Value-Aware Loss Function for Model-based Reinforcement Learning

Amir-massoud Farahmand
Mitsubishi Electric Research Laboratories (MERL), USA

André M.S. Barreto
National Laboratory for Scientific Computing (LNCC), Brazil

Daniel N. Nikovski
Mitsubishi Electric Research Laboratories (MERL), USA

Abstract

We consider the problem of estimating the transition probability kernel to be used by a model-based reinforcement learning (RL) algorithm. We argue that estimating a generative model that minimizes a probabilistic loss, such as the log-loss, is an overkill because it does not take into account the underlying structure of decision problem and the RL algorithm that intends to solve it. We introduce a loss function that takes the structure of the value function into account. We provide a finite-sample upper bound for the loss function showing the dependence of the error on model approximation error, number of samples, and the complexity of the model space. We also empirically compare the method with the maximum likelihood estimator on a simple problem.

1 INTRODUCTION

The standard approach to model-based reinforcement learning (RL) [Sutton and Barto, 1998; Szepesvári, 2010] is to use data $\mathcal{D}_n = \{(X_i, A_i, R_i, X'_i)\}_{i=1}^n$ to estimate the transition probability kernel $\mathcal{P}^*$ by $\hat{\mathcal{P}}$ and the expected reward function r by $\hat{r}$. The learned model is then used to generate new samples, see e.g., [Sutton et al., 2008; Farahmand et al., 2009; Hester and Stone, 2013; Deisenroth et al., 2013, 2015]. A standard RL/Planning algorithm can use these samples to find a close to optimal policy, possibly by first finding an approximation to the optimal (action-)value function. Estimating $\mathcal{P}^*$ by $\hat{\mathcal{P}}$ is the problem of conditional probability (density/distribution) estimation and the estimating r by $\hat{r}$ is a regression problem. In the rest of this work, we only focus on learning $\mathcal{P}^*$.[1]

There are several general approaches to estimate $\mathcal{P}^*$ such as Maximum Likelihood Estimation (MLE), Maximum Entropy (MaxEnt) estimation, the Maximum A Posteriori (MAP) estimation, and Bayesian posterior inference. We argue that these conventional approaches to find a generative model might be an overkill, thus may not be required.

For example, consider the ML estimate, which is the minimizer of the empirical negative-log loss, which in turn is an empirical approximation to the KL divergence $\mathsf{KL}(P_1||P_2) = \sum_{x \in \mathcal{X}} P_1(x) \log \frac{P_1(x)}{P_2(x)}$.[2] Minimizing the KL divergence is generally seen as a desirable goal for learning a probabilistic model because $\mathsf{KL}(P_1||P_2) = 0$ if and only if P_1 and P_2 are the same almost surely. Given dataset $\mathcal{D}_n = \{X_i\}_{i=1}^n$ with $X_i \sim P^*$, we define the empirical measure $P_n(\cdot) = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}(\cdot)$. The MLE within a probability model space $\mathcal{M}$ is

$$\hat{P} \leftarrow \operatorname*{argmin}_{P \in \mathcal{M}} \mathsf{KL}(P_n||P) \equiv \operatorname*{argmax}_{P \in \mathcal{M}} \frac{1}{n} \sum_{X_i \in \mathcal{D}_n} \log P(X_i). \tag{1}$$

In the context of model-based RL, learning $\hat{\mathcal{P}}$ that minimizes a negative-log loss or other "probabilistic" losses leads to an estimate that tries to model all aspects of the environment. This might be beyond the requirement of solving the RL problem effectively. It might be the case that some aspects of the environment are irrelevant to find a good or optimal policy. For example, consider a visually-enabled robot that is supposed to learn how to navigate in a building. If we consider the camera image as a part of the state of the robot, trying to learn a transition probability kernel means that we have to learn how the camera image changes when the robot takes certain actions. This is a very

Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. Copyright 2017 by the author(s).

[1]An extended abstract version of this paper has been presented at European Workshop on Reinforcement Learning [Farahmand et al., 2016].

[2]We use P, $\hat{P}$, etc. to denote an unconditional probability distribution over $\mathcal{X}$, and we use $\mathcal{P}$, $\hat{\mathcal{P}}$, etc. to denote a conditional transition probability kernel.

high-dimensional state space and trying to learn such a conditional distribution with high enough accuracy, in the log-loss sense, is quite difficult. Nonetheless, modeling the probability distribution at that level of accuracy is not required to learn a policy that can navigate the robot in the building just fine. The only aspect of the model that is really required is a crude model that describes the building's topology as well as distances between rooms, and maybe the location of the objects. The robot does not really need to know the detail of paintings on the walls, the texture of objects, and many other visual detail of the building. On the other hand, if the goal is to have an interior decorator robot that suggests how to redecorate the building to make it visually appealing, all those visual information is required.

The difference between the navigator robot and the decorator one is not in the transition model of their environment, but is in the decision problem that they have to solve. The difference in the decision problem is reflected in the difference in the reward functions and as a result in the value functions. It is desirable to have a model learning formalism that takes the decision problem, or at least some aspects of it, into account.

Furthermore, the implicit assumption that model approximation error can be made zero, that is, $\mathcal{P}^*$ belongs to $\mathcal{M}$ used for estimation, may not be correct for many estimators. When we have the model approximation error, the model learning method must make a compromise in the choice of the estimate: None of the models in $\mathcal{M}$ would be the same as $\mathcal{P}^*$ (e.g., in the almost sure sense), so the estimation method has to choose a model with a minimum error with respect to (w.r.t.) some loss function. The choice of the loss function becomes important then. A loss function that is designed for a particular decision problem in hand provides a better approximation, for the task of solving the very same decision problem, than a probabilistic one that does not take the decision problem into account.

These arguments suggest that generic distribution estimation approaches such as MLE, which minimizes the KL-divergence w.r.t. the empirical distribution, might not be the best candidate for learning a model to be used within a model-based RL framework. Can we design a better "decision-aware" loss function that takes the decision problem into account?

This paper is a step towards incorporating some aspects of the underlying decision problem into model learning. We go beyond the "vanilla" model learning, and define a new loss function that incorporate the structure of the value function to learn the transition model (Section 2). We call the approach based on this loss function *Value-Aware Model Learning (VAML)*.

