Model-Based Reinforcement Learning with Value-Targeted Regression

Alex Ayoub [1] **Zeyu Jia** [2] **Csaba Szepesvári** [1 3] **Mengdi Wang** [4 3] **Lin F. Yang** [5]

Abstract

This paper studies model-based reinforcement learning (RL) for regret minimization. We focus on finite-horizon episodic RL where the transition model P belongs to a known family of models $\mathcal{P}$, a special case of which is when models in $\mathcal{P}$ take the form of linear mixtures: $P_\theta = \sum_{i=1}^d \theta_i P_i$. We propose a model based RL algorithm that is based on the optimism principle: In each episode, the set of models that are 'consistent' with the data collected is constructed. The criterion of consistency is based on the total squared error that the model incurs on the task of predicting *state values* as determined by the last value estimate along the transitions. The next value function is then chosen by solving the optimistic planning problem with the constructed set of models. We derive a bound on the regret, which, in the special case of linear mixtures, takes the form $\tilde{\mathcal{O}}(d\sqrt{H^3T})$, where H, T and d are the horizon, the total number of steps and the dimension of θ, respectively. In particular, this regret bound is independent of the total number of states or actions, and is close to a lower bound $\Omega(\sqrt{HdT})$. For a general model family $\mathcal{P}$, the regret bound is derived based on the Eluder dimension.

1. Introduction

In reinforcement learning (RL), a core problem in artificial intelligence (Russel & Norvig, 2003; Sutton & Barto, 2018), an agent learns to control a possibly complex, initially unknown environment in a sequential trial and error process. The application of RL algorithms to various domains, such as games, robotics and science, has witnessed phenomenal empirical advances during the last few years (e.g., Mnih et al. 2015; Silver et al. 2017; AlQuraishi 2019; Arulkumaran et al. 2019). In online RL, an agent has to learn to act in an unknown environment "from scratch", collect data as she acts, and adapt her policy to maximize the reward collected, or, equivalently, to minimize her regret. Designing RL algorithms that provably achieve sublinear regret in some class of environments has been the subject of much research, mainly focusing on the so-called tabular, and linear-factored MDP settings (e.g., Jaksch et al. 2010; Osband et al. 2014; Azar et al. 2017; Dann et al. 2017; 2018; Agrawal & Jia 2017; Osband et al. 2017; Jin et al. 2018; Yang & Wang 2019a; Jin et al. 2019). An appealing alternative to studying these structured cases is to consider learning and acting when the environment is described by a general model class, the topic of the current paper. Despite its appeal, as it appears, prior work, considered this option exclusively in a Bayesian setting. In particular, (Strens, 2000) introduced posterior sampling to RL, which was later analyzed by (Osband & Van Roy, 2014; Abbasi-Yadkori & Szepesvári, 2015; Theocharous et al., 2017) (for a more in-depth discussion of related work, the reader is referred to Section 5). As opposed to these works, in the present paper we are interested in developing algorithms for bounding the worst-case expected regret.

The specific setting that we adopt is that of episodic reinforcement learning in an environment where the unknown transition probability model that describes the environment's stochastic dynamics belongs to a family of models that is given to the learner. The model family $\mathcal{P}$ is a general set of models, and it may be either finitely parametrized or nonparametric. In particular, our approach accommodates working with smoothly parameterized models (e.g., Abbasi-Yadkori & Szepesvári, 2015), and can find use in both robotics (Kober et al., 2013) and queueing systems (Kovalenko, 1968). An illuminating special case is when elements of $\mathcal{P}$ take the form $P_\theta = \sum_i \theta_i P_i$ where $P_1, P_2, \ldots, P_d$ are fixed, known basis models and $\theta = (\theta_1, \ldots, \theta_d)$ are unknown, real-valued parameters. Model P_θ can be viewed as a linear mixture model that aggregates a finite family of known basic dynamical models (Modi et al., 2019). As an important special case, linear mixture models include the linear-factor MDP model of

[1]Amii and Department of Computing Science, University of Alberta [2]School of Mathematical Science, Peking University [3]DeepMind [4]Department of Electrical Engineering, Princeton University [5]Department of Electrical and Computer Engineering, University of California, Los Angeles. Correspondence to: Csaba Szepesvári <csaba.szepesvari@gmail.com>, Mengdi Wang <mengdiw@princeton.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

(Yang & Wang, 2019a).

The main contribution of this paper is a new model-based upper confidence RL algorithm. The main novelty is the criterion to select models that are deemed consistent with past data. As opposed to the most common approach where the models are selected based on their ability to predict next states or raw observations (cf. Jaksch et al. 2010; Yang & Wang 2019a or Strens 2000; Osband & Van Roy 2014; Abbasi-Yadkori & Szepesvári 2015; Ouyang et al. 2017; Agrawal & Jia 2017 in a Bayesian setting), we propose to evaluate models based on their ability to predict the values of a value function at next states, where the value function used is an estimate of the optimal that our algorithm produces based on past information. In essence, our algorithm selects models based on their ability to produce small prediction errors in an appropriately constructed *value-targeted* regression (VTR) problem. Our algorithm combines VTR for constructing sets of plausible models with (standard) optimistic planning. The idea of using a value function estimate to "fit" models has been explored in the context of batch RL by Farahmand (2018).

VTR is attractive for multiple reasons: *(i)* Firstly, VTR permits model learning to *focus on task-relevant aspects* of the transition dynamics. This is important as the dynamics can be quite complicated and in a resource bounded setting, modelling irrelevant aspects of the dynamics can draw valuable resources away from modelling task-relevant aspects. *(ii)* Secondly, VTR poses model learning as a real-valued regression problem, which should be easier than the usual approaches to build probabilistic models. In particular, when state-representation available to the agent takes values in a high-dimensional space then building a faithful probability model can be highly demanding. *(iii)* Thirdly, VTR aims to control directly what matters in terms of controlling the regret. Specifically the objective used in value-targeted is obtained from an expression that upper bounds the regret, hence it is natural to expect that minimizing value prediction errors will lead to a small regret.

An additional attractive feature of our algorithm is its modular structure. As a result, advances on the components (faster optimistic planning, tighter confidence sets for VTR) are directly translated into an improved algorithm. On the skeptical side, one may question whether VTR is going "too far" in ignoring details of the dynamics. In particular, since the value functions used in defining the regression targets are derived based on imperfect knowledge, the model may never get sufficiently refined in a way that would keep the regret small. Similarly, one may be worried about that by ignoring details of the observations (i.e., the identity of states), the approach advocated is ignoring information available in the data, which may slow down learning. This leads to central question of our article:

Is value-targeted regression sufficient and efficient for model-based online RL?

