Operator Splitting Value Iteration

Amin Rakhsha[1,2] **Andrew Wang**[1,2] **Mohammad Ghavamzadeh**[3]

Amir-massoud Farahmand[2,1]

[1]Department of Computer Science, University of Toronto
[2]Vector Institute [3]Google Research

Abstract

We introduce new planning and reinforcement learning algorithms for discounted MDPs that utilize an approximate model of the environment to accelerate the convergence of the value function. Inspired by the splitting approach in numerical linear algebra, we introduce Operator Splitting Value Iteration (OS-VI) for both Policy Evaluation and Control problems. OS-VI achieves a much faster convergence rate when the model is accurate enough. We also introduce a sample-based version of the algorithm called OS-Dyna. Unlike the traditional Dyna architecture, OS-Dyna still converges to the correct value function in presence of model approximation error.

1 Introduction

Consider a planning problem for a discounted MDP with dynamics $\mathcal{P}$. Suppose that we have access to an approximate model $\hat{\mathcal{P}} \approx \mathcal{P}$ as well. For example, $\mathcal{P}$ might be a high-fidelity, but slow, simulator, and $\hat{\mathcal{P}}$ is a lower-fidelity, but fast, simulator. Or in a different, but relevant, context of model-based reinforcement learning (MBRL), $\mathcal{P}$ is the unknown dynamics of a real-world system, from which we can only acquire expensive samples, and $\hat{\mathcal{P}}$ is a learned model, from which samples can be cheaply acquired. Can we use this approximate model $\hat{\mathcal{P}}$ to *accelerate* the computation of the value function of a policy π (Policy Evaluation (PE) problem) or the optimal value function (Control problem)?

The Value Iteration (VI) algorithm, and its approximate variant, is a fundamental algorithm in Dynamic Programming that can find the value of a policy or the optimal value function. It is also the backbone of many reinforcement learning (RL) algorithms such as Temporal Difference Learning [Sutton, 1988], Fitted Value Iteration [Gordon, 1995, Ernst et al., 2005, Munos and Szepesvári, 2008], and Deep Q Network [Mnih et al., 2015]. This algorithm, however, can be slow when the discount factor is close to 1, as its convergence rate is $O(\gamma^k)$. Moreover, even though we could use VI using $\hat{\mathcal{P}}$ instead of $\mathcal{P}$, effectively avoiding any need for expensive queries to $\mathcal{P}$, the obtained value function would converge to a solution different from the value function of the original MDP.

This paper proposes an algorithm called Operator Splitting Value Iteration (OS-VI) that benefits from an approximate model $\hat{\mathcal{P}}$ to potentially accelerate the convergence of the value function sequence to the value function with respect to (w.r.t.) the true model $\mathcal{P}$ (Section 3). This algorithm is for both PE (Section 3.1) and Control (Section 3.2) problems. The acceleration is not uniform though, and depends on how close $\hat{\mathcal{P}}$ is to $\mathcal{P}$ (Section 4).

A key inspiration behind OS-VI is the (matrix) splitting approach in the numerical linear algebra, which is used for iterative solution of large linear systems of equations [Varga, 2000, Saad, 2003, Golub and Van Loan, 2013]. With a proper choice of splitting, one may change the convergence rate of linear system of equation solvers. We show that the conventional VI for PE can be seen as a particular choice of splitting. This observation suggests that one may choose other forms of splitting in order to change the convergence rate. It turns out that we can choose a splitting that benefits from having access to $\hat{\mathcal{P}}$ (Section 2). The new splitting leads to OS-VI for PE. For the Control problem, the connection between solving linear systems of equations and VI is not as straightforward anymore, as the former is linear, while the latter is not, but we can still get inspired from the splitting approach to design OS-VI for Control. The key step of such an algorithm is a new *policy improvement* step.

The form of the OS-VI algorithm opens up a connection to MBRL where the approximate model $\hat{\mathcal{P}}$ is learned using data. This leads to the OS-Dyna algorithm, inspired by a generic Dyna architecture [Sutton, 1990]. OS-Dyna is a hybrid of model-free and model-based algorithms. It uses the learned model in its inner planning loop, but uses samples from the true model $\mathcal{P}$ in order to correct the effect of errors in the model. Existing MBRL algorithms would converge to an incorrect solution if the approximate model $\hat{\mathcal{P}}$ does not converge to the true model $\mathcal{P}$. This would be the case whenever model approximation error exists. On the other hand, OS-Dyna can still converge to the correct value function even when $\hat{\mathcal{P}}$ does not converge to $\mathcal{P}$. As far as we know, this is the first model-based RL algorithm with such property.

2 From value iteration to splitting-based linear system of equations solvers and back

We briefly describe the VI algorithm and the splitting methods for solving linear systems of equations, and explain their connections. We consider a discounted Markov Decision Process (MDP) $(\mathcal{X}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma)$ [Bertsekas and Tsitsiklis, 1996, Szepesvári, 2010, Sutton and Barto, 2019]. We defer formal definitions to the supplementary material. We only mention that for a policy π, we denote by $\mathcal{P}^\pi$ its transition kernel, by $r^\pi : \mathcal{X} \to \mathbb{R}$ the expected value of its reward distribution, and by $V^\pi = V^\pi(\mathcal{R}, \mathcal{P})$ its state-value function. We also represent the optimal state-value function by $V^* = V^*(\mathcal{R}, \mathcal{P})$ and the optimal policy by $\pi^* = \pi^*(\mathcal{R}, \mathcal{P})$. The Bellman operator $T^\pi : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$ for policy π and the Bellman optimality operator $T^* : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$ are[1]

$$(T^\pi V)(x) \triangleq r^\pi(x) + \gamma \int \mathcal{P}^\pi(\mathrm{d}y|x) V(y); \quad (T^* V)(x) \triangleq \max_{a \in \mathcal{A}} \left\{ r(x,a) + \gamma \int \mathcal{P}(\mathrm{d}y|x,a) V(y) \right\}.$$

These operators can be written more compactly as $T^\pi : V \mapsto r^\pi + \gamma \mathcal{P}^\pi V$ and $T^* : V \mapsto \max_\pi \{r^\pi + \gamma \mathcal{P}^\pi V\}$. The *greedy* policy at state $x \in \mathcal{X}$ is

$$\pi_g(x; V) \leftarrow \operatorname*{argmax}_{a \in \mathcal{A}} \left\{ r(x,a) + \gamma \int \mathcal{P}(\mathrm{d}y|x,a) V(y) \right\},$$

or more compactly, $\pi_g(V) \leftarrow \operatorname{argmax}_\pi T^\pi V$. We have $T^* V = T^{\pi_g(V)} V$, that is, the effect of the Bellman optimality operator T^* applied to a value function V is the same as applying the Bellman operator of the *greedy* policy w.r.t. V to V.