Algorithm 1 Generic Model-based Reinforcement Learning Algorithm

```
// MDP $(\mathcal{X}, \mathcal{A}, \mathcal{R}^*, \mathcal{P}^*, \gamma)$
// $K$: Number of interaction episodes
// $\mathcal{M}$: Space of transition probability kernels
// $\mathcal{G}$: Space of reward functions
Initialize a policy $\pi_0$
for $k = 0$ to $K - 1$ do
    Generate training set $\mathcal{D}_n^{(k)} = \{(X_i, A_i, R_i, X_i')\}_{i=1}^n$ by interacting with the true environment (potentially using $\pi_k$), i.e., $(X_i, A_i) \sim \nu_k$ with $X_i' \sim \mathcal{P}^*(\cdot|X_i, A_i)$ and $R_i \sim \mathcal{R}^*(\cdot|X_i, A_i)$.
    $\hat{\mathcal{P}} \leftarrow \mathrm{argmin}_{\mathcal{P} \in \mathcal{M}} \mathrm{Loss}_{\mathcal{P}}(\mathcal{P}; \cup_{i=0}^k \mathcal{D}_n^{(i)})$ {e.g., by the gradient descent specified in Theorem 1 for VAML or (12) for MLE}
    $\hat{r} \leftarrow \mathrm{argmin}_{r \in \mathcal{G}} \mathrm{Loss}_{\mathcal{R}}(r; \cup_{i=0}^k \mathcal{D}_n^{(i)})$
    $\pi_{k+1} \leftarrow \mathsf{Planner}(\hat{\mathcal{P}}, \hat{\mathcal{R}})$ {e.g., Fitted Q-Iteration}
end for
return $\pi_K$
```

We also provide a finite-sample upper bound guarantee for VAML in Section 3 showing the effect of the model approximation error, number of training samples, and the complexity of the model space. This guarantees the soundness of the algorithm. We empirically study the model learned by VAML/MLE within a complete model-based RL framework in a simple finite MDP problem. In the supplementary material, which is the extended version of this paper, we provide the proofs of all results. Moreover, we analyze the model approximation properties of VAML vs. MLE through a series of simple, but illuminating, examples. We also provide additional empirical results studying various aspects of VAML vs. MLE. The general conclusion of these results is that VAML is superior to MLE whenever we have a model approximation error, i.e., the true transition model does not belong to the class of models in which our estimator is selected.

2 VALUE-AWARE MODEL LEARNING

Algorithm 1 describes a generic model-based RL agent. It interacts with the environment, which is specified by an unknown Markov Decision Process (MDP) $(\mathcal{X}, \mathcal{A}, \mathcal{R}^*, \mathcal{P}^*, \gamma)$, to collect data $\mathcal{D}_n$. Here $\mathcal{X}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{R}^*$ is the reward distribution, and $\mathcal{P}^*$ is the transition probability kernel, and $0 \le \gamma < 1$ is the discount factor [Szepesvári, 2010]. The data is used to learn an estimate $\hat{\mathcal{P}}$ of the true transition probability $\mathcal{P}^*$ of the environment and an estimate $\hat{r}$ of the expected reward. The model learning step is usually done using an MLE, e.g., counting the number of transitions from state-action pair (x, a) to another state x' in a finite state-action MDP is such an estimate. The learned model is then used by a planning algorithm Planner to find a policy, with the goal of finding a close to optimal policy. The new policy might be used to generate more data and improve the model.

There are many variations to each step of this generic algorithm such as how to collect new data points (cf. [Hester and Stone, 2013]) or what Planner should we use from all possible value-based, policy search, etc. algorithms. Moreover, the interaction with the environment might be in a one-shot batch setting ($K = 1$) or in a continual online setting, and the spectrum in between.

The main thesis of this work is that estimating the model should be influenced by the way Planner is going to use it. So we focus on estimating the transition probability and we study how we should define a loss function $\mathrm{Loss}_{\mathcal{P}}$. We ignore all other important issues regarding designing a model-based RL algorithm for the moment in the rest of this work, except in the section on empirical studies (Section 4) where we choose a particular algorithm as Planner.

Let Planner be an algorithm that receives a model $\hat{\mathcal{P}}$ and returns a policy $\pi \leftarrow \mathsf{Planner}(\hat{\mathcal{P}})$. We assume that the reward function is already known to Planner, so we do not explicitly pass it as an argument. For a user-defined initial probability distribution $\rho \in \bar{\mathcal{M}}(\mathcal{X})$, with $\bar{\mathcal{M}}(\mathcal{X})$ being the space of probability distributions on $\mathcal{X}$, we evaluate the performance of π by

$$J(\pi) = \int \mathrm{d}\rho(x) V^{\pi}(x). \tag{2}$$

The goal of a successful model learner can then be defined as follows: Given a dataset $\mathcal{D}_n = \{(X_i, A_i, X_i')\}_{i=1}^n$ with $Z_i = (X_i, A_i) \sim \nu(\mathcal{X} \times \mathcal{A}) \in \bar{\mathcal{M}}(\mathcal{X} \times \mathcal{A})$, potentially different from ρ, and $X_i' \sim \mathcal{P}^*(\cdot|X_i, A_i)$, find $\hat{\mathcal{P}}$ such that $J(\pi)$ for $\pi \leftarrow \mathsf{Planner}(\hat{\mathcal{P}})$ is as large as possible. This is a very generic goal. To make it more concrete, we have to make a few choices. First suppose that Planner uses the Bellman optimality operator defined based on $\hat{\mathcal{P}}$ to find a $\hat{Q}^*$, that is $\hat{T}^* : Q \mapsto r + \gamma \hat{\mathcal{P}} \max_a Q$, and then outputs $\pi = \hat{\pi}(\cdot; \hat{Q}^*)$, the greedy policy w.r.t. $\hat{Q}^*$ defined as $\hat{\pi}(x; Q) = \operatorname{argmax}_{a \in \mathcal{A}} Q(x, a)$. The use of the Bellman [optimality] operator is central to value-based approaches such as the class of (Approximate) Value Iteration or (Approximate) Policy Iteration algorithms.