The main contribution of this paper is a positive answer to this question. In particular, the regret bounds we derive conclusively show that the despite the imperfection and non-stationarity of the value targets, our algorithm will not get "stuck" (i.e., it enjoys sublinear regret). Our results further suggest that in the worst case sense, for common settings, there may be no performance penalty associated with using value-targeted regression. We are careful here as this conclusion is based on comparing worst-case upper bounds, which cannot provide a definitive answer. Finally, it is worth noting that the regret bound does not depend on the size of either the state *or* the action space.

To complement the theoretical findings, results from a number of small-scale, synthetic experiments confirm that our algorithm is competitive in terms of its regret. The experiments also allow us to conclude that it is value-targeted regression *together* with optimistic planning that is effective. In particular, if optimism is taken away (i.e., ϵ-greedy for exploration), value-targeted regression performs worse than using a canonical approach to estimate the model. Similarly, if value-targeted regression is taken away, optimism together with the canonical model-estimation approach is less effective. Note that our results do not rule out that certain combinations of value-targeted regression and canonical model building are more effective than value-targeted regression. In fact, given the vast number of possibilities, we find this to be quite probable. While our proofs can be adjusted to deal with adding simultaneous alternative targets, sadly, our current theoretical tools are unable to capture the tradeoffs that one expects to arise as a result of such modifications.

It is interesting to note that, in an independent and concurrent work, value-targeted regression has also been suggested as the main model building tool of the MuZero algorithm (Schrittwieser et al., 2019), which was empirically evaluated on a number of RL benchmarks, such as the 57 Atari "games", the game of "Go", chess and shogi, and was found to be highly competitive with its state-of-the-art alternatives. This reinforces the conclusion that training models using value-targeted regression is indeed a good approach to build effective model-based RL algorithms. Since MuZero does *not* implement optimistic planning and our results show that optimism is not optional in a worst case sense, the good results of MuZero on these benchmark may seem to contradict our experimental findings that value-targeted regression is ineffective without an appropriate, 'smart' exploration component. However, there is no contradiction: Smart exploration may be optional in some environments; our experiments show that it is not optional on some environments. In short, for robust performance across a wide range of environments, smart exploration is necessary but

smart exploration may be optional in some environments.

2. Problem Formulation

We study learning to control episodic Markov decision processes (MDPs, for short), described by a tuple $M = (\mathcal{S}, \mathcal{A}, P, r, H, s_\circ)$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the transition kernel, r is a reward function, $H > 0$ is the episode length, or horizon, and $s_\circ \in \mathcal{S}$ is the initial state. In the online RL problem, the learning agent is given $\mathcal{S}$, $\mathcal{A}$, H and r but does not know P.[1] The agent interacts with its environment in a number of episodes. Each episode begins at state $s_\circ$ and ends after the agent made H decisions. At state $s \in \mathcal{S}$, the agent, after observing the state s, can choose an action $a \in \mathcal{A}$. As a result, the immediate reward $r(s, a)$ is incurred. Then the process transitions to a random next state $s' \in \mathcal{S}$ according to the transition law $P(\cdot|s, a)$.[2] The agent's goal is to maximize the total expected reward received over time.

If P is known, the behavior that achieves maximum expected reward over any number of episodes can be described by applying a deterministic policy π. Such a policy is a mapping from $\mathcal{S} \times [H]$ into $\mathcal{A}$ (note for a natural number n, $[n] = \{1, \dots, n\}$). Following the policy means that the agent upon encountering state s in stage h will choose action $\pi(s, h)$. In what follows, we will use $\pi_h(s)$ as an alternate notation, as this makes some of the formulae more readable. (We will follow the same convention of moving h to the subindex position when it comes to other functions whose domain is $\mathcal{S} \times [H]$.)

The value function $V^\pi : \mathcal{S} \times [H] \to \mathbb{R}$ of a policy π is defined via

$$V_h^\pi(s) = \mathbb{E}_\pi \left[\sum_{i=h}^{H} r(s_i, \pi(s_i)) \,\middle|\, s_h = s \right], \qquad s \in \mathcal{S},$$

where s_i is the state encountered at stage $i \in [H]$ and the subscript π (which we will often suppress) signifies that the probabilities underlying the expectation are jointly governed by π and P (P is suppressed for clarity). An optimal policy π^* and the optimal value function V^* are defined to be a policy and its value function such that $V_h^\pi(s)$ with $\pi = \pi^*$ achieves the maximum value among all possible policies for any $s \in \mathcal{S}$ and $h \in [H]$. Note that this is well-defined and in fact, as noted above, there is no loss of generality in restricting the search of optimal policies to deterministic policies.

[1] Our results are easy to extend to the case when r is not known.

[2] The precise definitions require measure-theoretic concepts (Bertsekas & Shreve, 1978), i.e., P is a Markov kernel, mapping from $\mathcal{S} \times \mathcal{A}$ to distributions over $\mathcal{S}$, hence, all these spaces need to be properly equipped with a measurability structure. For the sake of readability and also because they are well understood, we omit these technical details.

In online RL, a learning agent will use all past observations to come up with its decisions. The performance of such an agent is measured by its regret, which is the total reward the agent misses because she did not follow the optimal policy from the beginning. In particular, the total expected regret of an agent $\mathcal{A}$ across K episodes is given by

$$R(T) = \mathbb{E}\left[\sum_{k=1}^{K} \Big(V_1^*(s_1^k) - \sum_{h=1}^{H} r(s_h^k, a_h^k) \Big) \right], \quad (1)$$

where $T = KH$ is the total number of time steps that the agent interacts with its environment, $s_1^k = s_\circ$ is the initial state at the start of the k-th episode, and $s_1^1, a_1^1, \dots, s_H^k, a_H^k, s_1^2, a_1^2, \dots, s_H^2, a_H^2, s_1^K, a_1^K, \dots, s_H^K, a_H^K$ are the $T = KH$ state-action pairs in the order that they are encountered by the agent. The regret is sublinear if $R(T)/T \to 0$ as $T \to \infty$. As is well known, the worst-case value of $R(T)$ over a set of sufficiently large model class, grows at least as fast as $\sqrt{T}$ regardless of the algorithm used (e.g., Jaksch et al., 2010).

In this paper, we aim to design a general model-based reinforcement learning algorithm, with a guaranteed sublinear regret, for any (not too large) family of transition models:

Assumption 1 (Known Transition Model Family)**.** *The unknown transition model P belongs to a family of models $\mathcal{P}$ which is available to the learning agent. The elements of $\mathcal{P}$ are transition kernels mapping state-action pairs to signed distributions over $\mathcal{S}$.*

That we allow signed distributions only increases generality; this may be important when one is given a model class that can be compactly represented but only when it also includes non-probability kernels (see Pires & Szepesvári 2016 for a discussion of this). Parametric and nonparametric transition models are common in modelling complex stochastic controlled systems. For example, robotic systems are often smoothly parameterized by unknown mechanical parameters such as friction and parameters that describe the geometry of the robot.

