IS MODEL ENSEMBLE NECESSARY? MODEL-BASED RL VIA A SINGLE MODEL WITH LIPSCHITZ REGULARIZED VALUE FUNCTION

ABSTRACT

Probabilistic dynamics model ensemble is widely used in existing model-based reinforcement learning methods as it outperforms a single dynamics model in both asymptotic performance and sample efficiency. In this paper, we provide both practical and theoretical insights on the empirical success of the probabilistic dynamics model ensemble through the lens of Lipschitz continuity. We find that, for a value function, the stronger the Lipschitz condition is, the smaller the gap between the true dynamics- and learned dynamics-induced Bellman operators is, thus enabling the converged value function to be closer to the optimal value function. Hence, we hypothesize that the key functionality of the probabilistic dynamics model ensemble is to regularize the Lipschitz condition of the value function using generated samples. To validate this hypothesis, we devise two practical robust training mechanisms through computing the adversarial noise and regularizing the value network's spectral norm to directly regularize the Lipschitz condition of the value functions. Empirical results show that combined with our mechanisms, model-based RL algorithms with a single dynamics model outperform those with an ensemble of probabilistic dynamics models. These findings not only support the theoretical insight, but also provide a practical solution for developing computationally efficient model-based RL algorithms.

1 INTRODUCTION

Model-based reinforcement learning (*MBRL*) improves the sample efficiency of an agent by learning a model of the underlying dynamics in a real environment. One of the most fundamental questions in this area is how to learn a model to generate good samples so that it maximally boosts the sample efficiency of policy learning. To address this question, various model architectures are proposed such as Bayesian nonparametric models (Kocijan et al., 2004; Nguyen-Tuong et al., 2008; Kamthe & Deisenroth, 2018), inverse dynamics model (Pathak et al., 2017; Liu et al., 2022), multi-step model (Asadi et al., 2019), and hypernetwork (Huang et al., 2021).

Among these approaches, the most popular and common approach is to use an ensemble of probabilistic dynamics models (Janner et al., 2019; Lai et al., 2020; Clavera et al., 2020; Froehlich et al., 2022; Li et al., 2022). It is first proposed by Chua et al. (2018) to capture both the aleatoric uncertainty of the environment and the epistemic uncertainty of the data. In practice, MBRL methods with an ensemble of probabilistic dynamics models can often achieve higher sample efficiency and asymptotic performance than using only a single dynamics model.

However, while the uncertainty-aware perspective seems reasonable, previous works only directly apply probabilistic dynamics model ensemble in their methods without an in-depth exploration of why this structure works. There still lacks enough theoretical evidence to explain the superiority of probabilistic neural network ensemble. In addition, extra computation time and resources are needed to train an ensemble of neural networks.

In this paper, we provide a new perspective on why training a probabilistic dynamics model ensemble can significantly improve the performance of the model-based algorithm. We find that the ensemble model-generated state transition samples that are used for training the policies and the critics are much more "diverse" than samples generated from a single deterministic model. we hypothesize that it implicitly regularizes the Lipschitz condition of the critic network especially over a local region where the model is uncertain (i.e., *a model-uncertain local region*). Therefore, the Bellman operator induced by the learned transition dynamics will yield an update to the agent's value

function close to the true underlying Bellman operator's update. We provide systematic experimental results and theoretical analysis to support our hypothesis.

Based on this insight, we propose two simple yet effective robust training mechanisms to regularize the Lipschitz condition of the value network. The first one is *spectral normalization* (Miyato et al., 2018) which provides a global Lipschitz constraint for the value network. It directly follows our theoretical insights to explicitly control the Lipschitz constant of the value function. However, based on both of our theoretical and empirical observations, only the local Lipschitz constant around a model-uncertain local region is required for a good performance. So we also propose the second mechanism, *robust regularization*, which stabilizes the local Lipschitz condition of the value network by computing the adversarial noise with fast gradient sign method (FGSM) (Ian J. Goodfellow, 2015). To compare the effectiveness of controlling global versus local Lipschitz, systematic experiments are implemented. Our methods are orthogonal to most of the state-of-the-art (SOTA) MBRL algorithms and are "plug-and-play" modules that can be combined with those SOTA algorithms. Experimental results on five MuJoCo environments verify that the proposed Lipschitz regularization mechanisms with single deterministic dynamics model improves SOTA performance of the probabilistic ensemble MBRL algorithms, using less computational time and resources.

Our contributions are summarized as follows. **(1)** We propose a new insight into why an ensemble of probabilistic dynamics models can significantly improve the performance of the MBRL algorithms, supported by both theoretical analysis and experimental evidence. **(2)** We introduce two robust training mechanisms for MBRL algorithms, which directly regularize the Lipschitz condition of the agent's value function. **(3)** Experimental results on five MuJoCo tasks demonstrate the effectiveness of our proposed mechanisms and validate our insights, improving SOTA asymptotic performance using less computational time and resources.

2 Preliminaries and Background

Reinforcement learning. We consider a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{P}_0, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state space and action space respectively, $\mathcal{P}(s'|s,a)$ is the transition dynamics, $\mathcal{P}_0$ is the initial state distribution, $r(s,a)$ is the reward function and γ is the discount factor. In this paper, we focus on value-based reinforcement learning algorithms. Define the *optimal Bellman operator* $\mathcal{T}^*$ such that $\mathcal{T}^*Q(s,a) = r(s,a) + \gamma \int \mathcal{P}(s'|s,a) \arg\max_{a'} Q(s',a')ds'$. The goal of the value-based algorithm is to learn a Q_ϕ to approximate the optimal state-action value function Q^* where $Q^* = \mathcal{T}^*Q^*$ is the fixed point of the optimal Bellman operator.