This is still too general, so we focus on the more specified goal of finding a $\hat{\mathcal{P}}$ such that the difference between T^*Q and $\hat{T}^*Q$ is not large. We may express this goal by defining the following cost (loss):

$$c(\hat{\mathcal{P}}, \mathcal{P}^*; V)(x, a) = \left| \left\langle \mathcal{P}^*(\cdot|x, a) - \hat{\mathcal{P}}(\cdot|x, a) \, , \, V \right\rangle \right| = \left| \int \left[\mathcal{P}^*(\mathrm{d}x'|x, a) - \hat{\mathcal{P}}(\mathrm{d}x'|x, a) \right] V(x') \right|, \tag{3}$$

in which we substituted $\max_a Q(\cdot, a)$ with V to simplify the presentation. In the rest of the paper, we may sometimes use $\mathcal{P}_z(\cdot)$ with $z = (x, a) \in \mathcal{Z} = \mathcal{X} \times \mathcal{A}$ to refer to the probability distribution $\mathcal{P}(\cdot|x, a)$, so $\mathcal{P}_z V = \int \mathcal{P}(\mathrm{d}y|x, a) V(\mathrm{d}y)$.

It might be argued that since

$$\left| \left\langle \mathcal{P}^*(\cdot|x, a) - \hat{\mathcal{P}}(\cdot|x, a) \, , \, V \right\rangle \right| \leq \left\| \mathcal{P}^*(\cdot|x, a) - \hat{\mathcal{P}}(\cdot|x, a) \right\|_1 \|V\|_\infty , \tag{4}$$

it is enough to learn $\hat{\mathcal{P}}$ such that the ℓ_1-norm of its difference with the true $\mathcal{P}^*$ is small. This can be achieved by minimizing the KL divergence because Pinsker's inequality shows that for two probability distributions P_1 and P_2, we have

$$\|P_1 - P_2\|_1 \leq \sqrt{2\mathsf{KL}(P_1 || P_2)}. \tag{5}$$

These two upper bounds together justify the use of MLE since MLE is the minimizer of the empirical approximation of the KL divergence, as shown in Section 1. This is the argument, sometimes implicit, behind most model-based RL algorithms that use a log-loss or a similar "probabilistic" loss to estimate the model.

Finding a minimizer for the KL divergence, Hellinger distance, ℓ_1 loss, or other losses that depend only on the probabilities, however, ignores the underlying decision problem, which is specified through the reward/value function. As an extreme example, suppose that $r(x) = c$ for all $x \in \mathcal{X}$, so V^π is constant for all policies, and the optimal policy would not have any preference over any of the actions. So even if $\mathcal{X}$ is a very large space (e.g., a subset of $\mathbb{R}^d$ with a large d), and however complex $\mathcal{P}^*$ is (e.g., the dynamics is not very regular), learning a $\hat{\mathcal{P}}$ sufficient to find the optimal policy is indeed very easy: Any transition probability distribution suffices to find the optimal policy. In contrast, $\|\hat{\mathcal{P}} - \mathcal{P}^*\|_1$ goes to zero at a convergence rate that depends on dimension d and regularities of $\mathcal{P}^*$, and can be very slow, e.g., $O(n^{-1/2d})$. An estimator for $\mathcal{P}^*$ that ignores this extra information requires more samples in order to provide a guarantee that the error in the model-based planning is small.[3]

Moreover, and maybe more importantly, if the true transition kernel $\mathcal{P}^*$ does not belong to the model space $\mathcal{M}$ from which we estimate the model $\hat{\mathcal{P}}$, we can only hope to find the "closest" model within $\mathcal{M}$ to $\mathcal{P}^*$. The notion of closeness, however, depends on the distance measure. A distance measure that explicitly takes into

[3]The relationship between the probabilistic loss and the LHS of (4) is a bit more subtle than what we portrayed here, but this should be enough for our current discussion. Refer to the supplementary material for a discussion on this issue.

account the decision problem and what really matters for Planner can be superior to the one that does not.

Returning to (3), there are three hurdles that should be addressed. The first is that $c(\hat{\mathcal{P}}, \mathcal{P}^*; V)(x,a)$ is defined as a pointwise measure of error, but we would like to learn a model that is valid for the whole state-action space $\mathcal{X} \times \mathcal{A}$. The second is that V itself is not known, so one cannot optimize this cost as is. The third is that $\mathcal{P}^*$, which is the main object of interest, is not known. Instead we have $\mathcal{D}_n = \{(X_i, A_i, X'_i)\}_{i=1}^n$ and as a result the empirical conditional distribution $\mathcal{P}_n(\cdot|x,a) = \frac{1}{n}\sum_{i=1}^n \delta_{X'_i|X_i,A_i}(\cdot|x,a)$. Here the conditional Dirac's delta function is defined as follows: For a measurable set S, $\delta_{X'_i|X_i,A_i}(S|x,a) = 1$ whenever $(x,a) = (X_i, A_i)$ and $X'_i \in S$, and 0 otherwise.

We can easily address the first concern by defining the cost functional as the expected squared pointwise cost w.r.t. a probability distribution $\nu \in \bar{\mathcal{M}}(\mathcal{X} \times \mathcal{A})$, i.e.,

$$c^2_{2,\nu}(\hat{\mathcal{P}}, \mathcal{P}^*; V) = \int \mathrm{d}\nu(x,a) \left| \int \left[\mathcal{P}^*(\mathrm{d}x'|x,a) - \hat{\mathcal{P}}(\mathrm{d}x'|x,a)\right] V(x') \right|^2 . \tag{6}$$

The choice of the $L_2(\nu)$-norm of the pointwise cost is motivated by the relation between the performance loss $J(\pi^*) - J(\pi) = \|V^* - V^\pi\|_{1,\rho}$ and the $L_2(\nu)$ of quantities such as the Bellman error $Q - T^\pi Q$ in API or the approximation error $T^* Q_k - Q_{k+1}$ in AVI [Farahmand et al., 2010]. Somehow looser relationship also exists between the performance loss and the $L_1(\nu)$ error [Munos, 2007], but working with the squared error is easier in our future derivations. One could use the supremum norm too, but it would be too conservative. The choice of ν determines where in the state-action space we have to emphasize the accuracy of the model, in the sense of how well it can approximate the effect of the Bellman operator evaluated at that point. When the choice of ν is clear from the context, we may simply write $c(\hat{\mathcal{P}}, \mathcal{P}^*; V)$.