An important special case is the class of linear mixture models:

Definition 1 (Linear Mixture Models)**.** *We say that $\mathcal{P}$ is the class of linear mixture models with component models $P_1, \dots, P_d$ if $P_1, \dots, P_d$ are transition kernels that map state-action pairs to signed measures and $P \in \mathcal{P}$ if and only if there exists $\theta \in \mathbb{R}^d$ such that for all $(s, a) \in \mathcal{S} \times \mathcal{A}$,*

$$P(ds'|s, a) = \sum_{j=1}^{d} \theta_j P_j(ds'|s, a) = P_\cdot(ds'|s, a)^\top \theta\,. \quad (2)$$

The linear mixture model can be viewed as a way of aggregating a number of known basis models as considered by

(Modi et al., 2019). We can view each $P_j(\cdot|\cdot)$ as a basis latent "mode". When θ is restricted to lie in the $(d-1)$ simplex, the actual transition is a probabilistic mixture of these latent modes. As an example of when mixture models arise, consider large-scale queueing networks where the arrival rate and job processing speed for each queue is not known. By using a discrete-time Bernoulli approximation, the transition probability matrix from time t to $t+\Delta t$ becomes increasingly close to linear with respect to the unknown arrival/processing rates as $\Delta t \to 0$. In this case, it is common to model the discrete-time state transition as a linear aggregation of arrival/processing processes with unknown parameters (Kovalenko, 1968).

Another interesting special case is the linear-factored MDP model of (Yang & Wang, 2019a) where, assuming a discrete state space for a moment, P takes the form

$$P(s'|s,a) = \phi(s,a)^\top M \psi(s') = \sum_{i=1}^{d_1}\sum_{j=1}^{d_2} M_{ij}\left[\psi_j(s')\phi_i(s,a)\right],$$

where $\phi(s,a) \in \mathbb{R}^{d_1}, \psi(s') \in \mathbb{R}^{d_2}$ are given features for every $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ (when the state space is continuous, ψ becomes an $\mathbb{R}^{d_2}$-valued measure over $\mathcal{S}$). The matrix $M \in \mathbb{R}^{d_1 \times d_2}$ is an unknown matrix and is to be learned. It is easy to see that the factored MDP model is a special case of the linear mixture model (19) with each $\psi_j(s')\phi_i(s,a)$ being a basis model (this should be replaced by $\psi_j(ds')\phi_i(s,a)$ when the state space is continuous). In this case, the number of unknown parameters in the transition model is $d = d_1 \times d_2$. In this setting, without additional assumptions, our regret bound will match that of (Yang & Wang, 2019a).

3. Upper Confidence RL with Value-Targeted Model Regression

Our algorithm (Alg 1) can be viewed as a generalization of UCRL (Jaksch et al., 2010), following ideas of (Osband & Van Roy, 2014). In particular, at the beginning of episode $k = 1, 2, \ldots, K$, the algorithm first computes a subset B_k of the model class $\mathcal{P}$ that contains the set of models that are deemed to be consistent with all the data that has been collected in the past. The new idea, value-targeted regression is used in the construction of B_k. The details of how this is done are postponed to a later section. Given B_k, the algorithm needs to find the model that maximizes the optimal value, and the corresponding optimal policy. Denoting by V_P^* the optimal value function under a model P, this amounts to finding the model $P \in B_k$ that maximizes the value $V_{P,1}^*(s_1^k)$. Given the model P_k that maximizes this value, an optimal policy is extracted from the model as in standard dynamic programming, detailed in the next section.

Algorithm 1 UCRL-VTR

Input: Family of MDP models $\mathcal{P}$, $d, H, T = KH$, sequence $\{\beta_k\}_{k=1,2,\ldots}$.
$B_1 = \mathcal{P}$
for $k = 1, 2, \ldots, K$ **do**
  Observe the initial state s_1^k of episode k
  Optimistic planning:
    $P_k = \text{argmax}_{P' \in B_k} V_{P',1}^*(s_1^k)$
    Compute $Q_{1,k}, \ldots Q_{H,k}$ for P_k using (3)
  for $h = 1, 2, \ldots, H$ **do**
    Choose the next action greedily with respect to $Q_{h,k}$:
    $$a_h^k = \arg\max_{a \in \mathcal{A}} Q_{h,k}(s_h^k, a)$$
    Observe state s_{h+1}^k
    Compute and store value predictions: $y_{h,k} \leftarrow V_{h+1,k}(s_{h+1}^k)$
  end for
  Construct confidence set using value-targeted regression as described in Section 3.2:
  $$B_{k+1} = \{P' \in \mathcal{P} | L_{k+1}(P', \hat{P}_{k+1}) \le \beta_k\}$$
end for

At the end of the episode, the data collected is used to refine the confidence set B_k.

3.1. Model-Based Optimistic Planning

Upper confidence methods are prominent in sequential online learning. As noted before, we let

$$P_k = \text{argmax}_{P' \in B_k} V_{P',1}^*(s_1^k).$$

Given model P_k, the optimal policy for P_k can be computed using dynamic programming. In particular, for $1 \le h \le H+1$, define

$$\begin{aligned} &Q_{H+1,k}(s,a) = 0, \\ &V_{h,k}(s) = \max_{a \in \mathcal{A}} Q_{h,k}(s,a), \\ &Q_{h,k}(s,a) = r(s,a) + \langle P_k(\cdot|s,a), V_{h+1,k}\rangle, \end{aligned} \tag{3}$$

where for a measure μ and function f that share a common domain, $\langle \mu, f\rangle$ denotes the integral of f with respect to μ. It follows that, taking the action at state s and stage h that maximizes $Q_{h,k}(s,\cdot)$ gives an optimal policy for model P_k. As long as $P \in B_k$ with high probability, the preceding calculation gives an optimistic (that is, upper) estimate of value of an episode. Next, we show how to construct the confidence set B_k.