2.1 Value Iteration

The value function V^π and the optimal value function V^* are the fixed points of the operators T^π and T^*, respectively, and satisfy the Bellman equation. For the PE problem, this means that

$$V^\pi = r^\pi + \gamma \mathcal{P}^\pi V^\pi \Rightarrow (\mathbf{I} - \gamma \mathcal{P}^\pi) V^\pi = r^\pi. \tag{2.1}$$

There are several ways to compute the value function of a policy π or the optimal value function V^*, including the iterative methods such as Value Iteration (VI) and Policy Iteration (PI) algorithms, or solving a linear systems of equations (for PE) or linear programming (for Control). We focus on the VI algorithm in this work. VI repeatedly applies the Bellman operator to the most recent approximation of the value function: Given an initial value function V_0, it generates a sequence $(V_k)_{k \geq 0}$ as follows:

$$V_k \leftarrow \begin{cases} T^\pi V_{k-1}, & \text{(Policy Evaluation)} \\ T^* V_{k-1}. & \text{(Control)} \end{cases} \tag{2.2}$$

[1]For countable state and action spaces, the integrals are replaced by summations. We present OS-VI and its theoretical analysis for general state/action spaces, but limit our experiments to finite state/action problems.

VI for Control can be written in an equivalent form: At iteration k, we first compute the greedy policy $\pi_k \leftarrow \pi_g(V_{k-1})$ (policy improvement step), and then $V_k \leftarrow T^{\pi_k} V_{k-1}$. Therefore, the policy improvement step is obtained through finding a policy that is greedy w.r.t. the last value function V_{k-1}, that is, the best policy if we only look one step ahead. This form will be conductive for our later developments. As the Bellman operator is a γ-contraction w.r.t. the supremum norm, the convergence rate of V_k to V^π (or V^*) would be $O(\gamma^k)$. This rate can be slow when γ is very close to 1.

2.2 Matrix splitting for solving linear systems of equations

The VI for PE can be seen as a (matrix) splitting-based iterative method to solve the linear system of equations (2.1). Consider the linear systems of equations $Az = b$, with $A \in \mathbb{R}^{d \times d}$ and $z, b \in \mathbb{R}^d$. Suppose that A is decomposed to $A = M - N$ for some choices of $M, N \in \mathbb{R}^{d \times d}$ (more generally, A, M, and N can be linear operators). Therefore, z satisfies $Mz = Nz + b$. The splitting-based iterative approach defines the new approximation z_k given the current z_{k-1} by solving

$$Mz_k = Nz_{k-1} + b,$$

or equivalently

$$z_k = M^{-1}(Nz_{k-1} + b). \tag{2.3}$$

To analyze the convergence of this iterative method, consider the error $e_k \triangleq z_k - z$. As $Mz = Nz + b$, the dynamics of the error is

$$e_k = M^{-1}Ne_{k-1} = (M^{-1}N)^2 e_{k-2} = \cdots = (M^{-1}N)^k e_0. \tag{2.4}$$

Let $G \triangleq M^{-1}N$. The norm of the sequence $(e_k)_{k \geq 1}$ can be upper bounded as

$$\|e_k\| = \left\|G^k e_0\right\| \leq \left\|G^k\right\| \|e_0\| \leq \|G\|^k \|e_0\|. \tag{2.5}$$

The errors are (norm-) convergent if $\|G\| = \|M^{-1}N\| < 1$, for some choice of norm. More generally, the necessary and sufficient condition for convergence is that the spectral radius $\rho(G)$, the maximum absolute value of eigenvalues of G, is smaller than one, see e.g., Theorem 4.1 of Saad [2003] or Theorem 11.2.1 of Golub and Van Loan [2013].[2] The convergence is faster if the spectral radius (or norm) is closer to zero.

The success of this iterative procedure depends on how we choose M and N such that the norm (or spectral radius) is as small as possible. Also we want to choose an M such that solving $Mz_k = Nz_{k-1} + b$ is not very expensive. For example, if M is an identity matrix $\mathbf{I}$, we get that $N = \mathbf{I} - A$, and the iteration becomes $z_k = (\mathbf{I} - A)z_{k-1} + b$. This iteration is convergent if $\rho(\mathbf{I} - A) < 1$. Other commonly used choices lead to the Jacobi and Gauss-Seidel methods that are described in the supplementary material.

We are now ready to make the connection between splitting-based iterative methods and VI for PE. If we choose $A = \mathbf{I} - \gamma\mathcal{P}^\pi$, we see that equation $AV^\pi = r^\pi$ is indeed the Bellman equation for policy π (2.1). The VI for PE, which is $V_k = \gamma\mathcal{P}^\pi V_{k-1} + r^\pi = (\mathbf{I} - A)V_{k-1} + r^\pi$, corresponds to the choice of $M = \mathbf{I}$ and $N = \gamma\mathcal{P}^\pi$. This brings up the question of whether it is possible to split A to other choices of M and N so that the resulting VI-like procedure has better convergence properties. We suggest a particular choice in the next section.

3 Operator splitting value iteration algorithm

This section introduces the Operator Splitting Value Iteration (OS-VI) algorithm. We start from the PE problem and introduce the Control version based on that.

3.1 OS-VI for policy evaluation

Given a policy π, true model $\mathcal{P}$, and approximate model $\hat{\mathcal{P}}$, we split $\mathbf{I} - \gamma\mathcal{P}^\pi$ to M^π and N^π as

$$M^\pi = \mathbf{I} - \gamma\hat{\mathcal{P}}^\pi \quad , \quad N^\pi = \gamma(\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi).$$

[2]For any matrix norm, we have $\rho(G) \leq \|G\|$, so the condition on the norm is sufficient, but not necessary. In this work, most of our analysis will be norm-based.

Following the recipe of (2.3), the OS-VI algorithm for PE would be

$$V_k \leftarrow (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \Big[r^\pi + \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi) V_{k-1} \Big], \tag{3.1}$$

starting from an initial V_0.[3] To gain more intuition and prepare for further developments, we define a few notations. We define the *Varga operator* $S^\pi : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$, named after Richard S Varga (1928 – 2022) who has made significant contributions to matrix analysis, as the mapping between the space of all bounded functions over $\mathcal{X}$ to the same space as

$$S^\pi : V \mapsto (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \Big[r^\pi + \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi) V \Big].$$

Observe that (3.1) can be compactly written as

$$V_k \leftarrow S^\pi V_{k-1}. \tag{3.2}$$

It is not difficult to see that $S^\pi V^\pi = V^\pi$, i.e., the value function of a policy π is a fixed-point of the Varga operator S^π. This and other properties of the Varga operator are in the supplementary material.