To address the second concern, not knowing V, we may take a robust approach w.r.t. the choice of value function. We define the cost function to reflect that our goal is to find a $\hat{\mathcal{P}}$ that is suitable for all V in a given value function space $\mathcal{F}$. Therefore, we define

$$c^2_{2,\nu}(\hat{\mathcal{P}}, \mathcal{P}^*) = \int \mathrm{d}\nu(x,a) \sup_{V \in \mathcal{F}} \left| \int \left[\mathcal{P}^*(\mathrm{d}x'|x,a) - \hat{\mathcal{P}}(\mathrm{d}x'|x,a)\right] V(x') \right|^2 . \tag{7}$$

To understand this loss better, let us focus on a single state-action pair (x,a) and study the pointwise cost.[4] Note that even though

$$\sup_{V \in \mathcal{F}} \left| [\mathcal{P}^*(\cdot|x,a) - \hat{\mathcal{P}}(\cdot|x,a)] V(\cdot) \right| \leq \left\| \hat{\mathcal{P}}(\cdot|x,a) - \mathcal{P}^*(\cdot|x,a) \right\|_1 \sup_{V \in \mathcal{F}} \|V\|_\infty , \tag{8}$$

the LHS is often much smaller than the upper bound. They would only become equal when $\mathcal{F}$ is the space of bounded measurable functions, which is much larger than the usual function spaces that we often deal with, e.g., defined based on a finite set of basis or even a reproducing kernel Hilbert space (RKHS). As the goal is to minimize the LHS of (8), and because its RHS can be a loose upper bound for most choices of $\mathcal{F}$, directly optimizing the LHS can lead to better models compared to minimizing the ℓ_1 loss or the KL distance (minimized by MLE), which itself is yet another level of upper bounding according to (5).

The loss function (7) reflects the influence of the value function on the model-learning objective. If we happen to know that V has certain regularities, e.g., it belongs to the Sobolev space $\mathbb{W}^k(\mathbb{R}^d)$ or a reproducing kernel Hilbert space, this loss function lets us focus on learning a model that can discriminate between such value functions, and not more.

To address the last concern, one approach is to follow the usual recipe in machine learning and statistics, the Empirical Risk Minimization (ERM), by replacing the true state transition kernel $\mathcal{P}^*$ with the observed empirical distribution $\mathcal{P}_n$ and $\nu \in \bar{\mathcal{M}}(\mathcal{X} \times \mathcal{A})$ with the empirical measure $\nu_n(\cdot) = \frac{1}{n}\sum_{i=1}^n \delta_{(X_i,A_i)}(\cdot)$. Theorem 2 shows that under certain standard conditions, this is indeed a sound procedure. The result would be the following cost functional:

$$\begin{aligned} c^2_{2,n}(\hat{\mathcal{P}}) &= c^2_{2,\nu_n}(\hat{\mathcal{P}}, \mathcal{P}_n) = \\ &\frac{1}{n} \sum_{(X_i,A_i) \in \mathcal{D}_n} \sup_{V \in \mathcal{F}} \left| \int \left[\mathcal{P}_n(\mathrm{d}x'|X_i,A_i) - \hat{\mathcal{P}}(\mathrm{d}x'|X_i,A_i)\right] V(x') \right|^2 \\ &= \frac{1}{n} \sum_{(X_i,A_i) \in \mathcal{D}_n} \sup_{V \in \mathcal{F}} \left| V(X'_i) - \int \hat{\mathcal{P}}(\mathrm{d}x'|X_i,A_i) V(x') \right|^2 . \end{aligned} \tag{9}$$

The output of VAML is

$$\hat{\mathcal{P}} \leftarrow \operatorname*{argmin}_{\mathcal{P} \in \mathcal{M}} c^2_{2,n}(\hat{\mathcal{P}}). \tag{10}$$

To completely specify the algorithm, we have to choose $\mathcal{F}$ and $\mathcal{M}$. We do this in the rest of this section.

[4] One might argue that it would be better to have the supremum over V *outside* the integral over state-actions. We study this a bit further in the supplementary material, but we do not pursue this path much more as it does not seem to be as computationally appealing as the current formulation.

2.1 The Gradient of $c_{2,n}^2(\hat{\mathcal{P}})$

In this section, we compute the gradient of the cost function $c_{2,n}^2(\hat{\mathcal{P}})$ for a particular choice of model space $\mathcal{M}$ and the value function space $\mathcal{F}$. We choose $\hat{\mathcal{P}} = \hat{\mathcal{P}}_w$ as an exponential family described by features $\phi' : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \mathbb{R}^{p'}$ and the weight vector $w \in \mathbb{R}^{p'}$, i.e.,

$$\hat{\mathcal{P}}_w(\mathrm{d}x'|x,a) = \frac{\exp\left(\phi'^\top(x'|x,a)w\right)}{\int \exp\left(\phi'^\top(x''|x,a)w\right)\mathrm{d}x''}\mathrm{d}x'. \quad (11)$$

We consider the case that the value function belongs to the function space $\mathcal{F} = \mathcal{F}_B = \left\{ V_\theta(x) = \phi^\top(x)\theta : \theta \in \mathbb{R}^p, \|\theta\|_2 \le B \right\}$ with $\phi : \mathcal{X} \to \mathbb{R}^p$ being the feature map. The use of linear value function approximator is a common practice in the RL literature.

Note that in general features $\phi'(x'|x,a)$ of the probability model $\hat{\mathcal{P}}$ are different from the features $\phi(x)$ used to represent the value function. Of course, we may choose them to be related, for example by defining $\phi'(x|x,a) = h(x,a)\phi(x')$ for some function $h(x,a)$.

We use the following key relationship in our derivations, which are presented in the supplementary material.

$$\sup_{V_\theta \in \mathcal{F}_B} \left\langle \mathcal{P}_z^* - \hat{\mathcal{P}}_z \, , \, V \right\rangle_{\mathcal{X}} = B \left\| \int (\mathcal{P}^* - \hat{\mathcal{P}})(\mathrm{d}y|z)\phi^\top(y) \right\|_2 .$$

Denote $\mathbf{Cov}_{X' \sim \hat{\mathcal{P}}_w(\cdot|X_i,A_i)}\left(\phi(X'), \phi'(X'|X_i,A_i)\right)$ as the cross-covariance between $\phi(X')$ and $\phi'(X'|X_i,A_i)$ w.r.t. the probability distribution $\hat{\mathcal{P}}_w(\cdot|X_i,A_i)$.