3.2. Value-Targeted Regression for Confidence Set Construction

Every time we observe a transition (s, a, s') with $s' \sim P(\cdot|s, a)$, we receive information about the model P. A standard approach to use this information would be either using a maximum likelihood approach, or regressing "onto" s'. As our goal is not to find the best model, we propose an alternate approach where we set up a regression problem where the model is used to predict the value assigned to s' by our more recent value function estimate:

$$\hat{P}_{k+1} = \text{argmin}_{P' \in \mathcal{P}} \sum_{k'=1}^{k} \sum_{h=1}^{H} L'(P'), \tag{4}$$

$$L'(P') = \left(\langle P'(\cdot|s_h^{k'}, a_h^{k'}), V_{h+1,k'} \rangle - y_{h,k'} \right)^2$$

$$y_{h,k'} = V_{h+1,k'}(s_{h+1}^{k'}), \quad h \in [H], k' \in [k].$$

In the above regression procedure, the regret target keeps changing as the algorithm refines the value estimates. This is in contrast to typical supervised learning for building models, where the regression targets are often fixed objects (such as raw observations, features or keypoints; e.g. (Jaksch et al., 2010; Osband & Van Roy, 2014; Abbasi-Yadkori & Szepesvári, 2015; Xie et al., 2016; Agrawal & Jia, 2017; Yang & Wang, 2019a; Kaiser et al., 2019)). For a confidence set construction, we get inspiration from Proposition 5 in the paper of (Osband & Van Roy, 2014). The set is centered at $\hat{P}_{k+1}$. Defining

$$L_{k+1}(P, \hat{P}_{k+1}) = \sum_{k'=1}^{k} \sum_{h=1}^{H} \left(\langle P(\cdot|s_h^{k'}, a_h^{k'}) - \hat{P}_{k+1}(\cdot|s_h^{k'}, a_h^{k'}), V_{h+1,k'} \rangle \right)^2$$

we let

$$B_{k+1} = \{P' \in \mathcal{P} \mid L_{k+1}(P', \hat{P}_{k+1}) \le \beta_{k+1}\},$$

where the value of β_k is obtained using a calculation similar to that done in Proposition 5 of the paper of (Osband & Van Roy, 2014). In turn, this calculation is based on the nonlinear least-squares confidence set construction from Russo & Van Roy (2014), which we describe and refine in the appendix. It is not hard to see that the confidence set can also be written in the alternative form

$$B_{k+1} = \{P' \in \mathcal{P} \mid \tilde{L}_{k+1}(P') \le \tilde{\beta}_{k+1}\}$$

with a suitably defined $\tilde{\beta}_{k+1}$ and where

$$\tilde{L}_{k+1}(P') = \sum_{k'=1}^{k} \sum_{h=1}^{H} \left(\langle P'(\cdot|s_h^{k'}, a_h^{k'}), V_{h+1,k'} \rangle - y_{h,k'} \right)^2.$$

Note that the above formulation strongly exploits that the MDP is time-homogeneous: The same transition model is used at all stages of an episode. When the MDP is time-inhomogeneous, the construction can be easily modified to accommodate this.

3.3. Implementation of UCRL-VTR

Algorithm 1 gives a general and modular template for model-based RL that is compatible with regression methods/optimistic planners. While the algorithm is conceptually simple, and the optimization and evaluation of the loss in value-targeted regression appears to be at advantage in terms of computation as compared to standard approaches typically used in model-based RL, the implementation of UCRL-VTR is nontrivial in general and for now it requires a case-by-base design.

Computation efficiency of the algorithm depends on the specific family of models chosen. For the linear-factor MDP model considered by Yang & Wang (2019a), the regression is linear and admits efficient implementation; further, optimistic planning for this model can be implemented in poly(d) time by using Monte-Carlo simulation and sketching as argued in the cited paper. Other ideas include loosening the confidence set to come up with computationally tractable methods, or relaxing the requirement that the same model is used in all stages. This latter idea is what we use in our experiments. In the general case, optimistic planning is computationally intractable. However, we expect that randomized (e.g. (Osband et al., 2017; 2014; Lu & Van Roy, 2017)) and approximate dynamic programming methods (tree search, roll out, see e.g., (Bertsekas & Tsitsiklis, 1996)) will often lead to tractable and good approximations. As was mentioned above, in some special cases these have been rigorously shown to work. In similar settings, the approximation errors are known to mildly impact the regret (Abbasi-Yadkori & Szepesvári, 2015) and we expect the same to hold in our setting. If we look beyond methods with rigorous guarantees, there are practical deep RL algorithms that implement parts of UCRL-VTR. As mentioned earlier, the MuZero algorithm of Schrittwieser et al. (2019) is a state-of-the-art algorithm on multiple domains and this algorithm implements value-targeted-regression to learn a model which is fed to a planner that uses Monte Carlo tree search, although the planner does not implement optimistic planning.

4. Theoretical Analysis

We will need the concept of Eluder dimension due to (Russo & Van Roy, 2014). Let $\mathcal{F}$ be a set of real-valued functions with domain $\mathcal{X}$. For $f \in \mathcal{F}$, $x_1, \ldots, x_t \in \mathcal{X}$, introduce the notation $f|_{(x_1,\ldots,x_t)} = (f(x_1), \ldots, f(x_t))$. We say that $x \in \mathcal{X}$ is ϵ-independent of $x_1, \ldots, x_t \in \mathcal{X}$ given $\mathcal{F}$ if there exists $f, f' \in \mathcal{F}$ such that $\|(f - f')|_{(x_1,\ldots,x_t)}\|_2 \le \epsilon$ while $f(x) - f'(x) > \epsilon$.

Definition 2 (Eluder dimension, Russo & Van Roy 2014)**.** The Eluder dimension $\dim_{\mathcal{E}}(\mathcal{F}, \epsilon)$ of $\mathcal{F}$ at scale ϵ is the length of the longest sequence $(x_1, \ldots, x_n)$ in $\mathcal{X}$ such that for some $\epsilon' \ge \epsilon$, for any $2 \le t \le n$, x_t is ϵ'-independent of

$(x_1, \dots, x_{t-1})$ given $\mathcal{F}$.

Let $\mathcal{V}$ be the set of optimal value functions under some model in $\mathcal{P}$: $\mathcal{V} = \{V^*_{P'} : P' \in \mathcal{P}\}$. Note that $\mathcal{V} \subset \mathcal{B}(\mathcal{S}, H)$, where $\mathcal{B}(\mathcal{S}, H)$ denotes the set of real-valued measurable functions with domain $\mathcal{S}$ that are bounded by H. We let $\mathcal{X} = \mathcal{S} \times \mathcal{A} \times \mathcal{V}$. Choose $\mathcal{F}$ to be the collection of functions $f : \mathcal{X} \to \mathbb{R}$ as follows:

$$\mathcal{F} = \left\{ f \,\middle|\, \begin{array}{l} \exists P \in \mathcal{P} \text{ s.t. for any } (s,a,v) \in \mathcal{S} \times \mathcal{A} \times \mathcal{V} \\ f(s,a,v) = \int P(ds'|s,a) v(s') \end{array} \right\}. \tag{5}$$

Note that $\mathcal{F} \subset \mathcal{B}(\mathcal{X}, H)$. For a norm $\|\cdot\|$ on $\mathcal{F}$ and $\alpha > 0$ let $\mathcal{N}(\mathcal{F}, \alpha, \|\cdot\|)$ denote the $(\alpha, \|\cdot\|)$-covering number of $\mathcal{F}$. That is, this if $m = \mathcal{N}(\mathcal{F}, \alpha, \|\cdot\|)$ then one can find m elements of $\mathcal{F}$ such that any element in $\mathcal{F}$ is at most α away from one of these elements in norm $\|\cdot\|$. Denote by $\|\cdot\|_\infty$ the supremum norm: $\|f\|_\infty = \sup_{x \in \mathcal{X}} |f(x)|$.

