_{k=1}^{[\frac{T}{H}]} \mathbb{E}[\Delta_k(P)] \leq d\frac{2}{1-\alpha/2} T^{1-\frac{\alpha}{2}} R_{\max} H^{1+\frac{\alpha}{2}}. \tag{46}$$

The expression implies that

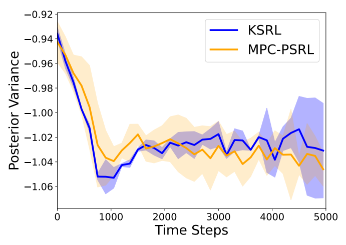
$$\sum_{k=1}^{[\frac{T}{H}]} \mathbb{E}[\Delta_k(P)] = \mathcal{O}\left(dT^{1-\frac{\alpha}{2}} H^{1+\frac{\alpha}{2}}\right). \tag{47}$$



(a) .



(b)



(c)

Figure 6: **(a)-(c)** compares the posterior variance of our KSRL(blue) with MPC-PSRL [19] for Cartpole, Pendulum & Reacher environments across the timesteps which shows we are learning the true posterior effectively without any significant bias (even we are compressing the dictionary). From **(a)-(c)** it is evident that posterior variance from KSRL(blue) with compression converges to very similar posterior variance achieved by MPC-PSRL which highlights that the uncertainty estimation in our KSRLis accurate. Plot is in logarithmic scale. Solid curves represent the average across five trials (seeds), shaded areas correspond to the standard deviation amongst the trials

The same derivation would hold for the term $\mathbb{E}[\Delta_k(r)]$ (similar logic to [19, Sec. 3.4]0), which would imply that $\sum_{k=1}^{[\frac{T}{H}]} \mathbb{E}[\Delta_k(r)] \leq \mathcal{O}\left(dT^{1-\frac{\alpha}{2}}H^{1+\frac{\alpha}{2}}\right)$. From (38) and (47), we can write

$$\sum_{k=1}^{[\frac{T}{H}]} \mathbb{E}[\Delta_k^{II}] = \mathcal{O}\left(dT^{1-\frac{\alpha}{2}}H^{1+\frac{\alpha}{2}}\right). \tag{48}$$

Hence proved. □

F Additional Experiments and Analysis

In this section, we first provide details of the environments and the complexities added in order to validate multiple aspects of our KSRL.

Low-dimensional environments: We consider the Continuous Cartpole ($d_s = 4$, $d_a = 1, H = 200$) environment with a continuous action space which is a modified version of the discrete action classic Cartpole environment. The continuous action space enhances the complexity of the environment and makes it hard for the agent to learn. We also consider the Pendulum Swing Up ($d_s = 3$, $d_a = 1, H = 200$) environment, a modified version of Pendulum where we limit the start state to make it harder and more challenging for exploration. To introduce stochasticity into the dynamics, we modify the physics of the environment with independent Gaussian noises ($\mathcal{N}(0, 0.01)$). However, these environments are primarily lower dimensional environments.

Higher-dimensional environments We consider the 7-DOF Reacher ($d_s = 17, d_a = 7, H = 150$) and 7-DOF pusher ($d_s = 20, d_a = 7, H = 150$) two challenging continuous control tasks as detailed in [15]. We increase the complexity from lower to higher dimensional environments and with added stochasticity to see the robustness of our algorithm and the consistency in performance. We conduct the experiments both with and without true oracle rewards and compare the performance with other baselines.

F.1 Experimental Details

It is shown in literature [19] that a simple Bayesian Linear regression with non-linear feature representations learnt by training a Neural network works exceptionally well in the context of posterior sampling reinforcement learning. For a fair comparison of our KSRL, we follow a similar architecture as [19] where we first train a deep neural network for both

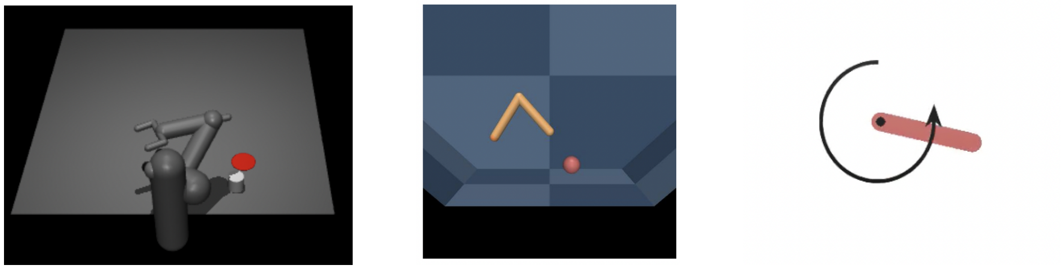


Figure 7: Continuous Control Environments **Pusher, Reacher, Pendulum** from [55] [8] used to validate the performance of KSRLin our experiments

.