Given any value function V, define an auxiliary reward function $\bar{r}_V : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ as

$$\bar{r}_V(x,a) \triangleq r(x,a) + \gamma \int \Big(\mathcal{P}(\mathrm{d}y|x,a) - \hat{\mathcal{P}}(\mathrm{d}y|x,a) \Big) V(y). \tag{3.3}$$

Similar to the notation for r^π, we define $\bar{r}^\pi_V : \mathcal{X} \to \mathbb{R}$ as $\bar{r}^\pi_V(x) = \bar{r}_V(x, \pi(x))$ for a deterministic policy π (and similarly for a stochastic policy). With this notation,

$$S^\pi V = (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \bar{r}^\pi_V.$$

This is the value function of following π in an MDP with dynamics $\hat{\mathcal{P}}$ and reward $\bar{r}_V$. Therefore, at each iteration of OS-VI (PE), we evaluate the policy π according to the approximate dynamics, and a reward function that consists of the original reward r and the correction term $\gamma(\mathcal{P} - \hat{\mathcal{P}})V_{k-1}$. The computation of this value function is a standard PE problem with the approximate model. For instance, we may use another VI (PE) with dynamics $\hat{\mathcal{P}}$ to solve it: We initialize $U_0 \leftarrow V$, and then for $i \geq 1$, we set $U_i \leftarrow \bar{r}^\pi_V + \gamma \hat{\mathcal{P}}^\pi U_{i-1}$. This converges to $S^\pi V = (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \bar{r}^\pi_V$ with the usual rate of $O(\gamma^i)$. Note that aside from the computation of $\bar{r}^\pi_V$, which requires querying $\mathcal{P}$ in order to compute the $\mathcal{P}^\pi V$ term, this iterative process only uses the approximate model $\hat{\mathcal{P}}$, which is assumed to be cheap to access.

What is the benefit of this OS-VI procedure? If the approximate model $\hat{\mathcal{P}}$ is close to the true dynamics $\mathcal{P}$, this leads to a faster convergence of V_k to V^π, as shall be quantified soon. The faster convergence is in terms of the number of queries to $\mathcal{P}$, which is assumed to be expensive. To see this, consider the hypothetical case that $\hat{\mathcal{P}}$ is exactly the same as $\mathcal{P}$, for example, if the cheap simulator happens to perfectly match the reality. Then, $S^\pi V = (\mathbf{I} - \gamma \mathcal{P}^\pi)^{-1}(r^\pi + 0V) = V^\pi$, and the value function for the original MDP is obtained in one iteration of OS-VI. Of course, we often can only hope for $\hat{\mathcal{P}} \approx \mathcal{P}$. In Section 4, we study the impact of error in $\hat{\mathcal{P}}$ on the convergence rate of OS-VI in more details, and show that the convergence of OS-VI can be much faster than classic algorithms even if $\hat{\mathcal{P}}$ is only a close approximation of $\mathcal{P}$.

3.2 OS-VI for control

The VI for Control can be seen as an iterative procedure that computes the greedy policy $\pi_k \leftarrow \pi_g(V_{k-1}) = \operatorname{argmax}_\pi T^\pi V_{k-1}$ in its policy improvement step, and uses one step of the Bellman operator w.r.t. the obtained policy π_k to compute the new estimate of the value function $V_k \leftarrow T^{\pi_k} V_{k-1}$, as described after (2.2). The OS-VI algorithm for Control follows a similar structure with the difference that **1)** the improved policy is the optimizer of the Varga operator, and not the Bellman operator, and **2)** the new value function is obtained by applying the Varga operator of the newly obtained policy. To be concrete, given a value function V, define the S-improved policy

$$\pi_V(V) \triangleq \operatorname*{argmax}_\pi S^\pi V [= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \bar{r}^\pi_V]. \tag{3.4}$$

[3] Although splitting is originally studied mostly in the context of linear algebra and matrices, we are applying the idea more generally. We are not assuming that the state space $\mathcal{X}$ is finite, and allow it to be more general, such as a subset of $\mathbb{R}^d$. Consequently, M^π, N^π, $\mathcal{P}^\pi$, etc. are operators rather than matrices.

This policy is the *optimal* policy for the auxiliary MDP $(\mathcal{X}, \mathcal{A}, \bar{r}_V, \hat{\mathcal{P}}, \gamma)$. We also define the *Varga optimal operator* $S^* : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$ as

$$S^* : V \mapsto \max_\pi S^\pi V [= \max_\pi (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \bar{r}_V^\pi].$$

The function $S^* V$ is equal to $S^{\pi_V(V)} V$, i.e., the Varga operator of the S-improved policy w.r.t. V applied to a value function V (compare it with $T^* V = T^{\pi_g(V)} V$).

The OS-VI (Control) is then simply

$$V_k \leftarrow S^* V_{k-1}, \tag{3.5}$$

which in its expanded form, consists of the following two steps:

$$\pi_k \leftarrow \pi_V(V_{k-1}), \qquad \text{(policy improvement)}. \tag{3.6}$$

$$V_k \leftarrow S^{\pi_k} V_{k-1} \; [= (\mathbf{I} - \hat{\mathcal{P}}^{\pi_k})^{-1} (r^{\pi_k} + \gamma (\mathcal{P}^{\pi_k} - \hat{\mathcal{P}}^{\pi_k}) V_{k-1})], \text{(partial policy evaluation)}. \tag{3.7}$$

Comparing the S-improved policy (3.4) used in the policy improvement step (3.6) of OS-VI with the conventional greedy policy is insightful. The greedy policy is $\text{argmax}_\pi T^\pi V$. Expanding $T^\pi V$, we see that the greedy policy is the maximizer of $r^\pi + \gamma \mathcal{P}^\pi V$. The function $r^\pi + \gamma \mathcal{P}^\pi V$ is only a single-step bootstrapped estimate of the value of V^π, and its maximizer, the greedy policy, is in general different from the optimal policy, which maximizes V^π. On the other hand, the S-improved policy $\pi_V(V)$ solves a full MDP with an approximate model $\hat{\mathcal{P}}$ and the reward function that has both the original reward r and the correction term $\gamma(\mathcal{P} - \hat{\mathcal{P}})V$. In the special case that $\hat{\mathcal{P}} = \mathcal{P}$, the correction term is zero, and $\pi_V(V)$ would be the optimal policy π^* for the original MDP. As often $\hat{\mathcal{P}} \approx \mathcal{P}$, the value function of policy $\pi_V(V)$ is not exactly the optimal value. The partial policy evaluation step (3.7) updates the value function to a value that is closer to the optimal value function.

Remark. The use of matrix splitting-based ideas, either explicitly or implicitly, in the context of dynamic programming is not completely novel to this work. Kushner and Kleinman [1971] is one of the earliest paper that mentions the Jacobi and Gauss-Seidel procedures for computing the value function. Porteus [1975] proposes several transformations to the reward and probability transition matrix with the goal of improving the computational cost of solving the transformed MDP. One of the transformations, called *pre-inverse transform*, has some similarities with the operator splitting of this work. The end result, however, is different. Bacon and Precup [2016] provide a matrix splitting perspective on planning with options. The connection between multi-step models and matrix splitting is developed in Chapter 4 of Bacon [2018]. Refer to the supplementary material for more discussion.