Theorem 1. *Consider the parameterization (11) of the estimated probability transition kernel $\hat{\mathcal{P}}_w$, and the value function space $\mathcal{F} = \mathcal{F}_B$. The gradient of $c_{2,n}^2(\hat{\mathcal{P}}_w)$ w.r.t. the parameter w is*

$$\nabla_w c_{2,n}^2(\hat{\mathcal{P}}_w) = \frac{2B^2}{n} \sum_{i=1}^n \left[\mathbb{E}_{X' \sim \hat{\mathcal{P}}_w(\cdot|X_i,A_i)}\left[\phi(X')\right] - \phi(X_i') \right]^\top \cdot \mathbf{Cov}_{X' \sim \hat{\mathcal{P}}_w(\cdot|X_i,A_i)}\left(\phi(X'), \phi'(X'|X_i,A_i)\right).$$

The loss function has two main terms. The first term computes the difference between the empirical average of the *value* function features $\phi(X_i')$ and its expectation $\mathbb{E}_{X' \sim \hat{\mathcal{P}}_w(\cdot|X_i,A_i)}\left[\phi(X')\right]$ according to the model with parameter w. So this term encourages finding a model that "matches" according to the features ϕ of the value function space $\mathcal{F}$. The other term is the cross-covariance between the features ϕ of the value function and the features ϕ' of the model. This term might be seen as a weighting term for the first one.

It is instructive to compare this gradient with the gradient of the negative log-loss (1) with the same exponential model, which is

$$\frac{1}{n}\sum_{i=1}^n \left[\mathbb{E}_{X' \sim \hat{\mathcal{P}}_w(\cdot|X_i,A_i)}\left[\phi'^\top(X'|X_i,A_i)\right] - \phi'^\top(X_i'|X_i,A_i) \right]. \quad (12)$$

One can interpret this by saying that MLE is trying to find $\hat{\mathcal{P}}_w$ such that the expected value of *model* features ϕ' evaluated at the next-state matches the empirical values. The matching is based on model features ϕ' and not the value features ϕ. One might see that for finite MDPs with exact representation of both value function and the model (i.e., lookup tables for both ϕ and ϕ'), the asymptotic solutions of VAML and MLE are the same, but since their gradients on the way are not, they approach that point differently.

Working with exponential family can be computationally expensive, no matter whether we use MLE or VAML. To begin with, even sampling from (11) requires the computation of the normalizing factor (i.e., partition function), which except in special cases such as for Gaussian distributions, does not have a closed-form solution. The second issue is that to compute the gradients required for MLE or VAML, we require to estimate certain expectations (and covariance matrices). This can be challenging too. On the positive side, however, these computations, have been the subject of many years of research; and there are already many methods, exact or approximate, to evaluate this general family of probability distributions, e.g., various Monte Carlo estimates or variational methods [MacKay, 2003; Goodfellow et al., 2016]. Moreover, we may not really need to have a very accurate estimate of the gradients in VAML or MLE in order to minimize the cost function. It might be enough to only have a few samples from the learned distribution to estimate the "direction" of the gradient correctly. This is one of the ideas behind Contrastive Divergence [Carreira-Perpinan and Hinton, 2005], which we can use for VAML too.

3 STATISTICAL ANALYSIS OF VAML

We provide a finite sample error upper bound showing that VAML is indeed a sound algorithm in the sense that the minimizer $\hat{\mathcal{P}}$ of the empirical loss $c_{2,n}^2$, if attained, has a small error (7), given enough data points n and under standard capacity condition on the function spaces $\mathcal{M}$. The result of this section is not limited to exponential models of $\mathcal{M}$.

Consider a family of probability distributions $\mathcal{M}_0$ and a pseudo-norm $J : \mathcal{M}_0 \to [0,\infty)$. Let the set $\mathcal{M}$ used by VAML be a subset $M = M_B = \{ \mathcal{P} \in \mathcal{M}_0 : J(\mathcal{P}) \le B \}$ for some $B > 0$. We can think of J of a measure of complexity of functions in $\mathcal{M}_0$, so

$\mathcal{M}$ would be a ball with a fixed radius B w.r.t. J. If $\mathcal{M}_0$ is defined based on an RKHS, we can think of J as the inner product norm of the RKHS. We have the following assumptions on the metric entropy (logarithm of the covering number) of $\mathcal{M}$.

Assumption A1 (Capacity of Function Space) For $B > 0$, let $M = M_B = \{ \mathcal{P} \in \mathcal{M}_0 : J(\mathcal{P}) \leq B \}$. There exist constants $C > 0$ and $0 < \alpha < 1$ such that for any $u, B > 0$ and all sequence $z_1, \dots, z_n \in \mathcal{Z}$, the following metric entropy condition is satisfied:

$$\log \mathcal{N}\left(u, \mathcal{M}_B, L_2(P_{z_{1:n}})\right) \leq C \left(\frac{B}{u}\right)^{2\alpha}.$$

Metric entropy of $\mathcal{M}_B$ is a measure of the size of $\mathcal{M}_B$, and roughly speaking, it is the logarithm of the minimum number of balls with radius u that are required to completely cover $\mathcal{M}_B$. In general, it is more difficult to estimate a function when the metric entropy grows fast when u decreases. Here $L_2(P_{z_{1:n}})$ is the L_2-norm defined w.r.t. the empirical measure (cf. e.g., Section 9.3 of [Györfi et al., 2002];). For many examples of the metric entropy results, refer to [van de Geer, 2000; Györfi et al., 2002; Giné and Nickl, 2015]. After stating this assumption, we are ready to state the theorem.

Theorem 2. *Given a dataset $\mathcal{D}_n = \{(X_i, A_i, X'_i)\}_{i=1}^n$ with independent and identically distributed samples $(X_i, A_i) \sim \nu$, with $X'_i \sim \mathcal{P}^*(\cdot|X_i, A_i)$, let $\hat{\mathcal{P}}$ be the minimizer of the VAML algorithm, i.e., $\hat{\mathcal{P}} \leftarrow$ argmin$_{\mathcal{P} \in \mathcal{M}} c^2_{2,n}(\hat{\mathcal{P}})$, with the previously specified choice of value function space $\mathcal{F}$. Let Assumption A1 hold. Furthermore, assume that $\sup_{x \in \mathcal{X}} \|\phi(x)\|_\infty \leq 1$ and $\sup_{x \in \mathcal{X}} \|\phi(x)\|_2 \leq 1$. Fix $\delta > 0$. There exists a constant $c > 0$ such that*

$$\mathbb{E}\left[\sup_{V \in \mathcal{F}} \left|(\hat{\mathcal{P}}_Z - \mathcal{P}^*_Z)V\right|^2\right] \leq \inf_{\mathcal{P} \in \mathcal{M}} \mathbb{E}\left[\sup_{V \in \mathcal{F}} \left|(\mathcal{P}_Z - \mathcal{P}^*_Z)V\right|^2\right] + c(1 + B^\alpha)p\sqrt{\frac{\log(p/\delta)}{n}} + \frac{16\log(4/\delta)}{3n},$$

with probability at least $1 - \delta$.