For model-based RL (MBRL), we denote the approximate model for state transition and reward function as $\hat{\mathcal{P}}_\theta$ and $\hat{r}_\theta$ respectively. Similarly, we define the *model-induced Bellman Operator* $\widehat{\mathcal{T}}^*$ such that $\widehat{\mathcal{T}}^*Q(s,a) = r(s,a) + \gamma \int \hat{\mathcal{P}}_\theta(s'|s,a) \arg\max_{a'} Q(s',a')ds'$.

Probabilistic dynamics model ensemble. Probabilistic dynamics model ensemble consists of K neural networks $\hat{\mathcal{P}}_\theta = \{\hat{\mathcal{P}}_{\theta_1}, \hat{\mathcal{P}}_{\theta_2}, ..., \hat{\mathcal{P}}_{\theta_K}\}$ with the same architecture but randomly initialized with different weights. Given a state-action pair (s,a), the prediction of each neural network is a Gaussian distribution with diagonal covariances of the next state: $\hat{\mathcal{P}}_{\theta_k}(s'|s,a) = \mathcal{N}(\mu_{\theta_k}(s,a), \Sigma_{\theta_k}(s,a))$. The model is trained using negative log likelihood loss (Janner et al., 2019):

$$\mathcal{L}(\theta_k) = \sum_{t=1}^{N} [\mu_{\theta_k}(s_t,a_t) - s_{t+1}]^\top \Sigma_{\theta_k}^{-1}(s_t,a_t)[\mu_{\theta_k}(s_t,a_t) - s_{t+1}] + \log\det \Sigma_{\theta_k}(s_t,a_t) \tag{1}$$

During model rollouts, the probabilistic dynamics model ensemble first randomly selects a network from the ensemble and then samples the next state from the predicted Gaussian distribution.

Local Lipschitz constant. We now give the definition of local Lipschitz constant, a key concept in the following discussion.

Definition 2.1. *Define **local** (ϵ, p)**-Lipschitz constant** of a scalar valued function $f : \mathbb{R}^N \to \mathbb{R}$ as:*

$$L^{(\epsilon,p)}(f) = \sup_{B^{(p)}(x,\epsilon)} \sup_{y_1,y_2 \in B^{(p)}(x,\epsilon)} \frac{|f(y_1) - f(y_2)|}{\|y_1 - y_2\|_p} \quad (y_1 \neq y_2). \tag{2}$$

If $L^{(\epsilon,p)}(f) = C$ is finite, we say that f is (ϵ, p) locally Lipschitz with constant C. Here, $B^{(p)}(x,\epsilon) = \{x' \mid \|x' - x\|_p \leq \epsilon\}$ denotes the L_p norm ball of radius ϵ around x.

It is clear from this definition that given $\epsilon_1 \leq \epsilon_2$, $L^{(\epsilon_1,p)}(f) \leq L^{(\epsilon_2,p)}(f)$. In particular, we can view the **global Lipschitz constant** as $L^{(\infty,p)}$ which upper bounds the local Lipschitz constant.

3 Local Lipschitz Condition of Value Functions in MBRL

3.1 Key Insight: Implicit Regularization on Value Network by Probabilistic Ensemble Model

In this section, we provide insight into why a value-based reinforcement learning algorithm trained from simulated samples generated by a probabilistic ensemble model performs well in practice. We first conduct an experiment on the MuJoCo Humanoid environment with MBPO (Janner et al., 2019), the SOTA model-based algorithm using Soft Actor-Critic (SAC) (Haarnoja et al., 2018) as the backbone algorithm for policy and value optimization.We run MBPO with four different types of trained environment models: (1) a single deterministic dynamics model, (2) a single probabilistic dynamics model, (3) an ensemble of seven deterministic dynamics models, and (4) an ensemble of seven probabilistic dynamics models. As we can see from Figure 1a, in the Humanoid environment, the agent trained from an ensemble of the probabilistic dynamics model indeed achieves much better performance. A similar observation is found across all the other MuJoCo environments.

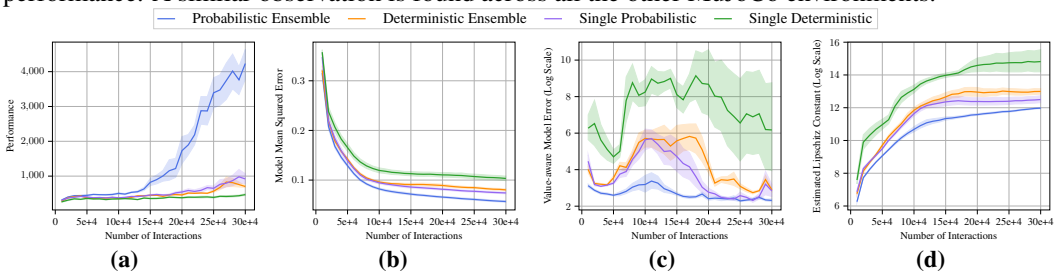


Figure 1: **(a)** Performance of MBPO algorithm trained with single deterministic, single probabilistic, deterministic ensemble, and probabilistic ensemble model respectively on Humanoid. **(b)** Model mean squared error. **(c)** Value-aware model error in log scale. **(d)** Upper bound of Lipschitz constant of value network. Results are averaged over 8 random seeds.