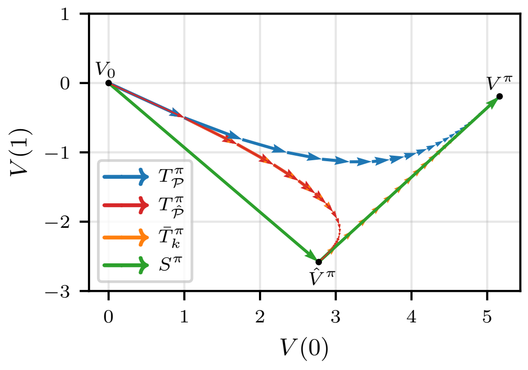
3.3 Visualizing how OS-VI works

We visualize the value function trajectories of several algorithms, including OS-VI, on a 2-state MDP in order to provide some intuition on how OS-VI works. We consider the policy evaluation for the dynamics $\mathcal{P}^\pi = \left[\begin{smallmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{smallmatrix}\right]$ with the reward $r^\pi = \left(\begin{smallmatrix} 1 \\ -0.5 \end{smallmatrix}\right)$ and $\gamma = 0.9$. We consider two approximate models: a relatively accurate one $\hat{\mathcal{P}}^\pi_{\text{accurate}} = \left[\begin{smallmatrix} 0.85 & 0.15 \\ 0.05 & 0.95 \end{smallmatrix}\right]$, and an inaccurate one $\hat{\mathcal{P}}^\pi_{\text{inaccurate}} = \left[\begin{smallmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{smallmatrix}\right]$. The results are shown in Figure 1.

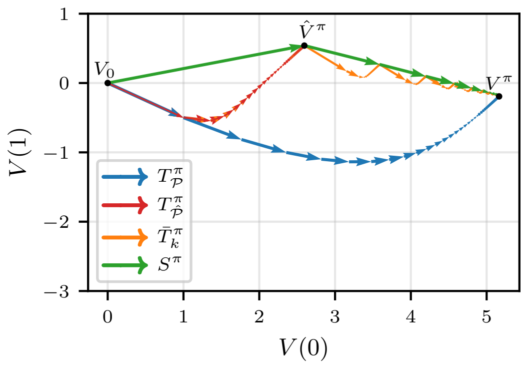
The first algorithm is the conventional VI (PE), which repeatedly applies the Bellman operator according to the true model $\mathcal{P}^\pi$ to the most recent approximation of the value function. We use $T^\pi_{\mathcal{P}}$ to refer to its Bellman operator and to label the corresponding trajectory in the value space. This algorithm converges to the true value function V^π.

The second algorithm is VI (PE) that uses $\hat{\mathcal{P}}^\pi$ as the model. This procedure is the basis of the Dyna architecture. We use $T^\pi_{\hat{\mathcal{P}}}$ to refer to its Bellman operator and to label the corresponding trajectory in the value space. Due to the error of $T^\pi_{\hat{\mathcal{P}}}$ compared to $T^\pi_{\mathcal{P}}$, the algorithm converges to a wrong value function $\hat{V}^\pi$, as both figures show. We observe that even when the model is relatively accurate as in Figure 1a, the converged value function is quite wrong. This illustrates one limitation of the conventional model-based RL algorithms where an inaccurate model may lead to significantly inaccurate estimate of the value function.

The OS-VI algorithm repeatedly applies the Varga operator S^π to the most recent approximation of the value function. As discussed earlier, each computation of $S^\pi V_{k-1}$ corresponds to solving an



(a) More accurate model $\hat{\mathcal{P}}_{\text{accurate}}$



(b) Less accurate model $\hat{\mathcal{P}}_{\text{inaccurate}}$

Figure 1: The value function trajectories of VI (PE) with the true model ($T^\pi_{\mathcal{P}}$), VI (PE) with approximate model $T^\pi_{\hat{\mathcal{P}}}$, and OS-VI (PE) ($S^\pi$; and $\bar{T}_k$ for its inner loop) for a 2-state problem.

auxiliary MDPs $(\mathcal{X}, \mathcal{A}, \bar{r}_{V_{k-1}}, \hat{\mathcal{P}}, \gamma)$. We denote the Bellman operator of this auxiliary MDP by $\bar{T}^\pi_k$. The figures show the trajectory generated by the iterative application of S^π on the most recent value function as well as the trajectory for solving each auxiliary MDP, indicated by $\bar{T}^\pi_k$. We observe that the OS-VI algorithm converges to the correct value function despite using an incorrect model. When the model is more accurate, very few iteration of OS-VI gets a value close to V^π (two iterations in Figure 1a); when the model is less accurate, a few more iterations are needed. Compared to VI, at least in these examples, the total number of iterations of OS-VI is significantly fewer.

When the initial value function is $V_0 = 0$, the result of the first iteration of OS-VI is the same value function computed by the VI with the wrong model $\hat{\mathcal{P}}^\pi$. This is because $V_1 \leftarrow S^\pi V_0 = (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \bar{r}^\pi_{V_0}$ and $\bar{r}^\pi_{V_0} = r^\pi$, so $V_1 = (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} r^\pi$. This is the same solution as the value function obtained using the approximate model $\hat{\mathcal{P}}$. In these figures, this shows itself by the overlapping of the red arrows followed by $T^\pi_{\hat{\mathcal{P}}}$ and the first segment of the orange arrows, which are generated by the repeated application of $\bar{T}^\pi_1$. In further iterations of OS-VI, the auxiliary MDPs change and the path followed by $\bar{T}^\pi_k$ ($k \geq 2$) deviates from the solution of the VI with the wrong model.

4 Convergence analysis of operator splitting value iteration

In this section, we present the convergence analysis of OS-VI. Our results show that OS-VI has an $O(\gamma'^k)$ convergence rate for an effective discount factor γ' that depends on the error between $\hat{\mathcal{P}}$ and $\mathcal{P}$. For small enough error, $\gamma' < \gamma$ and OS-VI has a faster convergence rate compared to the classic VI, Policy Iteration (PI), and Modified Policy Iteration (MPI), which all have $O(\gamma^k)$ behaviour. We provide results for both the L_∞ and L_p norms.

4.1 Convergence of OS-VI for policy evaluation

We study the convergence behaviour of OS-VI (PE) in presence of error in value updates. Specifically, we consider that in each iteration k, the update (3.2) has an error, i.e.,

$$V_k = S^\pi V_{k-1} + \epsilon^{\text{value}}_k \tag{4.1}$$

The error $\epsilon^{\text{value}}_k$ encompasses various sources of errors. One source is the function approximation error due to using a function approximator to represent V_k, which is often required in large state-action spaces. Another is the estimation (i.e., statistical) error caused due to having a finite number of samples, instead of direct access to $\mathcal{P}$, in the RL setting. Refer to Györfi et al. [2002], Steinwart and Christmann [2008] for the discussion of function approximation and estimation errors in the supervised learning context and to Munos and Szepesvári [2008], Antos et al. [2008], Farahmand et al. [2016], Chen and Jiang [2019], Fan et al. [2019] for their analysis in the RL context. In this work, we do not analyze how the number of samples, the function approximator, etc. affect the errors $\epsilon^{\text{value}}_k$. The result of this section provides an error propagation result similar to Munos [2007] (approximate VI), Munos [2003] (approximate PI), and Scherrer et al. [2015] (approximate modified PI).