This upper bound shows the usual model (or function) approximation error (first term) and the estimation error (second and third terms). The dominant term in the estimation error behaves $O(n^{-1/2})$, which is the usual behaviour of the supremum of the empirical process for models that are not very large. The size of the function space $\mathcal{M}$, specified by B in Assumption A1, appears in the bound. We also see the effect of size of ϕ vector, specifying the value function space $\mathcal{F}$, appears linearly. We believe that this dependence on p is suboptimal, and can be improved further.

Maybe more interesting is the effect of the model approximation error. The bound shows that the error of the estimated $\hat{\mathcal{P}}$ is comparable to the error of the best choice in the model class $\mathcal{M}$, i.e., $\inf_{\mathcal{P} \in \mathcal{M}} \mathbb{E}\left[\sup_{V \in \mathcal{F}} \left|(\mathcal{P}_Z - \mathcal{P}^*_Z)V\right|^2\right]$. This is reassuring since VAML was motivated by the fact that the important property of an estimated model $\hat{\mathcal{P}}$ should be that $\left|\left\langle \hat{\mathcal{P}}(\cdot|z) - \mathcal{P}^*(\cdot|z)\, , \, V \right\rangle\right|$ is small only for $V \in \mathcal{F}$ that might be encountered by the algorithm, and not necessarily for all possible value functions, which cannot even be represented by the value-based algorithm.

One could obtain faster estimation error (i.e., $O(n^{-1})$) by studying the modulus of the continuity of the empirical process instead of the supremum of the empirical process, as we do here. We decided not to provide such a result because of two reasons. The first is that faster rates require increasing the constant in front of the approximation error (it would not be 1 anymore). In the regime that we have an approximation error ($\mathcal{P}^* \notin \mathcal{M}$), which is the regime that can make VAML superior to MLE, this would lead to asymptotically worse results. The other reason is to simplify the proofs and making them more accessible. The proofs can be found in the same section of the supplementary material.

A few other short remarks are in order. The first is that this is a statistical guarantee, and is valid under the condition that $\min_{\mathcal{P} \in \mathcal{M}} c^2_{2,n}(\hat{\mathcal{P}})$ is indeed attained. We have not shown that this minimum can be achieved by following the gradient of Theorem 1, especially since the VAML's objective is not necessarily convex. Another remark is that we do not analyze the effect of the model estimation error on the quality of the policy obtained by Planner($\hat{\mathcal{P}}$). Ávila Pires and Szepesvári [2016] provide such a policy error bound.

4 EMPIRICAL STUDIES

In this section, we empirically study the performance of VAML and MLE-based estimators when they are used within a complete model-based RL algorithm. We provide several others experiments designed to better understand various aspects of VAML and MLE better in the supplementary material.

We consider a finite MDP. We choose several partition-based (i.e., aggregation) model space $\mathcal{M}$ to which the true model $\mathcal{P}^*$ does not belong. Also we consider a partition-based value function space $\mathcal{F}^{|\mathcal{A}|}$. When the resolution of the partitioning is lower than the number of states, the optimal value function Q^* might be outside $\mathcal{F}^{|\mathcal{A}|}$. We use the approximate value iteration as Planner. That is, given a model $\hat{\mathcal{P}}$, which is chosen to be either the true model $\mathcal{P}^*$ of the MDP or the estimated models $\mathcal{P}_{\text{VAML}}$ or $\mathcal{P}_{\text{MLE}}$, we repeatedly apply

$$\hat{Q}_{k+1} \leftarrow \Pi_{\mathcal{F}^{|\mathcal{A}|}}(\hat{T}^*_{\hat{\mathcal{P}}}\hat{Q}_k)$$

to obtain an approximation $\hat{Q}^*_{\mathcal{P}^*/\mathcal{P}_{\text{VAML}}/\mathcal{P}_{\text{MLE}}}$ to Q^*, the true optimal action-value function. Here $\Pi_{\mathcal{F}^{|\mathcal{A}|}}$ is the orthogonal projection onto $\mathcal{F}^{|\mathcal{A}|}$, hence the "approximate" part of AVI. We obtain Q^* (and hence V^*) using exact VI with $\mathcal{P}^*$. Note that $\hat{Q}^* = \hat{Q}^*_{\mathcal{P}^*}$ is only an approximation to Q^* because $\hat{Q}^*$ is obtained by AVI, so it is forced to be within $\mathcal{F}^{|\mathcal{A}|}$, but Q^* in general may not belong to $\mathcal{F}^{|\mathcal{A}|}$. The approximations $\hat{V}^*_{\mathcal{P}^*/\mathcal{P}_{\text{VAML}}/\mathcal{P}_{\text{MLE}}}$ are defined similarly.

After obtaining the various approximations of the optimal value function, we compute their corresponding greedy policies $\pi_{\text{MLE/VAML}} \leftarrow \mathsf{Planner}(\mathcal{P}_{\text{MLE}}/\mathcal{P}_{\text{VAML}})$. In particular, we are interested in $V^{\pi_{\text{MLE}}}$ and $V^{\pi_{\text{VAML}}}$, the true value function of the policies obtained by the estimated models.

We use two criteria to evaluate the quality of the estimated models. The first is that how close $\hat{V}^*_{\mathcal{P}_{\text{VAML}}/\mathcal{P}_{\text{MLE}}}$ is to $\hat{V}^*_{\mathcal{P}^*}$. We use the L_2-norm of these distances, i.e., $\|\hat{V}^*_{\mathcal{P}^*} - \hat{V}^*_{\mathcal{P}_{\text{VAML}}/\mathcal{P}_{\text{MLE}}}\|_2$. By comparing to $\hat{V}^*_{\mathcal{P}^*}$, instead of V^*, we separate the error caused by the choice of model space and the model estimation procedure (MLE or VAML), which is our main object of study, from the error caused by the choice of value function space $\mathcal{F}^{|\mathcal{A}|}$. The second criterion is the performance loss of the obtained policies compared to the optimal policy, that is, $\sum_{x \in \mathcal{X}} [V^*(x) - V^{\pi_{\text{MLE/VAML}}}(x)] = \|V^* - V^{\pi_{\text{MLE/VAML}}}\|_1$, where $\pi_{\mathcal{P}_{\text{VAML}}} = \hat{\pi}(\cdot; \hat{Q}^*_{\mathcal{P}_{\text{VAML}}})$ and $\pi_{\mathcal{P}_{\text{MLE}}} = \hat{\pi}(\cdot; \hat{Q}^*_{\mathcal{P}_{\text{MLE}}})$, the greedy policies w.r.t. $\hat{Q}^*_{\mathcal{P}_{\text{VAML}}}$ and $\hat{Q}^*_{\mathcal{P}_{\text{MLE}}}$. Because of the value function and model approximation errors, the performance loss might be non-zero. But it is possible that even though $\hat{Q}_{\mathcal{P}_{\text{VAML}}/\mathcal{P}_{\text{MLE}}} \neq Q^*$, the greedy policy is still an optimal policy, as we shortly see. This is due to the action-gap phenomenon [Farahmand, 2011].