Define the K-episode *pseudo-regret* as

$$R_K = \sum_{k=1}^{K} \left(V^*(s_0^k) - V^{\pi_k}(s_0^k) \right) .$$

Clearly, $R(KH) = \mathbb{E} R_K$ holds for any $K > 0$ where $R(T)$ is the expected regret after T steps of interaction as defined in (1). Thus, to study the expected regret, it suffices to study R_K. Our main result is as follows.

Theorem 1 (Regret of Algorithm 1)**.** *Let Assumption 1 hold and let $\alpha \in (0,1)$. For $k > 0$ let β_k be*

$$\begin{aligned} \beta_k &= 2H^2 \log\left(\frac{2\mathcal{N}(\mathcal{F}, \alpha, \|\cdot\|_\infty)}{\delta} \right) \\ &+ 2H(kH-1)\alpha \left\{ 2 + \sqrt{\log\left(\frac{4kH(kH-1)}{\delta} \right)} \right\} . \end{aligned} \tag{6}$$

Then, for any $K > 0$, with probability $1 - 2\delta$,

$$\begin{aligned} R_K &\le \alpha + H(d \wedge K(H-1)) + 4\sqrt{d \beta_K K(H-1)} \\ &+ H\sqrt{2K(H-1)\log(1/\delta)} , \end{aligned}$$

where $d = \dim_{\mathcal{E}}(\mathcal{F}, \alpha)$ is the Eluder dimension with $\mathcal{F}$ given by (5).

A typical choice of α is $\alpha = 1/(KH)$. In the special case of linear transition model, Theorem 1 implies a worst-case regret bound that depends linearly on the number of parameters.

Corollary 2 (Regret of Algorithm 1 for Linearly–Parametrized Transition Model)**.** *Let $P_1, \dots, P_d$ be d transition models, $\Theta \subset \mathbb{R}^d$ a nonempty set with diameter R measured in $\|\cdot\|_1$ and let $\mathcal{P} = \{\sum_j \theta_j P_j : \theta \in \Theta\}$. Then, for any $0 < \delta < 1$, with probability at least $1 - \delta$, the pseudo-regret R_K of Algorithm 1 when it uses the confidence sets given in Theorem 1 satisfies*

$$R_K = \tilde{O}(d\sqrt{H^3 K \log(1/\delta)}) .$$

We also provide a lower bound for the regret in our model. The proof is by reduction to a known lower bound and is left to Appendix B.

Theorem 3 (Regret Lower Bound)**.** *For any $H \ge 1$ and $d \ge 8$, there exist a state space $\mathcal{S}$ and action set $\mathcal{A}$, a reward function $r : \mathcal{S} \times \mathcal{A} \to [0,1]$, d transition models $P_1, \dots, P_d$ and a set $\Theta \subset \mathbb{R}^d$ of diameter of at most one such that for any algorithm there exists $\theta \in \Theta$ such that for sufficiently large number of episodes K, the expected regret of the algorithm on the H-horizon MDP with reward r and transition model $P = \sum_j \theta_j P_j$ is at least $\Omega(H\sqrt{dK})$.*

Rusmevichientong & Tsitsiklis (2010) gave a regret lower bound of $\Omega(d\sqrt{T})$ for linearly parameterized bandit with actions on the unit sphere (see also Section 24.2 of Lattimore & Szepesvári 2020). Our regret upper bound matches this bandit lower bound in d, T. Whether the upper or lower bound is tight (or none of them) remains to be seen. The theorems validate that, in the setting we consider, it is sufficient to use the predicted value functions as regression targets. That for the special case of linear mixture models the lower bound is close to the upper bound appears to suggest that little benefit if any can be derived from fitting the transition model to predict future observations. We conjecture that this is in fact true when considering the worst-case regret. Of course, a conclusion that is concerned with the worst-case regret has no implication for the behavior of the respective methods on particular MDP instances. We note in passing that by appropriately increasing β_k, the regret upper bounds can be extended to the so-called misspecified case when P can be outside of $\mathcal{P}$ (for related results, see, e.g., Jin et al. 2019; Lattimore & Szepesvári 2019). However, the details of this are left for future work.

Further, our method applies to handle the case where the linearly parameterized transition model is sparse. Suppose that model parameter θ is known to have at most s nonzero entries. In this case, the class of sparse linear models has a much smaller covering number, and the regret would improve and depend on both d,s. Details of this are left for future work.

4.1. Regret Bound with Model Misspecification

Next, we consider the case where the model family $\mathcal{P}$ does not exactly realize the true transition model P:

Assumption 2 (Model with misspecification error)**.** *The model family $\mathcal{P}$ ε-approximates P in the sense that there exists $P^* \in \mathcal{P}$ such that*

$$\sup_{s,a} \|P(\cdot|s,a) - P^*(\cdot|s,a)\|_{TV} \le \varepsilon, \tag{7}$$

where $\|\cdot\|_{TV}$ denotes the total variation distance.

This assumption indicates that the true transition model P of the MDP is close to the family $\mathcal{P}$, and ε measures the worst-case deviation. We handle the misspecification error by slightly enlarging the confidence set. This allows us to obtain a regret bound similar to our previous result with an additional linear term that is proportional to the misspecification error and slightly larger constants:

Theorem 4. *Let Assumption 2 hold, $\alpha, \delta \in (0,1)$. We choose β_k be*

$$\beta_k = 8H^2 \log\left(\frac{4\mathcal{N}(\mathcal{F}, \alpha, \|\cdot\|_\infty)}{\delta}\right) + 4H(kH-1)\alpha\left\{2 + \sqrt{\log\left(\frac{8kH(kH-1)}{\delta}\right)}\right\} + 8H^3 k\varepsilon^2.$$

Then, for any $K > 0$, with probability $1-2\delta$, the K-episode pseudo-regret R_K of Algorithm 1 satisfies

$$R_K \le \alpha + H(d \wedge K(H-1)) + 4\sqrt{d\beta_K K(H-1)} + H\sqrt{2K(H-1)\log(1/\delta)} + H^2 K\varepsilon,$$

where $d = \dim_{\mathcal{E}}(\mathcal{F}, \alpha)$ with $\mathcal{F}$ given by (5).

The proof of this theorem is given in Appendix D.