the transition and rewards model and extract the penultimate layer of the network for the Bayesian linear regression and Posterior sampling. As per our notation s_i, a_i, s_{i+1}, r_i where $h_i =< s_i, a_i >$ be the current state-action pair with the next state s_{i+1} and reward r_i and let's assume the representation from the penultimate layer of the deep neural network of the state-action pair is denoted by $z_i = NN(h_i) \in R^d$ where NN is the trained deep neural network model and d is the dimensionality of the representation. Then the Bayesian linear regression model deals with learning the posterior distribution $P_{post} = P(\beta|D)$, with the linear model given by $\delta_i = \beta^T z_i + \epsilon$, where $\delta_i = s_{i+1} - s_i$ as suggested in [17, 19] and $\epsilon \sim N(0, \sigma^2)$, D is the size of the dictionary. Similar to prior approaches, we choose a multivariate Gaussian prior with zero mean and Σ_{prior} (conjugate prior) to obtain a closed-form estimation of the posterior distribution which is also multivariate Gaussian. Now, we sample β from the posterior distribution P_{post} at the beginning of each episode and interact with the environment using MPC controller. As described in Appendix A at each timepoint, the MPC applies the first action from the optimal action sequence under the estimated dynamics and reward function by solving $\arg\max_{a_{i:i+\tau}} \sum_{t=i}^{i+\tau} \mathbb{E}[r(s_t, a_t)]$, where τ is the horizon and is considered as a hyperparameter. However, as described above the above method suffers from high computation complexity as the matrix multiplication step in posterior estimation is of order $O(d^2N)$ which scales linearly with the size of the dictionary prior to that episode N. This not only enhances the computational complexity but also makes the optimization with MPC extremely hard. Hence, in our algorithm KSRLwe construct an efficient posterior coreset with Kernelized Stein discrepancy measure from (**??**) and validate the average reward achieved with the compressed dictionary. We observe in all the cases with varied complexity, our algorithm KSRLperforms equally well or sometimes even better than the uncompressed current state of the art MPC-PSRL method with drastically reduced model complexity.

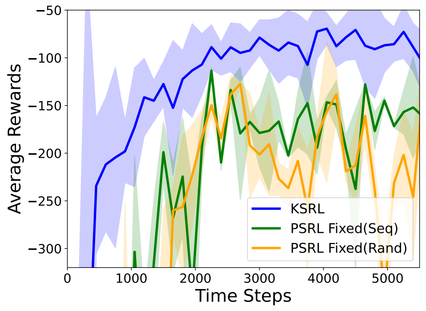


Figure 8: Comparison of our KSRLwith MPC-PSRL restricted with fixed size buffer on Reacher which clearly shows improvement of our KSRL

F.2 Experimental Analysis

We perform a detailed multifaceted analysis comparing our algorithm KSRLwith other baselines and state of the art methods in the continuous control environments. We perform the experiments in the environments with and without oracle rewards. The environmental setting of without oracle rewards is much more complex as here we have to model the reward function as well along with the dynamics which makes it much harder for the agent to learn with the added uncertainty in modelling. We also enhance the complexity of the environments by adding stochasticity which makes it harder for the agent to learn even for environments with oracle rewards and we also observe the performance by varying the dimensionality of the environments. In Figure 2 and Figure 4, we compare our KSRLwith other baselines and SOTA algorithms for Cartpole, Pendulum and Pusher environments without and with oracle rewards respectively and Figure 5 for Reacher with and without rewards. In all the cases, KSRLshows performance at-par or even better for some cases with drastically reduced model complexity where we can see a benefit of 80% improvement in the model complexity over the current SOTA with similar performance in terms of average rewards. In Figure 3, we validate the average reward achieved by our KSRLagainst baselines with respect to the runtime (wallclock time) in CPU minutes and clearly observe improved performance in-terms of wallclock time where KSRLachieves optimal performance prior to the baselines and MPC-PSRL. We also perform the convergence analysis from an empirical perspective and observe the convergence in-terms of both Kernelized Stein Discrepancy and Posterior Variance. In Figure 3 **(d) -(f)**, we study the convergence in-terms of KSD and observe the convergence of our algorithm KSRLwithout any bias and faster than the dense counterpart MPC-PSRL in-terms of wall clock time. We also study the convergence from the posterior variance perspective as it is extremely important for PSRL based algorithms to accurately estimate the uncertainty. Since, we are compressing the dictionary it is important to analyze and monitor the posterior variance over the timesteps to ensure that we are not diverging and close to the dense counterparts. In 6, we observe the posterior variance achieved by KSRLconverges to the posterior variance achieved by MPC-PSRL [19] even though we are compressing the dictionary which highlights the efficacy of our posterior coreset. Finally, from the above plots and analysis we conclude that our our algorithm KSRLachieves state of the art performance for continuous control environments with drastically reduced model complexity of its dense counterparts with theoretical guarantees of convergence.

F.3 Details of Hyperparamters

Environment	Cartpole	Pendulum	Pusher	Reacher
Steps per episode	200	200	150	150
Popsize	500	100	500	400
Number of elites	50	5	50	40
Network architecture	MLP with 2 hidden layers of size 200	MLP with 2 hidden layers of size 200	MLP with 4 hidden layers of size 200	MLP with 4 hidden layers of size 200
Planning horizon	30	20	25	25
Max iter	5			

Table 3: Hyperparameters used for our algorithm KSRL

We keep the hyperparameters and the network architecture almost similar to [19] for a fair comparison. For the baseline implementation of MPC-PSRL algorithm and our algorithm

`KSRL`we modify and leverage [1] and observe that we were able to achieve performance as described in [19]. For obtaining the results from model-free algorithms as shown in we use [2], and could replicate the results. Finally, for our posterior compression algorithm we modify the [3], [4] to fit to our scenario of posterior sampling reinforcement learning. We are thankful to all the authors for open-sourcing their repository.

Acknowledgments and Disclosure of Funding

Use unnumbered first level headings for the acknowledgments. All acknowledgments go at the end of the paper before the list of references. Moreover, you are required to declare funding (financial activities supporting the submitted work) and competing interests (related financial activities outside the submitted work). More information about this disclosure can be found at: `https://neurips.cc/Conferences/2022/PaperInformation/FundingDisclosure`.

Do **not** include this section in the anonymized submission, only in the final paper. You can use the `ack` environment provided in the style file to autmoatically hide this section in the anonymized submission.

[1]`https://github.com/yingfan-bot/mbpsrl`
[2]`https://github.com/dongminlee94/deep_rl`
[3]`https://github.com/colehawkins/KSD-Thinning`
[4]`https://github.com/wilson-ye-chen/stein_thinning`