Value-aware model error. Why does MBPO with an ensemble of probabilistic dynamics models perform much better? We try to answer this question by considering the effects of the learned transition models on the agent's value functions. In MBPO algorithm, it fits an ensemble of K Gaussian environment models $\hat{\mathcal{P}}_\theta^{(i)}(\cdot|s,a) \equiv \mathcal{N}\big(\mu_\theta^{(i)}(s,a), \Sigma_\theta^{(i)}(s,a)\big), i = 1, ..., K$ using the negative log-likelihood loss in Equation (1). With a deep neural network as a powerful function approximator for the mean and variance of the Gaussian distribution, the mean squared error $\mathbb{E}_{(s,a)\sim\rho, s'\sim\mathcal{P}(\cdot|s,a), \hat{s}'\sim\hat{\mathcal{P}}_\theta(\cdot|s,a)}[\|s' - \hat{s}'\|^2]$ is often small. However, as argued in many previous MBRL works (Grimm et al., 2020; 2021), the mean squared error is often not a good measurement for the quality of the learned transition model. Instead, the *value-aware model error* (Farahmand et al., 2017a) as defined below plays a crucial role here, which connects directly to the suboptimality of the MBRL algorithm.

$$
\begin{aligned}
\mathcal{L}_{vame}(\hat{\mathcal{P}}_\theta; Q, \rho) &= \mathbb{E}_{(s,a)\sim\rho}\big[\big(\mathcal{T}^* Q(s,a) - \widehat{\mathcal{T}}^* Q(s,a)\big)^2\big] \\
&= \gamma^2 \mathbb{E}_{(s,a)\sim\rho}\Big[\Big(\int \mathcal{P}(s'|s,a)V(s')ds' - \int \hat{\mathcal{P}}_\theta(\hat{s}'|s,a)V(\hat{s}')d\hat{s}'\Big)^2\Big] \\
&\leq \gamma^2 \mathbb{E}_{(s,a)\sim\rho, s'\sim\hat{\mathcal{P}}(\cdot|s,a), \hat{s}'\sim\hat{\mathcal{P}}_\theta(\cdot|s,a)}\big[\big(V(\hat{s}') - V(s')\big)^2\big],\ V(s) = \max_a Q(s,a)
\end{aligned}
$$

This value-aware model error measures the difference between the simulated and true Bellman operator acting on a given Q function. In other words, even though the learned transition model $\hat{\mathcal{P}}_\theta$ is approximately accurate, the model prediction error can still be amplified by the value function so that the two Bellman operators yield completely different updates on the value function.

Value function Lipschitz regulates value-aware model error. When the value function's local-Lipschitz constant is under control, the value-aware model error can be made small. As we see from Figure 1b, 1c, and 1d, although the probabilistic ensemble model and single deterministic model achieve similar mean squared errors, value functions trained from an ensemble of probabilistic dynamics model has a significantly smaller Lipschitz constant and value-aware model error. (Note that the value-aware model error plotted in Figure 1d is in log scale.)

To see why this is the case, we view the learned Gaussian transition models $\hat{\mathcal{P}}_\theta^{(i)}$ as $f_\theta^{(i)} + g_\theta^{(i)}$, where $f_\theta^{(i)}$ is a deterministic model and $g_\theta^{(i)}(\cdot|s,a) \equiv \mathcal{N}(0, \Sigma_\theta^{(i)}(s,a))$ is a noise distribution with zero mean. When training the value network of the MBRL agent, the target value is computed as $r(s,a)+\gamma\mathbb{E}_{i\sim\text{Cat}(K,\frac{1}{K}),\epsilon_i\sim g_\theta^{(i)}(\cdot|s,a)}[V\big(f_\theta^{(i)}(s,a)+\epsilon_i\big)]$. Here we view data generated from the probabilistic ensemble model as an implicit augmentation. The augmentation comes from two sources: (i) variation of prediction across different models in the ensemble and (ii) the noise added by each noise distribution $g_\theta^{(i)}$. By augmenting the transition with a diverse set of noise and then training the value functions with such augmented samples, it implicitly regularizes the local Lipschitz condition of the value network over the local region around the state where the model prediction is uncertain.

In the next section, we provide some theoretical insights into how the local Lipschitz constant can play a role in the suboptimality of the MBRL algorithms. Later in the algorithm and experiment section, we will provide two practical mechanisms to regularize the Lipschitz condition of the value network and demonstrates the effectiveness of such mechanisms, which further validates our claim.

3.2 Error Bound of Model-based Value Iteration with locally Lipschitz value functions

In this section, we formally analyze how the Lipschitz constant of the value function affects the learning dynamics of the model-based (approximate) value iteration algorithm, the prototype for most of the value-based MBRL algorithms. For simplicity, following previous works (Farahmand et al., 2017b; Grimm et al., 2020), we assume that we have the true reward function $r(s,a)$, but extending the results to reward function approximation should be straightforward.

We consider the following value iteration algorithm. At k th-iteration of the algorithm, we obtain a dataset $\mathcal{D}_k \triangleq \{(s_i,a_i,s_i')\}_{i=1}^N$, where (s_i,a_i) is sampled i.i.d. from $\rho \in \Delta(\mathcal{S}\times\mathcal{A})$, the empirical state-action distribution, and $s_i' \sim \mathcal{P}(\cdot|s_i,a_i)$ is the next state under the environment transition. Based on this dataset, we first approximate the transition kernel $\hat{\mathcal{P}}$ to minimize the mean L_2 difference between the predicted and true next states.

$$\hat{\mathcal{P}}_k \leftarrow \arg\min_{\hat{\mathcal{P}}\in\mathcal{M}} \frac{1}{N}\sum_{i=1}^N \int \hat{\mathcal{P}}(\hat{s}'|s_i,a_i)\big\|\hat{s}'-s_i'\big\| d\hat{s}' \tag{3}$$

Then at k th-iteration, we update the value function by solving the following regression problem:

$$\hat{Q}_k \leftarrow \arg\min_{\hat{Q}\in\mathcal{F}} \mathcal{L}_{reg}(\hat{Q};\hat{Q}_{k-1},\hat{\mathcal{P}}_k) := \arg\min_{\hat{Q}\in\mathcal{F}} \mathbb{E}_{(s,a)\sim\rho}\Big[\big(\hat{Q}(s,a)-\hat{\mathcal{T}}^*\hat{Q}_{k-1}(s,a)\big)^2\Big] \tag{4}$$

Empirically, we update the value function such that

$$\hat{Q}_k \leftarrow \arg\min_{\hat{Q}\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^N \Big|\hat{Q}(s_i,a_i) - \Big(r_i + \gamma\int \hat{\mathcal{P}}_k(\hat{s}'|s_i,a_i)\hat{V}_{k-1}(\hat{s}')d\hat{s}'\Big)\Big|^2 \tag{5}$$

where $\hat{V}_{k-1}(\hat{s}') = \max_{a'}\hat{Q}_{k-1}(\hat{s}',a')$.