5. Related Work

A number of prior efforts have established efficient RL methods with provable regret bounds. For tabular H-horizon MDP with S states and A actions, there have been results on model-based methods (e.g., Jaksch et al. 2010; Osband et al. 2014; Azar et al. 2017; Dann et al. 2017; Agrawal & Jia 2017; Dann et al. 2018; Kakade et al. 2018), and on model-free methods (e.g., Osband et al. 2017; Jin et al. 2018; Zhang et al. 2020). Both model-based and model-free methods are known to achieve a regret of $\tilde{\mathcal{O}}(\sqrt{H^2SAT})$, where $\tilde{O}(\cdot)$ hides log factors, T denotes the total number of timesteps and the bound applies to the non-stationary setting (when the transition models are not shared across stages). Moreover, apart from logarithmic factors, this bound is known to be unimprovable Jaksch et al. (2010); Osband et al. (2017); Jin et al. (2018); Zhang et al. (2020). There have been significant theoretical and empirical advances on RL with function approximation, including but not limited to (Baird, 1995; Tsitsiklis & Van Roy, 1997; Parr et al., 2008; Mnih et al., 2013; 2015; Silver et al., 2017; Yang & Wang, 2019b; Bradtke & Barto, 1996). Under the assumption that the optimal action-value function is captured by linear features, Zanette et al. (2019) considers the case when the features are "extrapolation friendly" and a simulation oracle is available, Wen & Van Roy (2013; 2017) tackle problems where the transition model is deterministic, Du et al. (2019) deals with a relaxation of the deterministic case when the transition model has low variance. Yang & Wang (2019b) considers the case of linear factor models, while Lattimore & Szepesvári (2019) considers the case when all the action-value functions of all deterministic policies are well-approximated using a linear function approximator. These latter works handle problems when the algorithm has access to a simulation oracle of the MDP. As for regret minimization in RL using linear function approximation, Yang & Wang (2019a) assumed the transition model admits a matrix embedding of the form $P(s'|s,a) = \phi(s,a)^\top M\psi(s')$, and proposed a model-based MatrixRL method with regret bounds $\tilde{\mathcal{O}}(H^2 d\sqrt{T})$ with stronger assumptions and $\tilde{\mathcal{O}}(H^2 d^2\sqrt{T})$ in general, where d is the dimension of state representation $\phi(s,a)$. Jin et al. (2019) studied the setting of linear-factor MDP and constructed a model-free least-squares action-value iteration algorithm, which was proved to achieve the regret bound $\tilde{\mathcal{O}}(\sqrt{H^3 d^3 T})$. Modi et al. (2019) considered a related setting where the transition model is an ensemble involving state-action-dependent features and basis models and proved a sample complexity $\frac{d^3K^2H^2}{\epsilon^2}$ where d is the feature dimension, K is the number of basis models and $d \cdot K$ is their total model complexity. As for RL with a general model class, in their seminal work, Osband & Van Roy (2014) provided a general posterior sampling RL method that works for any given classes of reward and transition functions. It established a Bayesian regret upper bound $O(\sqrt{d_K d_E T})$, where d_K and d_E are the Kolmogorov and the Eluder dimensions of the model class. In the case of linearly parametrized transition model (Assumption 1 of this paper), this Bayesian regret becomes $O(d\sqrt{T})$, and our worst-case regret result matches with the Bayesian one. Abbasi-Yadkori & Szepesvári (2015); Theocharous et al. (2017) also considered the Bayesian regret and, in particular, Abbasi-Yadkori & Szepesvári (2015) considered a smooth parameterization. To the authors' best knowledge, there are no prior works addressing the problem of designing low-regret algorithms for MDPs with linearly or non-linearly parameterized transition models. Model based PAC RL algorithms have been studied by Sun et al. (2019), who essentially adopt the value-aware loss of Farahmand et al. (2017), who considered this loss in the batch setting. Farahmand (2018) refines the work of Farahmand et al. (2017) by changing the algorithm to be similar to what we use here: In every iteration of fitted Q-iteration, first a model is obtained by minimizing the value prediction loss measured with the last value function, after which this model is used to obtain the next action-value function. The main result bounds the suboptimality of the policy that is greedy with respect to the last action-value function. A preliminary version of our paper was presented at L4DC.

Exploration/ Targets	Optimism	Dithering
Next states	UC-MatrixRL	EG-Freq
Values	UCRL-VTR	EG-VTR
Mixed	UCRL-Mixed	EG-Mixed

Table 1. Legend to the algorithms compared. Note that UC-MatrixRL of Yang & Wang (2019a) in the tabular case essentially becomes UCRL of Jaksch et al. (2010).

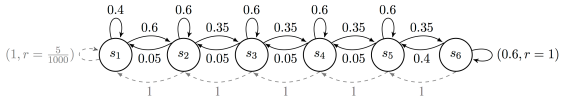

Figure 1. The "RiverSwim" environment with 6 states. State s_1 has a small associated reward, state s_6 has a large associated reward. The action whose effect is shown with the dashed arrow deterministically "moves the agent" left. The other action is stochastic, and with relatively high probability moves the agent towards state s_6: This represents swimming "against the current".

6. Numerical Experiments

The goal of our experiments is to provide insight into the benefits and/or pitfalls of using value-targets for fitting models, both with and without optimistic planning. We run our experiments in the tabular setting as it is easy to keep aspects of the test environments under control while avoiding approximate computations. Note that tabular environments are a special case of the linear model where $P_j(s'|s,a) = \mathbb{I}(j = f(s,a,s'))$, where $j \in [S^2A]$ and f is a bijection that maps its arguments to the set $[S^2A]$, making $d = S^2A$. The objective is either to minimize mean-squared error of predicting next states (alternatively, maximize log-likelihood of observed data), which leads to frequency based model estimates, or it is to minimize the value targets as proposed in our paper. The other component of the algorithms is whether they implement optimistic planning, or planning with the nominal model and then implementing an ϵ-greedy policy with respect to the estimated model ("dithering"). In the case of optimistic planning, the algorithm that uses mixed targets uses a union bound and takes the smallest value upper confidence bounds amongst the two bounds obtained with the two model-estimation methods. These leads to six algorithms, as shown in Table 1. Results for the "mixed" variants are very similar to the variant that uses VTR and can be found in Appendix G In the experiments we use confidence bounds that are specialized to the linear case. For details of these, see Appendix F. For ϵ-greedy, we optimize the value of ϵ in each environment to get the best results. This gives ϵ-greedy an "unfair advantage". As we shall see it soon, despite this advantage, ϵ-greedy will not fair particularly well in our experiments.

6.1. Measurements

We report the cumulative regret as a function of the number of episodes and the weighted model error to indicate how well the model is learned. The results are obtained from 30 independent runs for the ϵ-greedy algorithms and 10 independent runs for the UC algorithms. The weighted model error reported is as follows. Given the model estimate $\hat{P}$, we compute

$$E(\hat{P}) = \sum_{s,a} \sum_{s'} \frac{N(s,a,s')}{N'(s,a)} |\hat{P}(s' \mid s,a) - P^*(s' \mid s,a)|, \tag{8}$$

where N' is the observation-count of the state-action pair (s,a), N is the count of transitioning to s' from (s,a), and P^* is the true dynamics model. The weighting is introduced so that an algorithm that discards a state-action pair is not (unduly) penalized.