Let us define the parameters of the problem more concretely. We choose a finite state random-walk MDP with $|\mathcal{X}| = 25$, $\mathcal{A} = \{\text{left}, \text{right}\}$, and $\gamma = 0.9$. The choice of $a =$ "right" moves the agent to one of the four right-side states with equal probability (with a total probability of 0.7), does not move it (with probability of 0.2), and moves it to the left-side state (with probability of 0.1). The opposite holds for $a =$ "left". The boundaries are not connected, and the behaviour changes accordingly. The value function space $\mathcal{F}$ is defined based on partitioning of the state space $\mathcal{X}$ to M subsets. So we have $\mathcal{F} = \left\{ x \mapsto \sum_{j=1}^{M} v_j \mathbb{I}\{x \in J_j\} \,:\, v \in \mathbb{R}^M \right\}$. The action-value function space $\mathcal{F}^{|\mathcal{A}|}$ is simply $|\mathcal{A}| = 2$ copies of $\mathcal{F}$. We change M in our experiments.

The model space, used by both VAML and MLE, is an exponential family defined based on N partitions, cf. (11). The features ϕ' are defined so that each of them is an indicator function of whether given a state-action pair (x, a), the next-state x' would be in one of the N partitions or not. More precisely,

$$\mathcal{M} = \Big\{ \hat{\mathcal{P}}_w(x'|x,a) : \phi'_{i,k,l}(x'|x,a) = \mathbb{I}\{x' \in I_i, x = k, a = l\}, \\ i = 1, \dots, N, k = 1, \dots, |\mathcal{X}|, l = 1, \dots, |\mathcal{A}|, \\ w \in \mathbb{R}^{N|\mathcal{X}||\mathcal{A}|} \Big\}.$$

Note that the partitioning is only on the next-state x', and not the current state x. We change N in our experiments. A small detail is that because the state space is finite and partitions $\{I_i\}$ and $\{J_j\}$ have integer-valued lengths, the lengths of all of them are not exactly $|\mathcal{X}|/M$ (or $|\mathcal{X}|/N$). A state x belongs to J_j with $j = \lfloor \frac{x}{M} \rfloor$, and similarly for I_is.

We use gradient descent to optimize the loss functions for both VAML and MLE. In particular, we use ADAM by Kingma and Ba [2015] with the choice of $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\alpha = 0.25$, and $\varepsilon = 10^{-8}$ in their notations. We followed ADAM for 1000 iterations.

Figure 1 present the results of the experiments for $M = 11$ (more results in the supplementary material). The bold curves in each figure show the results when $\mathcal{P}^*$ is given as the input to VAML or MLE (so $c^2_{2,\nu}(\hat{\mathcal{P}}, \mathcal{P}^*)$ of (7) is minimized instead of $c^2_{2,\nu_n}(\hat{\mathcal{P}}, \mathcal{P}_n)$ of (9); and similarly for MLE), while the dashed curves are for when samples (i.e., empirical measure $\mathcal{P}_n$) are used for minimization. For the empirical measure, for each choice of a, we draw 100 states X_i uniformly from $\mathcal{X}$, and then draw 25 samples from the next state according to $\mathcal{P}(\cdot|X_i, a)$. So in total, we have $2 \times 100 \times 25 = 5000$ samples. Studying the behaviour of the algorithms under both the true distribution and the empirical distribution allows us to separate the errors due to model approximation error and the estimation error.

The left-side figures show the value function estimation error $\|\hat{V}^*_{\mathcal{P}^*} - \hat{V}^*_{\mathcal{P}_{\text{VAML}}/\mathcal{P}_{\text{MLE}}}\|_2$. We observe that as the number of partitions N for the model space increases, the error decreases too. The errors for the VAML model for most Ns are smaller than MLE's, sometimes significantly.

A curious observation is that when N is an integer multiply of M, the number of partitions in the representation of $\mathcal{F}$, the error of MLE becomes very small—sometimes even slightly smaller than VAML's (when $N = 11, 22$ in Figure 1). This is aligned with our theoretical analysis in Section 5 of the supplementary material. We observe a similar smallness of errors in the neighbourhood of those integer multiplies because the structure of the partition would be similar to the aforementioned case. When empirical data is used, the difference between VAML and MLE become less significant since the estimation error dominates the model approximation error. We also note that $\|\hat{V}^*_{\mathcal{P}^*} - V^*\|_2 = 1.40$ for $M = 11$.

The right-side figures show the performance loss $\|V^* -$

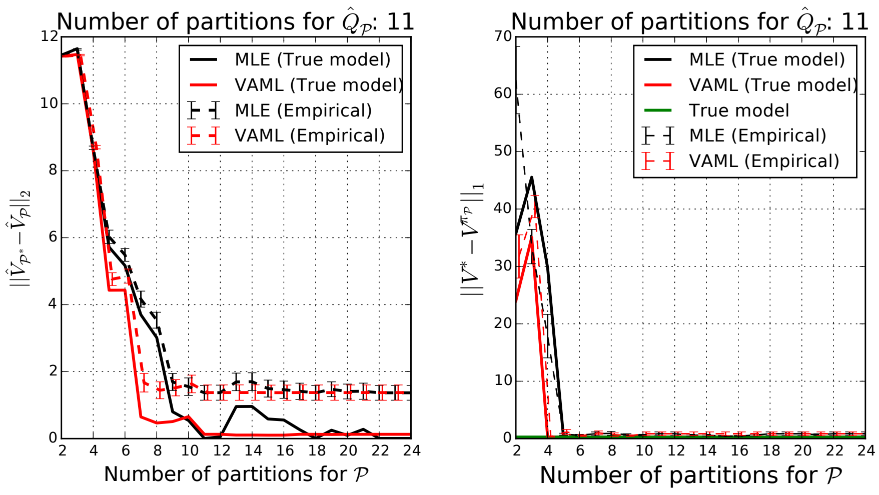

Figure 1: The effect of the number of partitions N of $\hat{\mathcal{P}}$ on MLE and VAML when $\mathcal{F}^{|\mathcal{A}|}$ has $M = 11$ partitions. The left figure shows the value function approximation error. The right figure shows the performance loss of using the obtained greedy policies. The dashed curves correspond to the empirical model. The error bars depict one standard error with the number of independent runs equal 20.