To simplify the analysis, we assume that we are given a fixed state-action distribution ρ such that for every iteration, we can sample i.i.d. from this data distribution. However, in practice, we may use a different data collecting policy at different iterations. As argued in (Farahmand et al., 2017a), a similar result can be shown in this case by considering the mixing behavior of the stochastic process.

Now we list the assumptions we make. Some assumptions are made only to simplify the finite sample analysis, while others characterize the crucial aspects of model and value learning.

First, we make the deterministic assumption of the environment. This is only for the purpose of the finite sample analysis. When the environment transition is stochastic, we will not have the finite sample guarantee, but our insights still hold. We will provide more discussion on this later when we present our finite sample guarantee.

Assumption 3.1. *The environment transition is deterministic.*

Next, to apply the concentration inequality in our analysis, we have to make the following technical assumption of the state space and reward function.

Assumption 3.2. ***(Boundedness of State Space and Reward Function)*** *There exists constants D, $R_{\max}$ such that for all $s\in\mathcal{S}, a\in\mathcal{A}$, $\|s\|_2 \le D$ and $r(s,a) < R_{\max}$.*

In addition, we make the following mild approximate realizability assumption on the model class of approximated transition kernels so that in the model space, at least one transition model close to the true underlying transition kernel should exist.

Assumption 3.3. $\big($***(ϵ, ρ)-Approximate Realizability***$\big)$

$$\inf_{\hat{\mathcal{P}} \in \mathcal{M}} \mathcal{L}_2(\hat{\mathcal{P}}) := \inf_{\hat{\mathcal{P}} \in \mathcal{M}} \int \hat{\mathcal{P}}(d\hat{s}'|s,a) \big\| \mathcal{P}(s,a) - \hat{s}' \big\| d\rho(s,a) \leq \epsilon \tag{6}$$

We also make a critical assumption of the local Lipschitz condition of the value function class. In particular, for the state-action value function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, we define that Q is (ϵ, p)-locally Lipschitz with constant L if for every $a \in \mathcal{A}$, the function $Q_a : s \mapsto Q(s,a)$ is (ϵ, p)-locally Lipschitz with constant less than or equal to L.

Assumption 3.4. ***(Local Lipschitz condition of value functions)*** *There exists a finite L, such that for every $Q \in \mathcal{F}$, it's $\big(2\epsilon, 2\big)$-locally Lipschitz with a constant less than or equal to L.*

Note here we only need to assume that value functions are all $((1+\beta)\epsilon, 2)$-locally Lipschitz with $\beta > 0$. We set $\beta = 1$ for simplicity. Finally, same as the previous work (Farahmand et al., 2017a), to provide a finite sample guarantee of model learning, we make the following assumption on the complexity of the model space of the approximated transition kernels

Assumption 3.5. ***(Complexity of Model Space)*** *Let $R > 0$, $J : \mathcal{M}_0 \to [0, \infty)$ be a pseudo-norm, where $\mathcal{M}_0$ is a space of transition kernels. Let $\mathcal{M} = \mathcal{M}_R = \{\mathcal{P} : J(\mathcal{P}) \leq R\}$. There exists constants $c > 0$ and $0 < \alpha < 1$ such that for any $\delta, R > 0$ and all sequence of state-action pairs $z_{1:n} \triangleq z_1, ..., z_n \in \mathcal{S} \times \mathcal{A}$, the following metric entropy condition is satisfied:*

$$\log \mathcal{N}(\epsilon, \mathcal{M}, L_2(z_{1:n})) \leq c\Big(\frac{R}{\delta}\Big)^{2\alpha} \tag{7}$$

where $\mathcal{N}(\delta, \mathcal{M}, L_2(z_{1:n}))$ is the covering number of $\mathcal{M}$ with respect to the empirical norm $L_2(z_{1:n})$ such that $\|\mathcal{P}\|^2_{2,z_{1:n}} = \dfrac{1}{N}\sum_{i=1}^{N} \|\mathcal{P}(\cdot|z_i)\|_2^2$

Under these assumptions, we can now present our main theorem, which relates the local Lipschitz constant to the suboptimality of the approximate model-based value iteration algorithm. In particular, we provide a finite sample analysis of model learning in Theorem B.2 and value-aware model error in Theorem B.4. Then we apply the error accumulation results from (Farahmand et al., 2017a), connecting the local Lipschitz constant with the suboptimality of the algorithm.