6.2. Results for RiverSwim

The schematic diagram of the RiverSwim environment is shown in Figure 1. RiverSwim consists of S states arranged in a chain. The agent begins on the far left and has the choice of swimming left or right at each state. There is a current that makes swimming left much easier than swimming right. Swimming left with the current always succeeds in moving the agent left, but swimming right against the current sometimes moves the agent right (with a small probability of moving left as well), but more often than not leaves the agent in the current state. Thus smart exploration is a necessity to learn a good policy in this environment. We experiment with small environments with $S = 5$ and set the horizon to $H = 20$. The optimal value of the initial state is 5.6 for our five-state RiverSwim. The initial state is the leftmost state (s_1 in the diagram). The value that we found to work the best for EGRL-VTR is $\epsilon = 0.2$ and the value that we found to work best for EG-Freq is $\epsilon = 0.12$. The results are shown in Figure 2. The regret per episode for an algorithm that "does not learn" is expected to be in the same range as the respective optimal values. Based on this we see that 10^5 episodes is barely sufficient for the algorithms other than UCRL-VTR to learn a good policy. Looking at the model errors we see that EGRL-VTR is doing quite poorly, EG-Freq is also lacking, the others are doing reasonably well. However, this is because EG-Freq visits more uniformly than the other methods the various state-action pairs. The results clearly indicate that *(i)* fitting to the state-value function alone provides enough of a signal for learning as evident by UCRL-VTR obtaining low regret as predicted by our theoretical results, and that

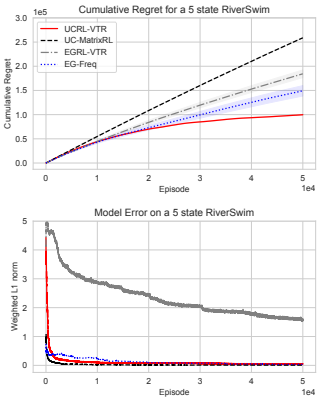

Figure 2. The results for the ϵ-greedy algorithms are averaged over thirty runs and error bars are reported for the regret plots.

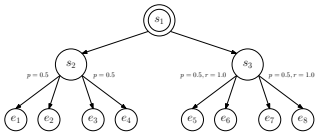

Figure 3. An eleven state WideTree MDP. The algorithm starts in the initial state s_1. From the initial state s_1 the algorithm has a choice of either deterministically transitioning to either state s_2 or state s_3. Finally from either state s_2 or state s_3 the algorithm picks one of two possible actions and transitions to one of the terminal states e_i. The choice of the initial action determines the delayed reward the algorithm will observe.

(ii) optimism is necessary when using VTR to achieve good results, as evident by UCRL-VTR achieving significantly better regret than EGRL-VTR and even in the smaller RiverSwim environment. It is also promising that value-targeted regression with optimistic exploration outperformed optimism based on the "canonical" model estimation procedure. We attribute this to the fact that value-targeted regression will learn a model faster that predicts the optimal values well than the canonical, frequency based approach. That value-targeted regression also learns a model with small weighted error appears to be an accidental feature of this environment. Our next experiments are targeted at further exploring whether VTR can be effective *without* learning a good model.

6.3. WideTree

We introduce a novel tabular MDP we call WideTree. The WideTree environment has a fixed horizon $H = 2$ and $S = 11$ states. A visualization of the WideTree environment is shown in Figure 3. In WideTree, an agent starts at the initial state s_1. The agent then progresses to one of the many bottom terminal states and collects a reward of either 0 or 1. The only significant action is whether to transition from s_1 to either s_2 or s_3. Note that the model in the second layer is irrelevant for making a good decision: Once in s_3, all actions lead to a reward of one, and once in s_2, all actions lead to a reward of zero. We vary the number of bottom states reachable from states s_2 and s_3 while still maintaining a reward structure but the results here are shown with $S = 11$. We set $\epsilon = 0.1$ in this environment, as this allows the model error of EG-Freq to match that of UC-MatrixRL. The results are shown in Figure 4. Both UCRL-VTR and EG-VTR learn equally poor models (their graphs are 'on the top of each other'). Yet, UCRL-VTR manages to quickly learn a good policy, as attested by its low regret. EG-Freq and EG-VTR perform equally poorly and UC-MatrixRL is even slower as it keeps exploring the environment. These experiments clearly illustrate that UCRL-VTR is able to achieve good results without learning a good model – its focus on values makes pays off swiftly in this well-chosen environment.

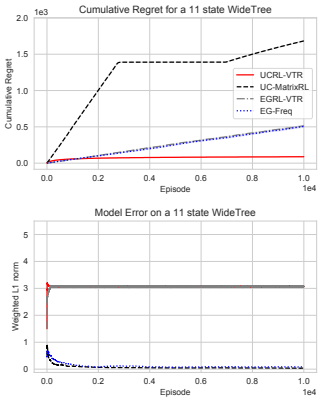

Figure 4. The results for the ϵ-greedy algorithms are averaged over thirty runs and error bars are reported for the regret plots.

7. Conclusions

We considered online learning in episodic MDPs and proposed an optimistic model-based reinforcement learning method (UCRL-VTR) with the unique characteristic to evaluate and select models based on their ability to predict value functions that the algorithm constructs during learning. The regret of the algorithm was shown to be bounded by a quantity that relates to the richness of the model class through the Eluder dimension and the metric entropy of an appropriately constructed function space. For the case of linear mixture models, the regret bound simplifies to $\tilde{O}(d\sqrt{H^3T})$ where d is the number of model parameters, H is the horizon, and T is the total number of interaction steps. Our experiments confirmed that the value-targeted regression objective is not only theoretically sound, but also yields a competitive method which allows task-focused model-tuning: In a carefully chosen environment we demonstrated that the algorithm achieves low regret despite that it ignores modeling a major part of the environment.

8. Acknowledgements

Csaba Szepesvári gratefully acknowledges funding from the Canada CIFAR AI Chairs Program, Amii and NSERC. Mengdi Wang gratefully acknowledges funding from the U.S. National Science Foundation (NSF) grant CMMI-1653435, Air Force Office of Scientific Research (AFOSR) grant FA9550-19-1-020, and C3.ai DTI.

References

Abbasi-Yadkori, Y. and Szepesvári, C. Bayesian optimal control of smoothly parameterized systems. In *UAI*, pp. 1–11, 2015.

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pp. 2312–2320, 2011.

Agrawal, S. and Jia, R. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In *Advances in Neural Information Processing Systems*, pp. 1184–1194, 2017.