$V^{\pi_{\text{MLE/VAML}}}\|_1$ for both cases. If the performance loss of a policy is zero, it means that it is optimal. As a baseline, the green line shows the performance loss of the greedy policy w.r.t. $\hat{Q}^*_{\mathcal{P}^*}$. For $M = 11$, the performance loss is 0.277. It is quite possible that even though there is a significant value function error, due the action-gap phenomenon, the greedy policy is behaving close to optimal or the best value function in the class. That is why we observe that in case of $M = 11$, the performance of $\pi_{\mathcal{P}_{\text{VAML}}}$ and $\pi_{\mathcal{P}_{\text{MLE}}}$ is as good as good as $\hat{\pi}(\cdot; \hat{Q}^*_{\mathcal{P}^*})$ after $N = 4$ for VAML and $N = 5$ for MLE. It is curious to note that the performance loss of MLE is sometimes slightly better than that of $\hat{\pi}(\cdot; \hat{Q}^*_{\mathcal{P}^*})$, particularly at those values of N when $\|\hat{V}^*_{\mathcal{P}^*} - \hat{V}^*_{\mathcal{P}_{\text{MLE}}}\|_2$ is larger. This basically means that wronger models happened to make better policies. We do not believe this is a general pattern beyond this particular problem, but further investigation might be interesting.

5 DISCUSSION AND FUTURE WORK

We presented a loss function to learn the probability transition model to be used by a model-based reinforcement learning algorithm. In contrast with the conventional approaches, we take some aspects of the decision problem, particularly the knowledge about the value function approximator, into account.

There are several methods for learning the transition probability kernel or quantities related to it. Methods such as [Ormoneit and Sen, 2002; Barreto et al., 2011] implicitly learn the transition model, but they do not benefit from the structure of the value function. Some other methods estimate auxiliary operators that are semantically different from $\mathcal{P}$ (e.g., they are not probability kernels), but can be used to compute the effect of Bellman operator on a value function [Grünewälder et al., 2012; Yao et al., 2014; Lever et al., 2016]. Comparing VAML, which learns a generative model, and these other approaches is an interesting question.

We empirically studied the behaviour of VAML and MLE when they are used within a model-based RL algorithm. The results showed that minimizing the loss suggested by VAML translates into having a better value function approximation error, as well as smaller performance loss when the learned model is used for planning. We also studied model approximation properties of VAML vs. MLE through some examples in the supplementary material.

We would like to mention that exponential family is not the only class of probability distributions for modeling of the environment. Another possibility, which deserves further study, is the adoption of the generative adversarial network to VAML's loss function [Goodfellow et al., 2014]. Finally, incorporating the structure of policy space into model learning is another interesting research topic along the research program of this work.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful feedback.

References

Bernardo Ávila Pires and Csaba Szepesvári. Policy error bounds for model-based reinforcement learning with factored linear models. In *Conference on Learning Theory (COLT)*, 2016. 6

André M.S. Barreto, Doina Precup, and Joelle Pineau. Reinforcement learning using kernel-based stochastic factorization. In J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS - 24)*, pages 720–728. 2011. 8

Miguel A Carreira-Perpinan and Geoffrey Hinton. On contrastive divergence learning. In *International Workshop on Artificial Intelligence and Statistics (AISTATS)*, volume 10, pages 33–40, 2005. 5

Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013. 1

Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussan. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2): 408–423, 2015. 1

Amir-massoud Farahmand. Action-gap phenomenon in reinforcement learning. In J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS - 24)*, pages 172–180. Curran Associates, Inc., 2011. 7

Amir-massoud Farahmand, Azad Shademan, Martin Jägersand, and Csaba Szepesvári. Model-based and model-free reinforcement learning for visual servoing. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 2917–2924, May 2009. 1

Amir-massoud Farahmand, Rémi Munos, and Csaba Szepesvári. Error propagation for approximate policy and value iteration. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS - 23)*, pages 568–576. 2010. 4

Amir-massoud Farahmand, André M.S. Barreto, and Daniel N. Nikovski. Value-aware loss function for model learning in reinforcement learning. In *13th European Workshop on Reinforcement Learning (EWRL)*, December 2016. 1

Evarist Giné and Richard Nickl. *Mathematical Foundations of Infinite-Dimensional Statistical Models*. Cambridge University Press, 2015. 6

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS - 27)*, pages 2672–2680. 2014. 8

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. 5

Steffen Grünewälder, Guy Lever, Luca Baldassarre, Arthur Gretton, and Massimiliano Pontil. Modelling transition dynamics in MDPs with RKHS embeddings. In *International Conference on Machine Learning (ICML)*, pages 535–542. ACM, 2012. 8

László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer Verlag, New York, 2002. 6

Todd Hester and Peter Stone. TEXPLORE: Real-time sample-efficient reinforcement learning for robots. *Machine Learning*, 90(3), 2013. 1, 3

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 7

Guy Lever, John Shawe-Taylor, Ronnie Stafford, and Csaba Szepesvári. Compressed conditional mean embeddings for model-based reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2016. 8

David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003. 5

Rémi Munos. Performance bounds in L_p norm for approximate value iteration. *SIAM Journal on Control and Optimization*, pages 541–561, 2007. 4

Dirk Ormoneit and Saunak Sen. Kernel-based reinforcement learning. *Machine Learning*, 49:161–178, 2002. 8

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998. 1

Richard S. Sutton, Csaba Szepesvári, Alborz Geramifard, and Michael Bowling. Dyna-style planning with linear function approximation and prioritized sweeping. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, 2008. 1

Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Morgan Claypool Publishers, 2010. 1, 2

Sara A. van de Geer. *Empirical Processes in M-Estimation*. Cambridge University Press, 2000. 6

Hengshuai Yao, Csaba Szepesvári, Bernardo Ávila Pires, and Xinhua Zhang. Pseudo-MDPs and factored linear action models. In *IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning (ADPRL)*, 2014. 8