Theorem 3.6. *Suppose $\hat{Q}_0$ is initialized such that $\hat{Q}_0(s,a) \leq \dfrac{R_{\max}}{1-\gamma}$ for $\forall\ (s,a)$. Under the assumptions of 3.1, 3.2, 3.3, 3.4, and 3.5, after K iterations of the model-based approximate value iteration algorithm, there exists a constant $\kappa(\alpha)$ which depends solely on $\alpha \in (0,1)$ such that,*

$$\mathbb{E}_{(s,a)\sim\mathcal{P}_0}\Big[\Big|Q^*(s,a) - \hat{Q}_K(s,a)\Big|\Big] \leq \frac{2\gamma}{(1-\gamma)^2}\Big[C(\rho,\mathcal{P}_0)\Big(\max_{0\leq k\leq K}\delta_k + \gamma^2\big(4\epsilon^2L^2\xi + \frac{(1-\xi)R^2_{\max}}{(1-\gamma)^2}\big)\Big) + 2\gamma^K R_{\max}\Big] \tag{8}$$

where $\delta_k^2 = \mathcal{L}_{reg}(\hat{Q}_k; \hat{Q}_{k-1}, \hat{\mathcal{P}}_k)$ is the regression error defined in Equation (4), $\xi = 1 - \exp(-\dfrac{\epsilon N^{\frac{1}{1+\alpha}}}{\kappa(\alpha)D^2R^{\frac{2\alpha}{1+\alpha}}})$, and $C(\rho,\mathcal{P}_0)$ is the concentrability constant defined in Definition A.1.

Remarks.

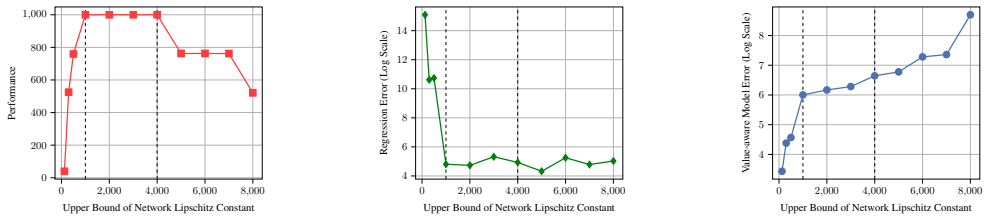
(1) As the number of samples $N \to \infty$, $\xi \to 1$. Consequently, the value-aware model error $\mathbb{E}\big[|\mathcal{T}^*Q_k(s,a) - \widehat{\mathcal{T}}^*Q_k(s,a)|^2\big]$ will be governed by the term $4\epsilon^2L^2\xi$ which is controlled by the local Lipschitz constant L. In addition, if the number of iterations $k \to \infty$, the left hand side of Equation (8) will be bounded by $\dfrac{2\gamma C(\rho,\mathcal{P}_0)}{(1-\gamma)^2}\Big(\max_{0\leq k\leq K}\delta_k + 4\epsilon^2L^2\xi\Big)$. From here, we can see the role that the local Lipschitz constant has played in controlling the suboptimality of the algorithm.

(2) Although we assume that the environment transition is deterministic, our insights should still hold when it is stochastic. For In particular, if the learned transition model has a low prediction error $\mathcal{L}_2(\hat{\mathcal{P}})$ (Eq. 6) (which is often the case with a neural network as the function approximator), and the

local Lipschitz constant of the value function is bounded, then we can still have a small value-aware model error and get a similar error propagation result as Theorem 3.6.

Tradeoff between regression error and value-aware model error. This theorem also reveals an essential trade-off between the regression error and value-aware model error through *the Lipschitz condition of the value function class*, i.e., the constant L. As L gets smaller, the model-induced Bellman operator gets closer to the actual underlying Bellman operator. However, with a smaller L, we also impose a stronger condition on the value function class. Therefore, the value function space will shrink, and the regression error δ_k is expected to get larger.

To further visualize this trade-off, we conduct an experiment on the Inverted-Pendulum environment, where we run the analyzed model-based approximate value iteration algorithms for 1000 iterations. We plot (1) the best evaluation performance for 20 episodes at every iteration (Figure 2a), (2) the maximum regression error across every iteration (Figure 2b), as well as (3) the maximum value-aware model error (Figure 2c). To upper bound the (local) Lipschitz constant, we constrain the spectral norm of the weight matrix for each layer of the value network.



(a) best evaluation performance **(b)** max regression error **(c)** max value-aware model error

Figure 2: Model-based value iteration on Inverted-Pendulum across value networks with different Lipschitz constraints. All the results are the median over 5 random seeds.

As we see from the results in Figure 2b, indeed, the regression error dramatically drops as the Lipschitz constant grows from 100 to 1000 and then levels off, which indicates that perhaps the Lipschitz constant of 1000 is rich enough for the value function class to be Bellman-complete for this environment. Meanwhile, the value-aware model error also increases with a bigger Lipschitz constant. However, we can achieve a good balance between these two errors when the Lipschitz constants are between 1000 and 4000, where the algorithm is observed to perform the best.

4 METHODS

In this section, we present two different approaches to regularize the (local) Lipschitz constant of the value function.

4.1 SPECTRAL NORMALIZATION

First, to explicitly control the upper bound of the Lipschitz constant, we adapt the technique of Spectral Normalization, which is originally proposed to stabilize the training of Generative Adversarial Network (GAN) (Miyato et al., 2018). By controlling the spectral norm of the weight matrix at every layer of the value network, its Lipschitz constant is upper bounded. In particular, during each forward pass, we approximate the spectral norm of the weight matrix $\|W\|_2$ with one step of power iteration. Then, we perform a projection step so that its spectral norm will be clipped to β if bigger than β, and unchanged otherwise. See Appendix C for more computational details.

4.2 ROBUST REGULARIZATION

Spectral Normalization directly bounds the global Lipschitz constant of the value function. However, as we argued in Section 3.2, we only require the local Lipschitz conditions around the model-uncertain local region. Such a strong regularization is not necessary and can even negatively impact the expressive power of value function. Now we present an alternative approach to quickly regularize the local Lipschitz constant of the value function based on a robust training mechanism.