AlQuraishi, M. AlphaFold at CASP13. *Bioinformatics*, 35 (22):4862–4865, 2019.

Arulkumaran, K., Cully, A., and Togelius, J. Alphastar: An evolutionary computation perspective. *arXiv preprint arXiv:1902.01724*, 2019.

Azar, M. G., Osband, I., and Munos, R. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 263–272. JMLR. org, 2017.

Baird, L. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pp. 30–37. Elsevier, 1995.

Bertsekas, D. P. and Shreve, S. *Stochastic optimal control: the discrete-time case*. Academic Press, 1978.

Bertsekas, D. P. and Tsitsiklis, J. N. *Neuro-dynamic programming*. Athena Scientific, 1996.

Bradtke, S. J. and Barto, A. G. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22(1-3):33–57, 1996.

Dann, C., Lattimore, T., and Brunskill, E. Unifying PAC and regret: Uniform PAC bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 5713–5723, 2017.

Dann, C., Li, L., Wei, W., and Brunskill, E. Policy certificates: Towards accountable reinforcement learning. *arXiv preprint arXiv:1811.03056*, 2018.

Du, S. S., Luo, Y., Wang, R., and Zhang, H. Provably efficient Q-learning with function approximation via distribution shift error checking oracle. *arXiv preprint arXiv:1906.06321*, 2019.

Farahmand, A.-M. Iterative value-aware model learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 9090–9101, 2018.

Farahmand, A.-M., Barreto, A., and Nikovski, D. Value-aware loss function for model-based reinforcement learning. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1486–1494. PMLR, 2017.

Jaksch, T., Ortner, R., and Auer, P. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. Is Q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pp. 4863–4873, 2018.

Jin, C., Yang, Z., Wang, Z., and Jordan, M. I. Provably efficient reinforcement learning with linear function approximation. *arXiv preprint arXiv:1907.05388*, 2019.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Mohiuddin, A., Sepassi, R., Tucker, G., and Michalewski, H. Model-based reinforcement learning for Atari. In *ICLR*, 2019.

Kakade, S., Wang, M., and Yang, L. F. Variance reduction methods for sublinear reinforcement learning. *arXiv preprint arXiv:1802.09184*, 2018.

Kober, J., Bagnell, J. A., and Peters, J. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Kovalenko, B. G. I. N. *Introduction to queueing theory*. Israel Program for Scientific Translation, Jerusalem, 1968.

Lattimore, T. and Szepesvári, C. Learning with good feature representations in bandits and in RL with a generative model, 2019.

Lattimore, T. and Szepesvári, C. *Bandit Algorithms*. Cambridge University Press, 2020. (to appear).

Lu, X. and Van Roy, B. Ensemble sampling. In *Advances in neural information processing systems*, pp. 3258–3266, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529, 2015.

Modi, A., Jiang, N., Tewari, A., and Singh, S. Sample complexity of reinforcement learning using linearly combined model ensembles. *arXiv preprint arXiv:1910.10597*, 2019.

Osband, I. and Van Roy, B. Model-based reinforcement learning and the Eluder dimension. In *Advances in Neural Information Processing Systems*, pp. 1466–1474, 2014.

Osband, I. and Van Roy, B. On lower bounds for regret in reinforcement learning. *arXiv preprint arXiv:1608.02732*, 2016.

Osband, I., Van Roy, B., and Wen, Z. Generalization and exploration via randomized value functions. *arXiv preprint arXiv:1402.0635*, 2014.

Osband, I., Van Roy, B., Russo, D., and Wen, Z. Deep exploration via randomized value functions. *arXiv preprint arXiv:1703.07608*, 2017.

Ouyang, Y., Gagrani, M., Nayyar, A., and Jain, R. Learning unknown Markov decision processes: A Thompson sampling approach. In *Advances in Neural Information Processing Systems*, pp. 1333–1342, 2017.

Parr, R., Li, L., Taylor, G., Painter-Wakefield, C., and Littman, M. L. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 752–759. ACM, 2008.

Pires, B. and Szepesvári, C. Policy error bounds for model-based reinforcement learning with factored linear models. In *COLT*, pp. 121–151, 2016.

Rusmevichientong, P. and Tsitsiklis, J. N. Linearly parameterized bandits. *Mathematics of Operations Research*, 35 (2):395–411, 2010.

Russel, S. and Norvig, P. *Artificial Intelligence – a modern approach*. Prentice Hall, 2003.

Russo, D. and Van Roy, B. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39 (4):1221–1243, 2014.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., , Lillicrap, T., and Silver, D. Mastering Atari, Go, chess and shogi by planning with a learned model. *arXiv preprint arXiv:1911.08265*, 2019.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., Driessche, G. v. d., Graepel, T., and Hassabis, D. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Strens, M. J. A. A Bayesian framework for reinforcement learning. In *ICML*, pp. 943–950, 2000.

Sun, W., Jiang, N., Krishnamurthy, A., Agarwal, A., and Langford, J. Model-based RL in contextual decision processes: PAC bounds and exponential improvements over model-free approaches. In *Conference on Learning Theory (COLT)*, pp. 2898–2933, 2019.

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

Theocharous, G., Wen, Z., Abbasi-Yadkori, Y., and Vlassis, N. Posterior sampling for large scale reinforcement learning. *arXiv preprint arXiv:1711.07979*, 2017.

Tsitsiklis, J. N. and Van Roy, B. Analysis of temporal-diffference learning with function approximation. In *Advances in neural information processing systems*, pp. 1075–1081, 1997.

Wen, Z. and Van Roy, B. Efficient exploration and value function generalization in deterministic systems. In *Advances in Neural Information Processing Systems*, pp. 3021–3029, 2013.

Wen, Z. and Van Roy, B. Efficient reinforcement learning in deterministic systems with value function generalization. *Mathematics of Operations Research*, 42(3): 762–782, 2017.

Xie, C., Patil, S., Moldovan, T., Levine, S., and Abbeel, P. Model-based reinforcement learning with parametrized physical models and optimism-driven exploration. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 504–511. IEEE, 2016.

Yang, L. F. and Wang, M. Reinforcement leaning in feature space: Matrix bandit, kernels, and regret bound. *arXiv preprint arXiv:1905.10389*, 2019a.

Yang, L. F. and Wang, M. Sample-optimal parametric Q-learning with linear transition models. *International Conference on Machine Learning*, 2019b.

Zanette, A., Lazaric, A., Kochenderfer, M. J., and Brunskill, E. Limiting extrapolation in linear approximate value iteration. In *Advances in Neural Information Processing Systems 32*, pp. 5616–5625. Curran Associates, Inc., 2019.

Zhang, Z., Zhou, Y., and Ji, X. Almost optimal model-free reinforcement learning via reference-advantage decomposition. *arXiv preprint arXiv:2004.10019*, April 2020.