To study convergence behaviour of OS-VI (PE), let $G^\pi = (\mathbf{I} - \gamma\hat{\mathcal{P}}^\pi)^{-1}\gamma(\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi)$. We use the fact that $S^\pi V^\pi = V^\pi$ and write

$$\begin{aligned}\|V^\pi - V_k\|_\infty &= \left\|S^\pi V^\pi - S^\pi V_{k-1} - \epsilon_k^{\text{value}}\right\|_\infty = \left\|G^\pi(V^\pi - V_{k-1}) - \epsilon_k^{\text{value}}\right\|_\infty \\ &\leq \|G^\pi\|_\infty \|V^\pi - V_{k-1}\|_\infty + \left\|\epsilon_k^{\text{value}}\right\|_\infty. \end{aligned} \tag{4.2}$$

Now, we have that

$$\|G^\pi\|_\infty = \left\|(\mathbf{I} - \gamma\hat{\mathcal{P}}^\pi)^{-1}\gamma(\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi)\right\|_\infty \leq \frac{\gamma}{1-\gamma}\left\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right\|_\infty, \tag{4.3}$$

where we used the fact that for any square matrix F with a matrix norm $\|F\|_p < 1$, it holds that $\|(\mathbf{I} - F)^{-1}\|_p \leq \frac{1}{1-\|F\|_p}$ (e.g., Lemma 2.3.3 of Golub and Van Loan 2013), and that the supremum norm of a stochastic matrix $\hat{\mathcal{P}}^\pi$ is 1. Assuming that $\|\epsilon_k^{\text{value}}\|_\infty \leq \epsilon^{\text{value}}$ for all $k \geq 1$ and defining effective discount factor $\gamma' = \frac{\gamma}{1-\gamma}\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$, the upper bounds (4.3) and (4.2) lead to $\|V^\pi - V_k\|_\infty \leq \gamma'^k \|V^\pi - V_0\|_\infty + \frac{1-\gamma'^k}{1-\gamma'}\epsilon^{\text{value}}$.

A few remarks are in order. First, whenever $\gamma' < \gamma$, this is guaranteed to be faster than the convergence rate of the conventional VI, which is $O(\gamma^k)$. This is the case if $\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty < 1 - \gamma$. If the model is very accurate, we obtain much faster rate than VI's. Since each iteration k corresponds to a query to the true model $\mathcal{P}$, a faster rate entails that the algorithm requires less number of queries to the expensive model to reach the same level of accuracy.

Second, although the model error $\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$ is a reasonable choice to measure the distances between distributions (it is the maximum of the Total Variation distance between $\mathcal{P}^\pi(\cdot|x)$ and $\hat{\mathcal{P}}^\pi(\cdot|x)$ over x, which itself can be upper bounded by their KL divergence; see the supplementary material), it is somewhat conservative as it takes the supremum over the state space. Likewise, requiring $\|\epsilon_k^{\text{value}}\|_\infty$ to be small is also conservative, as approximating $S^\pi V_{k-1}$ using a function approximator given samples (RL setting) often leads to an L_p-norm type of guarantee. We now provide a different analysis to address these issues.

To present the L_p-norm result, we need to define some notations. First, we define the conditional discounted future-state distribution of policy π under $\hat{\mathcal{P}}$ as the following probability distribution: Given a measurable set B, we have $\hat{\eta}^\pi(B|x) = (1-\gamma)\sum_{t=0}^\infty \gamma^t \cdot \mathbb{P}\Big(X_t \in B | X_0 = x, \pi, \hat{\mathcal{P}}\Big)$, where the chain $(X_t)_{t\geq 0}$ starts from state x and evolves by following policy π under transitions $\hat{\mathcal{P}}$. For an arbitrary distribution ρ over the state space, we define the *approximate discounted future-state distribution concentration coefficient* as

$$\hat{C}^\pi(\rho)^2 = \frac{1}{\gamma^2}\int \rho(\mathrm{d}x)\left\|\frac{\mathrm{d}\hat{\eta}^\pi(\cdot|x)}{\mathrm{d}\rho}\right\|_\infty^3. \tag{4.4}$$

Here $\frac{\mathrm{d}\hat{\eta}^\pi(\cdot|x)}{\mathrm{d}\rho}$ is the Radon–Nikodym derivative of the distribution $\hat{\eta}^\pi(\cdot|x)$ w.r.t. the distribution ρ, and it is assumed that for any $x \in \mathcal{X}$, $\hat{\eta}^\pi(\cdot|x) \ll \rho$, i.e., $\hat{\eta}^\pi(\cdot|x)$ is absolutely continuous w.r.t. ρ (otherwise, the coefficient would be set to infinity). This coefficient measures how concentrated the distribution $\hat{\eta}^\pi(\cdot|x)$ is compared to ρ. This is weighted according to the initial state distribution ρ. Similar concentrability coefficients, but not exactly this one, have appeared in the error propagation results [Kakade and Langford, 2002, Munos, 2003, 2007, Farahmand et al., 2010, Scherrer et al., 2015]. Finally, we define the weighted χ^2-divergence of $\hat{\mathcal{P}}^\pi$ and $\mathcal{P}^\pi$ as

$$\chi^2_\rho(\mathcal{P}^\pi \;||\; \hat{\mathcal{P}}^\pi) \triangleq \int \rho(\mathrm{d}x)\chi^2\Big(\mathcal{P}^\pi(\cdot|x) \;||\; \hat{\mathcal{P}}^\pi(\cdot|x)\Big) = \int \rho(\mathrm{d}x)\int \frac{\left|\hat{\mathcal{P}}^\pi(\mathrm{d}y|x) - \mathcal{P}^\pi(\mathrm{d}y|x)\right|^2}{\hat{\mathcal{P}}^\pi(\mathrm{d}y|x)}.$$

This notion of model error is less strict in requiring accurate approximation $\mathcal{P}$ in all states. Usually only a subset of state space is important or even reachable in a problem. The above model error can only focus on specific areas of the state space through the choice of distribution ρ.

We are now ready to to provide the main theorem for the approximate OS-VI (PE).