To regularize the local Lipschitz condition of the value network, we minimize the following loss:

$$\mathcal{L}_{\text{reg}}^{(\alpha)}(Q_\phi; \epsilon, \pi_\psi, \{s_i, a_i, s_i'\}_{i=1}^D) = \sum_{i=1}^{D} \max_{\|s_i' - s_i\|_\alpha \le \epsilon} \Big(Q_\phi\big(s_i', \pi_\psi(s_i')\big) - Q_\phi(s_i, a_i) \Big)^2 \tag{9}$$

This robust loss is to guarantee that the variation of the value function locally is small. Then we combine it with the original loss of the value function $\mathcal{L}_{org}(Q_\phi; \pi_\psi, \{s_i, a_i, s_i'\}_{i=1}^D)$ by minimizing $\mathcal{L}(Q_\phi; \pi_\psi, \{s_i, a_i, s_i'\}_{i=1}^D) = \mathcal{L}_{org}(Q_\phi; \pi_\psi, \{s_i, a_i, s_i'\}_{i=1}^D) + \lambda \mathcal{L}_{\text{reg}}^{(\alpha)}(Q_\phi; \epsilon, \pi_\psi, \{s_i, a_i, s_i'\}_{i=1}^D)$.

Here, λ and ϵ are two hyperparameters. A larger ϵ is required for the dynamics model which has larger prediction errors. In terms of λ, a bigger λ makes the value network varies smoother over the local region and gives better convergence guarantees, but it also hurts the expressive power of the value network. In practice, we find that using $\epsilon = 0.1$ is often enough for good performance. So we fix ϵ and search for the best λ. For a detailed discussion on the choice of λ s, see Appendix D.4.

To solve the perturbation s' in the constrained optimization within the robust loss, we use the fast gradient sign method (FGSM) (Ian J. Goodfellow, 2015). Here, let (s, a) be the state action pair. Then we compute $s' = \text{proj}\left(s_0 + \epsilon \,\text{sign}\left(\nabla_{s'}\big|_{s'=s_0}\left(Q_\phi\left(s', \pi_\psi(s')\right) - Q_\phi(s,a)\right)^2\right), B^{(\alpha)}(s_0, \epsilon)\right)$, where $s_0 \sim \mathcal{N}(s, \epsilon^2 I)$. Here the purpose of random initialization of s_0 is because the gradient $\nabla_{s'}\left(Q_\phi\left(s', \pi_\psi(s')\right) - Q_\phi(s,a)\right)^2)$ vanishes at s. In our experiments, we focus on the l_∞ norm, but our proposed robust loss should be applicable to any l_p norm.

In addition, we can apply more advanced constrained optimization solvers such as Projected Gradient Decent (PGD). But here, using FGSM together with a single deterministic environment model is mainly for efficiency purposes. In practice, we find that using PGD does not give much performance gain over FGSM method, and we refer the readers to Appendix D.3 for the experimental results.

5 Experiment

5.1 Empirical Evaluations of Proposed Mechanisms

We evaluate our two proposed training mechanisms on five MuJoCo tasks, including Walker, Humanoid, Ant, Hopper, and HalfCheetah. We compare the two training mechanisms with MBPO (Janner et al., 2019) using an ensemble of probabilistic transition models. We do not compare our methods with MBPO using an ensemble of deterministic transition models mainly because it is ensemble-based and across all five tasks, it is outperformed by probabilistic ensemble models. In addition, since our regularization mechanisms are only trained on top of a single deterministic transition model, we compare it with both MBPO using a single deterministic transition model and a single probabilistic model. We implement our methods and the baseline methods based on a PyTorch implementation of MBPO (Lin, 2022). More implementation details are provided in Appendix C.

Improved asymptotic performance. Figure 3 presents the learning curves for all methods. These results show that using just a single deterministic model, MBPO with our two Lipschitz regularization mechanisms achieves a comparable and even better performance across all five tasks than MBPO with a probabilistic ensemble model. In particular, the proposed robust regularization technique shows a larger advantage on three more sophisticated tasks: Humanoid, Ant, and Walker. For example, on Humanoid, it achieves the same final performance as that of a probabilistic ensemble with only about 60% of the environment interaction steps. In contrast, spectral normalization shows little improvement over a single deterministic model on these three tasks, showing the limitation of constraining the global Lipschitz constant. We provide more discussion on this in Section 5.3.

Time Efficiency. Compared with MBPO using an ensemble of probabilistic models, our proposed training mechanisms, especially robust regularization, is more time efficient. In Table 1 we record the wall clock time of the algorithms trained for 200,000 environment interaction steps on Walker and Ant, averaged over 8 random seeds. We see that robust regularization is much faster than spectral normalization. While spectral normalization takes slightly less the amount of time than probabilistic ensemble, robust regularization only takes about 70% of the computational time.

The significance of the experimental results is twofold. First, it further validates our insights and shows the importance of the local Lipschitz condition of value functions in MBRL. Second, it demonstrates that having an ensemble of transition models is not necessary. We can save the computational time and cost of training an ensemble of transition models

Table 1: Comparison of computational time.

Algorithm	Walker	Ant
	Time (h)	Time(h)
Single Deterministic	29.6	34.0
Probabilistic Ensemble	60.0	62.3
Robust Regularization	45.4	48.3
Spectral Normalization	54.2	57.1