Theorem 1. *Consider the approximate OS-VI algorithm for PE* (4.1)*. Let $\|\cdot\|_\star$ be either the supremum norm $\|\cdot\|_\infty$ ($\star = \infty$) or $\|\cdot\|_{4,\rho}$ for ρ being an arbitrary distribution over the state space ($\star = 4, \rho$). Assume that $\|\epsilon_k^{value}\|_\star \leq \epsilon^{value}$ for all $k \geq 1$. Furthermore, define the effective discount factor as*

$$\gamma' = \frac{\gamma}{1-\gamma} \begin{cases} \left\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right\|_\infty & (\star = \infty), \\ \sqrt{\hat{C}^\pi(\rho) \cdot \chi^2_\rho(\mathcal{P}^\pi \mid\mid \hat{\mathcal{P}}^\pi)} & (\star = 4, \rho). \end{cases}$$

For any $k \geq 0$, we have $\|V^\pi - V_k\|_\star \leq \gamma'^k \|V^\pi - V_0\|_\star + \frac{1-\gamma'^k}{1-\gamma'} \cdot \epsilon^{value}$.

4.2 Convergence of OS-VI for control

We turn to analyzing OS-VI for Control. We consider two types of errors: The first is the error between the computed value function and the true optimal value function of the auxiliary MDP, i.e., $V_k - S^* V_{k-1}$. The second is the suboptimality of obtained policy compared to the optimal policy of the auxiliary MDP, i.e., $S^{\pi_k} V_{k-1} - S^* V_{k-1}$. Concretely, we have

$$V_k = S^* V_{k-1} + \epsilon_k^{\text{value}}, \tag{4.5}$$

$$S^{\pi_k} V_{k-1} = S^* V_{k-1} + \epsilon_k^{\text{policy}}. \tag{4.6}$$

We have the following result for OS-VI (Control).

Theorem 2. *Consider the approximate OS-VI algorithm for control* (4.5)-(4.6)*. Let $\|\cdot\|_\star$ be either the supremum norm $\|\cdot\|_\infty$ ($\star = \infty$) or $\|\cdot\|_{4,\rho}$ for ρ being an arbitrary distribution over the state space ($\star = 4, \rho$). For any $k \geq 1$, let $\Pi_k = \{\pi^*, \pi_k\} \cup \{\pi_V(V_{i-1}) : 1 \leq i < k\}$. Assume that $\|\epsilon_k^{value}\|_\star \leq \epsilon^{value}$ for all $k \geq 1$. Furthermore, define the effective discount factor as*

$$\gamma' = \frac{\gamma}{1-\gamma} \begin{cases} \max_{\pi \in \Pi_k} \left\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right\|_\infty & (\star = \infty), \\ \max_{\pi \in \Pi_k} \sqrt{\sqrt{2} \cdot \hat{C}^\pi(\rho) \cdot \chi^2_\rho(\mathcal{P}^\pi \mid\mid \hat{\mathcal{P}}^\pi)} & (\star = 4, \rho). \end{cases}$$

We then have

$$\|V^{\pi_k} - V^*\|_\star \leq \frac{2\gamma'^k}{1-\gamma'} \|V_0 - V^*\|_\star + \frac{2\gamma'(1-\gamma'^{k-1})}{(1-\gamma')^2} \cdot \epsilon^{value} + \frac{1}{1-\gamma'} \cdot \left\|\epsilon_k^{policy}\right\|_\star.$$

We can compare this result with the convergence result of VI. For VI with the supremum norm, following the proof of Equation (2.2) by Munos [2007], we can show that $\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma^k}{1-\gamma}\|V^* - V_0\|_\infty + \frac{2\gamma(1-\gamma^{k-1})\varepsilon}{(1-\gamma)^2}$, with $\|V_i - T^* V_{i-1}\|_\infty \leq \varepsilon$ for all $i < k$ (similar result for the L_p-norm also holds, see Theorem 5.2 in Munos 2007). For the approximate VI, the initial error $\|V^* - V_0\|_\infty$ decays with the rate of $O(\gamma^k)$. This should be compared with $O(\gamma'^k)$ rate of OS-VI. The effect of error at each step ϵ^{value} is also similar: approximate VI has $(1-\gamma)^{-2}$ dependence while approximate OS-VI has $(1-\gamma')^{-2}$. What is remarkable is that as opposed to γ, which is a fixed parameter of the problem and can be close to 1, γ' can be made arbitrary close to zero when the model is accurate. The additional information given by $\hat{\mathcal{P}}$ allows us to get much faster rate than VI. Of course, this requires the model to be accurate. An inaccurate model might be detrimental to the convergence rate, and may even lead to divergence. Similar conclusions can be made in comparing OS-VI with Policy Iteration and Modified Policy Iteration, as discussed in the supplementary material.

5 Operator splitting Dyna

In this section, we generalize the operator splitting approach to an RL problem in which only samples from $\mathcal{P}$ are available. In this scenario, the approximate model $\hat{\mathcal{P}}$ may not be given, as assumed in previous sections. However, one can learn such an approximate model using samples from the environment as done in MBRL algorithms. Unlike MBRL approaches that solely rely on the model, our approach takes advantage of both the true environment and the model in its updates. Therefore, one can think of the operator splitting approach as a hybrid of MBRL and model-free RL.

Algorithm 1 OS-Dyna

```
1: Initialize $V_0, \bar{r} = 0$, and the approximate model $\hat{\mathcal{P}}$.
2: for $t = 1, 2, \ldots$ do
3:     Sample $(X_t, A_t, R_t, X'_t)$ from environment.
4:     Update the model $\hat{\mathcal{P}}$ with $(X_t, A_t, R_t, X'_t)$
5:     $\bar{r}(X_t, A_t) \leftarrow \bar{r}(X_t, A_t) + \alpha_t \cdot \Big(R_t + \gamma V_{t-1}(X'_t) - \gamma \mathbb{E}_{X' \sim \hat{\mathcal{P}}(\cdot|X_t,A_t)}[V_{t-1}(X')] - \bar{r}(X_t, A_t)\Big)$
6:     $V_t \leftarrow V^{\pi}(\bar{r}, \hat{\mathcal{P}})$ (For PE)   or   $V_t \leftarrow V^{*}(\bar{r}, \hat{\mathcal{P}})$ , $\pi_t \leftarrow \pi^{*}(\bar{r}, \hat{\mathcal{P}})$ (For Control)
7: end for
```

Using a learned $\hat{\mathcal{P}}$, we can calculate V_k from the auxiliary reward function $\bar{r}_k \triangleq \bar{r}_{V_{k-1}}$ by solving the PE or the control problem in the auxiliary MDP $(\mathcal{X}, \mathcal{A}, \bar{r}_k, \hat{\mathcal{P}})$, as discussed in Section 3. A possible challenge of using a learned model is that in many cases, we can only take samples from it and do not have access to the full matrices in calculations. Fortunately, solving PE or control problems with such a model is a very common practice in RL and is also done in Dyna architecture. Therefore, we do not focus on this procedure.