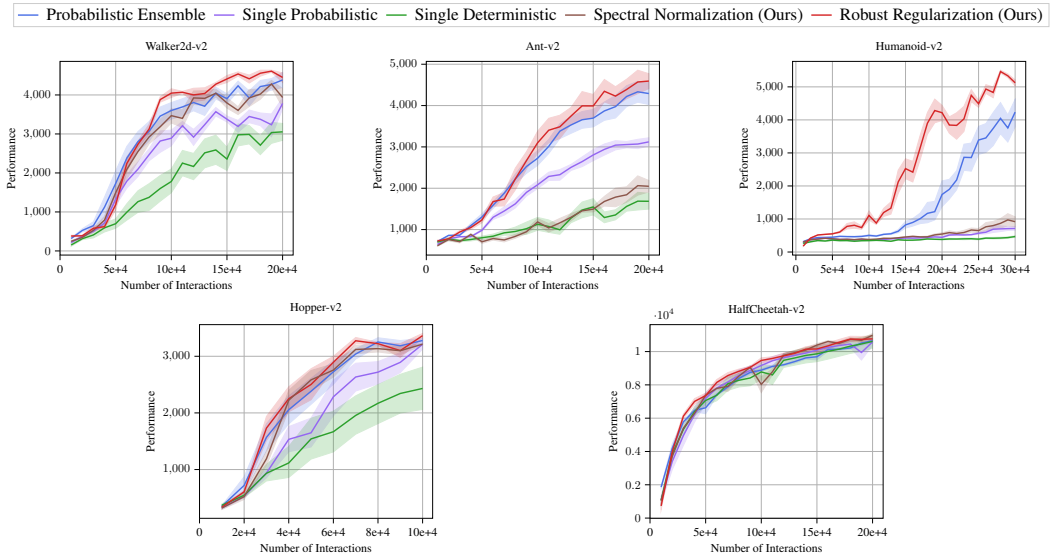


Figure 3: Performance of the proposed value function training mechanisms against baselines. Results are averaged over 8 random seeds and shaded regions correspond to the 95% confidence interval among seeds.

with a simple Lipschitz regularization mechanism. In practice, we can even use a single probabilistic model combined with our proposed mechanisms to get the best performance. See Appendix D.2 for the extra experimental results.

5.2 Visualizing the Value-aware Model Error

Given the excellent performance of robust regularization, we now verify its effectiveness in controlling the value-aware model error. On Walker, we compare it with a variant: computing the perturbation with uniform random noise instead of adversarially choosing the perturbation.

As we can see from the first figure in Figure 4, robust regularization has a much smaller value-aware model error than all other methods. Adding uniformly random noise can somewhat reduce the value-aware model error compared with a single deterministic model (without any noise added). However, it is still much less effective than robust regularization, which computes the error adversarially. In the second figure of Figure 4, we see that with uniform random noise, it achieves a slightly better performance than a single deterministic model but is still far worse than robust regularization. The results further verify that by adversarially choosing the noise, robust regularization is extremely effective at controlling the value-aware model error, resulting in great empirical performance. In Appendix D.1, we provide additional experimental results of value-aware model error in the rest of the four environments.

5.3 Robust Regularization vs. Spectral Normalization

In Table 1, we already see that robust regularization achieves a much better time efficiency than spectral normalization. Now we analyze their performance difference through the lens of value-aware model error. To reduce the value-aware model error, we only need to control the local Lipschitz constant of the value network over the model-uncertain region, and thus controlling the global Lipschitz constant of the value function with spectral normalization is not necessary. To see this, in the last two figures of Figure 4 , we visualize the value-aware model error and performance of the algorithm on Walker with varying spectral radius β, defined in Section 4.1.

As we see from the plot, to reduce the value-aware model error so that it is about the same as robust regularization, we need to choose the spectral radius β as small as 10. However, under this constraint, it achieves a significantly worse performance than robust regularization, indicating that this constraint is perhaps too strong. To get the best performance, spectral normalization has to use a larger spectral radius to trade value-aware model error for expressive power and thus has worse empirical performance. In addition, we observe from Figure 3 that the performance discrepancy

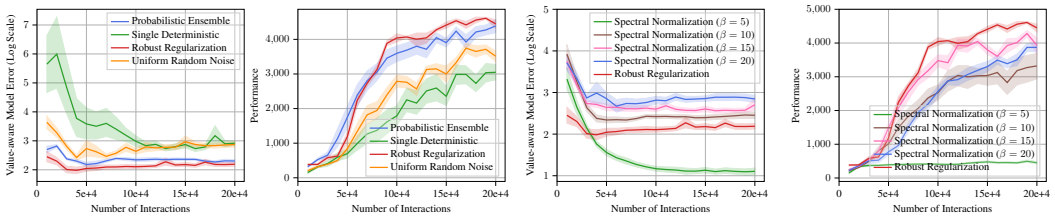


Figure 4: (Left Two) Robust Regularization with a comparison of uniform random noise on Walker. (Right Two) Spectral Normalization with different spectral radius β on Walker.

between the two methods is even greater in more complicated environments such as Ant and Humanoid, which further shows the limitation of constraining the global Lipschitz constant.

6 Related Work

Dynamics ensemble in model-based reinforcement learning. Dynamics model ensemble is first introduced in MBRL by Kurutach et al. (2018) to avoid the learned policy exploit the insufficient data regions. Then Chua et al. (2018) proposed a probabilistic dynamics ensemble to capture the aleatoric uncertainty and epistemic uncertainty, and achieves significant performance improvement compared to the deterministic dynamics ensemble. Probabilistic dynamics ensemble is now widely used in MBRL methods (e.g. Lai et al. (2020); Clavera et al. (2020); D'Oro & Jaśkowski (2020); Lai et al. (2021); Froehlich et al. (2022); Li et al. (2022)). However, despite its popularity, there is still no clear explanation why the use of probabilistic dynamics ensemble brings such a large improvement to the policy. In this paper, we fill this gap and propose a novel theoretical explanation on why probabilistic dynamics ensemble works well.

Lipschitz continuity in reinforcement learning. In the realm of model-free reinforcement learning, a few recent works also apply the technique of spectral normalization to regularize the Lipschitz condition of the RL agent's value function, which results in more stable optimization dynamics (Gogianu et al., 2021; Bjorck et al., 2021). In addition, Shen et al. (2020); Zhang et al. (2020) propose to use a similar robust regularizer as the robust loss in our robust regularization method, aiming at an adversarially robust policy. In our work, we focus on MBRL, a fundamentally different setting. As we explained in Section 3.1, if no restrictions are imposed on the value function class, then the model error can be amplified by the value function, making the induced Bellman operator diverge from the true underlying Bellman operator.