The ability of calculating V_k from $\bar{r}_k$ lets us see the value function as a function of $\bar{r}_k$. Based on this view, we shift our attention to $\bar{r}_k$. Note that since there is a fixed connection between $\bar{r}_k$ and V_k as the solution of PE or control in $(\mathcal{X}, \mathcal{A}, \bar{r}_k, \hat{\mathcal{P}})$, the convergence of $\bar{r}_k$ and V_k are tied together. Therefore, focusing on updating $\bar{r}_k$ will have the same convergence properties. The following update rules can be written for $\bar{r}_k$ itself. For every (x, a),

$$\bar{r}_k(x,a) = r(x,a) + \gamma\Big(\mathcal{P}(\cdot|x,a) - \hat{\mathcal{P}}(\cdot|x,a)\Big) \cdot V^{\pi}(\bar{r}_{k-1}, \hat{\mathcal{P}}) \qquad \text{(Policy Evaluation)} \qquad (5.1)$$

$$\bar{r}_k(x,a) = r(x,a) + \gamma\Big(\mathcal{P}(\cdot|x,a) - \hat{\mathcal{P}}(\cdot|x,a)\Big) \cdot V^{*}(\bar{r}_{k-1}, \hat{\mathcal{P}}) \qquad \text{(Control)} \qquad (5.2)$$

In OS-Dyna, we maintain a vector (or a function approximation) for $\bar{r}$, and as we will show how, update it by samples. The value function is updated to $V^{\pi}(\bar{r}, \hat{\mathcal{P}})$ (for PE) or $V^{*}(\bar{r}, \hat{\mathcal{P}})$ (for control) with each update of $\bar{r}$. This way, we have the value functions in update rules (5.1) and (5.2). The only challenge is that the above update rules need access to distributions $\mathcal{P}(\cdot|x,a)$ and $\hat{\mathcal{P}}(\cdot|x,a)$ for every (x, a), while we only have samples from these distributions in some (x, a) pairs. Fortunately, this challenge has been tackled in developing sample-based algorithms based on the classic VI:

$$\forall (x,a): \qquad Q_k(x,a) = r(x,a) + \gamma \mathcal{P}(\cdot|x,a) V_{k-1}, \qquad (5.3)$$

where $V_{k-1} = Q_{k-1}(x, \pi(x))$ in PE and $V_{k-1} = \max_{a'} Q_{k-1}(x, a)$ in Control. There are multiple approaches to develop sample-based algorithms based on (5.3) such as Fitted Q-Iteration and Stochastic Approximation (SA). In this paper we use SA to develop OS-Dyna, but we point out that other algorithms and techniques can also be applied to develop other versions of OS-Dyna. The key step in SA is to use samples to find an unbiased estimate of the intended update value. For a step (X_t, A_t, R_t, X'_t) in the true environment, we can have the following estimate $Z = R_t + \gamma V(X'_t) - \gamma \mathbb{E}_{X' \sim \hat{\mathcal{P}}(\cdot|X_t,A_t)}[V(X')]$, where the expectation can also be estimated by samples from $\hat{\mathcal{P}}(\cdot|X_t, A_t)$. Finally, we make the following update to $\bar{r}$ with learning rate α_t

$$\bar{r}(X_t, A_t) \leftarrow \bar{r}(X_t, A_t) + \alpha_t \cdot (Z - \bar{r}(X_t, A_t)). \qquad (5.4)$$

The final procedure of OS-Dyna is presented in Algorithm 1.

6 Experiments

We evaluate both OS-VI and OS-Dyna in a finite MDP, comparing our algorithms with existing methods. The MDP considered is a modified cliffwalk environment in a 6×6 grid with 4 actions (up, down, left and right). Full details of the environment is available in the supplementary material. Our convergence analysis shows that the convergence rates of our algorithms depend on the accuracy of $\hat{\mathcal{P}}$. To test OS-VI and OS-Dyna with models of different accuracies, we introduce the smoothed model $\hat{\mathcal{P}}$ of transitions $\mathcal{P}$ with smoothing parameter λ as

$$\hat{\mathcal{P}}(\cdot|x,a;\mathcal{P},\lambda) = (1-\lambda)\mathcal{P}(\cdot|x,a) + \lambda U\big(\{x'|\mathcal{P}(x'|x,a) > 0\}\big), \qquad (6.1)$$

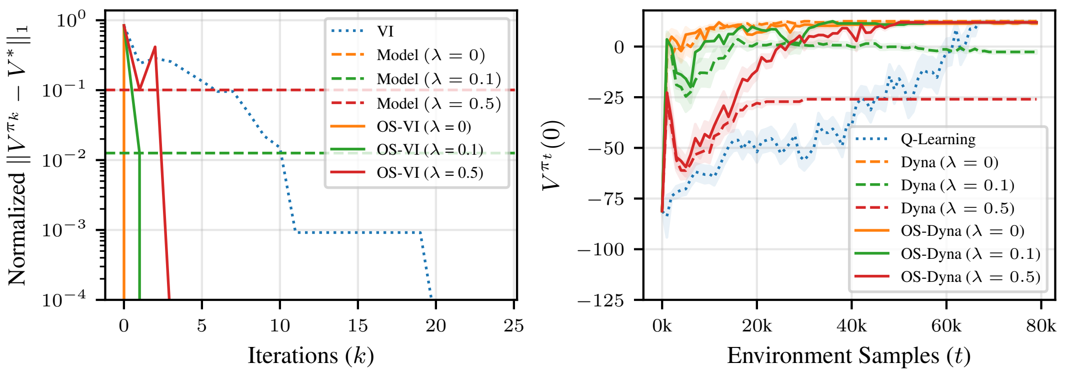


Figure 2: *(Left)* Normalized error comparisons of OS-VI, VI, and the optimal policy of model $\hat{\mathcal{P}}$. *(Right)* Comparison of OS-Dyna with Dyna and Q-Learning in the RL setting.

where $U(A)$ for some set A is the uniform distribution over A. Here, λ allows making adjustments to the amount of error introduced in $\hat{\mathcal{P}}$ w.r.t. $\mathcal{P}$. If $\lambda = 0$, $\hat{\mathcal{P}} = \mathcal{P}$ will be the accurate model, and if $\lambda = 1$, $\hat{\mathcal{P}}$ will be uniform over possible next states in $\mathcal{P}$.

We compare OS-VI and OS-Dyna for Control with other existing methods. The results for the PE problem are qualitatively similar and are provided in the supplementary material along with more experiments. The left plot in Figure 2 shows the convergence of OS-VI compared to VI and the solutions the model itself would lead to. The plot shows normalized error of V^{π_k} w.r.t V^*, i.e. $\|V^{\pi_k} - V^*\|_1 / \|V^*\|_1$. It can be seen that OS-VI has faster convergence with more accurate models and introduces acceleration compared to VI across different model errors. Note that the convergence of OS-VI has been achieved despite the error in the model. The dashed lines show how a fully model-based algorithm would obtain a suboptimal solution by only relying on the model.