In MBRL, Asadi et al. (2018) proposed to learn a generalized Lipschitz transition model with respect to the Wasserstein metric, resulting in a bounded multi-step model prediction error and Lipschitz optimal value function induced from the learned dynamics. In contrast, our paper focuses on the local Lipschitz condition of the value function instead of the underlying transition model, studying its relation with the suboptimality of MBRL algorithm.

7 Conclusion and Discussion

In this paper, we provide insight into why the probabilistic ensemble model can achieve great empirical performance. We demonstrate the importance of the local Lipschitz condition of value functions in MBRL algorithms and justify our hypothesis with both theoretical and empirical results. Based on our insight, we propose two training mechanisms that directly regularize the Lipschitz condition of the value function. Empirical studies demonstrate the effectiveness of the proposed mechanisms. One consequence of our paper is that if the model ensemble is not used, then many existing methods of estimating model uncertainty based on ensemble will be unusable, and we need to redesign the uncertainty estimation method based on single model. We leave this as future work.

REFERENCES

Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 264–273. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/asadi18a.html.

Kavosh Asadi, Dipendra Misra, Seungchan Kim, and Michel L Littman. Combating the compounding-error problem with a multi-step model. *arXiv preprint arXiv:1905.13320*, 2019.

Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005. ISSN 00905364. URL http://www.jstor.org/stable/3448616.

Nils Bjorck, Carla P Gomes, and Kilian Q Weinberger. Towards deeper deep reinforcement learning with spectral normalization. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8242–8255. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/4588e674d3f0faf985047d4c3f13ed0d-Paper.pdf.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in Neural Information Processing Systems*, 31, 2018.

Ignasi Clavera, Yao Fu, and Pieter Abbeel. Model-augmented actor-critic: Backpropagating through paths. In *International Conference on Learning Representations*, 2020.

Pierluca D'Oro and Wojciech Jaśkowski. How to learn a useful critic? model-based action-gradient-estimator policy optimization. *Advances in Neural Information Processing Systems*, 33:313–324, 2020.

Amir-Massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-Aware Loss Function for Model-based Reinforcement Learning. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1486–1494. PMLR, 20–22 Apr 2017a. URL https://proceedings.mlr.press/v54/farahmand17a.html.

Amir-massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-aware loss function for model-based reinforcement learning. In *Artificial Intelligence and Statistics*, pp. 1486–1494. PMLR, 2017b.

Lukas Froehlich, Maksym Lefarov, Melanie Zeilinger, and Felix Berkenkamp. On-policy model errors in reinforcement learning. In *International Conference on Learning Representations*, 2022.

Florin Gogianu, Tudor Berariu, Mihaela C Rosca, Claudia Clopath, Lucian Busoniu, and Razvan Pascanu. Spectral normalisation for deep reinforcement learning: An optimisation perspective. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3734–3744. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/gogianu21a.html.

Christopher Grimm, André Barreto, Satinder Singh, and David Silver. The value equivalence principle for model-based reinforcement learning. *Advances in Neural Information Processing Systems*, 33:5541–5552, 2020.

Christopher Grimm, Andre Barreto, Greg Farquhar, David Silver, and Satinder Singh. Proper value equivalence. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 7773–7786. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/400e5e6a7ce0c754f281525fae75a873-Paper.pdf.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Yizhou Huang, Kevin Xie, Homanga Bharadhwaj, and Florian Shkurti. Continual model-based reinforcement learning with hypernetworks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 799–805. IEEE, 2021.

Christian Szegedy Ian J. Goodfellow, Jonathon Shlens. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32:12519–12530, 2019.

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

Juš Kocijan, Roderick Murray-Smith, Carl Edward Rasmussen, and Agathe Girard. Gaussian process model based predictive control. In *Proceedings of the 2004 American control conference*, volume 3, pp. 2214–2219. IEEE, 2004.

Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018.

Hang Lai, Jian Shen, Weinan Zhang, and Yong Yu. Bidirectional model-based policy optimization. In *International Conference on Machine Learning*, pp. 5618–5627. PMLR, 2020.

Hang Lai, Jian Shen, Weinan Zhang, Yimin Huang, Xing Zhang, Ruiming Tang, Yong Yu, and Zhenguo Li. On effective scheduling of model-based reinforcement learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

Chongchong Li, Yue Wang, Wei Chen, Yuting Liu, Zhi-Ming Ma, and Tie-Yan Liu. Gradient information matters in policy optimization by back-propagating through model. In *International Conference on Learning Representations*, 2022.

Xingyu Lin. mbpo_pytorch. https://github.com/Xingyu-Lin/mbpo_pytorch, 2022.

Minghuan Liu, Zhengbang Zhu, Yuzheng Zhuang, Weinan Zhang, Jianye Hao, Yong Yu, and Jun Wang. Plan your target and learn your skills: Transferable state-only imitation learning via decoupled policy optimization. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Rémi Munos. Error bounds for approximate value iteration. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, pp. 1006. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

Duy Nguyen-Tuong, Jan Peters, and Matthias Seeger. Local gaussian process regression for real time online model learning. *Advances in neural information processing systems*, 21, 2008.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Qianli Shen, Yan Li, Haoming Jiang, Zhaoran Wang, and Tuo Zhao. Deep reinforcement learning with robust and smooth policy. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8707–8718. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/shen20b.html.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21024–21037. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/f0eb6568ea114ba6e293f903c34d7488-Paper.pdf.