We also compare OS-Dyna with with Dyna and Q-Learning in the RL setting. At each iteration t, the algorithms are given a sample (X_t, A_t, R_t, X'_t) where X_t, A_t are selected uniformly at random. For OS-Dyna and Dyna we use the smoothed Maximum-likelihood Estimation (MLE) model. If $\mathcal{P}_{\text{MLE}}$ is the current MLE estimation of the environment transitions, OS-Dyna and Dyna use $\hat{\mathcal{P}}(\mathcal{P}_{\text{MLE}}, \lambda)$ defined in (6.1) as their models. The learning rates are constant α for iterations $t \leq N$ and then decay in the form of $\alpha_t = \alpha/(t - N)$ afterwards. We have fine-tuned the learning rate schedule for *each* algorithm separately for the best results.

The right plot in Figure 2 shows the results for the RL setting. We evaluate the expected return of the policy at iteration t in the initial state of the environment, i.e. $V^{\pi_t}(0)$. Again, OS-Dyna has converged to the optimal policy much faster than Q-Learning. Unlike OS-Dyna, Dyna has failed to find the optimal policy in presence of model error. The results show that OS-Dyna can effectively converge faster than Q-Learning without introducing bias to the final solution due to model error.

7 Conclusion

This paper introduced the Operator Splitting Value Iteration (OS-VI) algorithm, which can benefit from an approximate model $\hat{\mathcal{P}} \approx \mathcal{P}$ to accelerate the convergence of the approximate value to the true value function in terms of the number of queries to the true model $\mathcal{P}$. With a small model error, its convergence rate is exponentially faster compared to well-known dynamical programming algorithms such as Value Iteration and Policy Iteration. We also proposed OS-Dyna as a hybrid model-based/model-free algorithm that can bring in the benefits of a model-based RL algorithm without converging to a biased solution, as Dyna or any other purely model-based RL algorithm does. This paper opens up several future directions. Empirically studying the algorithms on problems with large state spaces, for which a function approximator such as a DNN is required, is an obvious one. This is postponed to a future work as our aim was to build the mathematical foundation and conducting experiments without worrying about challenges such as the optimization of a DNN. There are other algorithmic and theoretical directions to be pursued. One is exploring the space of splittings of $\mathbf{I} - \gamma\mathcal{P}^\pi$. The other is whether we can design Operator Splitting variants of other DP algorithms such as Policy Iteration and Modified Policy Iteration, and study their convergence behaviour.

Acknowledgments and Disclosure of Funding

AMF acknowledges the funding from the Canada CIFAR AI Chairs program, as well as the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) through the Discovery Grant program. AR was partially supported by Borealis AI through the Borealis AI Global Fellowship Award. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

A List of appendices

- Section B includes the definitions and the background.
- Section C includes more details OS-Dyna
- Section D includes some basic properties of the Varga operator and the proofs of the theoretical results.
- Section E includes the additional experiments.
- Section F includes the extended related work and other common splitting choices.

B Background

B.1 Markov Decision Processes

We consider a discounted Markov Decision Process (MDP) $(\mathcal{X}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma)$ [Szepesvári, 2010, Bertsekas and Shreve, 1978, Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 2019]. Our notation is more similar to Szepesvári [2010]. Here $\mathcal{X}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{R} : \mathcal{X} \times \mathcal{A} \to \mathcal{M}(\mathbb{R})$ is the reward distribution, $\mathcal{P} : \mathcal{X} \times \mathcal{A} \to \mathcal{M}(\mathcal{X})$ is the transition probability kernel, and $0 \leq \gamma < 1$ is the discount factor.[4]

The policy $\pi : \mathcal{X} \to \mathcal{M}(\mathcal{A})$ (stochastic policy) or $\pi : \mathcal{X} \to \mathcal{A}$ (deterministic) is a Markov stationary policy. Given a policy, we can define $\mathcal{P}^\pi$ as the transition probability kernel of following π, and it would be

$$\mathcal{P}^\pi(\cdot|x) = \int \mathcal{P}(\cdot|x,a)\pi(\mathrm{d}a|x).$$

We can define $\mathcal{P}^{\pi(m)} : \mathcal{X} \to \mathcal{M}(\mathcal{X})$ for $m \geq 0$ recursively. For $m = 0$, we use the convention that it is equal to $\mathbf{I}$, the identity operator (or matrix). For $m \geq 1$, we have

$$\mathcal{P}^{\pi(m)}(\cdot|x) = \int \mathcal{P}^\pi(\mathrm{d}y|x)\mathcal{P}^{\pi(m-1)}(\cdot|y).$$

The discounted future-state distribution η^π is defined based on m-step transition as

$$\eta^\pi(\cdot|x) = (1-\gamma)\sum_{m=0}^{\infty} \gamma^m \mathcal{P}^{\pi(m)}(\cdot|x).$$

We can define $\mathcal{R}^\pi : \mathcal{X} \to \mathcal{M}(\mathbb{R})$ in a similar fashion. The functions $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ and $r^\pi : \mathcal{X} \to \mathbb{R}$ are the expected value of the reward distribution.

We use V^π and Q^π to denote the state-value and action-value functions for a policy π. We use V^* and Q^* to denote the optimal value and action-value functions. In this work, we mostly use the state-value function, which we simply call the value function.

The Bellman operator $T^\pi : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$ for policy π and the Bellman optimality operator $T^* : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$ are defined as

$$(T^\pi V)(x) \triangleq r^\pi(x) + \gamma \int \mathcal{P}^\pi(\mathrm{d}y|x)V(y),$$

$$(T^* V)(x) \triangleq \max_{a\in\mathcal{A}}\left\{ r(x,a) + \gamma \int \mathcal{P}(\mathrm{d}y|x,a)V(y) \right\}.$$

For countable state and action spaces, the integrals are replaced by summations. The Bellman operators $T^\pi : \mathcal{B}(\mathcal{X} \times \mathcal{A}) \to \mathcal{B}(\mathcal{X} \times \mathcal{A})$ and $T^* : \mathcal{B}(\mathcal{X} \times \mathcal{A}) \to \mathcal{B}(\mathcal{X} \times \mathcal{A})$ (both applied on the action-value function) are defined similarly. We do not use them in the paper, so we do not explicitly define them here.

We denote $\pi(\cdot; V)$ as the greedy policy w.r.t. V, i.e., at each state x, we have

$$\pi_g(x; V) \leftarrow \operatorname*{argmax}_{a\in\mathcal{A}}\left\{ r(x,a) + \gamma \int \mathcal{P}(\mathrm{d}y|x,a)V(y) \right\}.$$

[4]For a set Ω, the space of bounded functions is denoted by $\mathcal{B}(\Omega)$, and the space of probability distributions is denoted by $\mathcal{M}(\Omega)$. Here we do not go into the measurability issues, so we omit the detail of the necessary σ-algebra.

B.2 Norms and metrics

For function $f : \mathcal{X} \to \mathbb{R}$, the $L_p(\rho)$-norm with respect to distribution $\rho \in \mathcal{M}(\mathcal{X})$ over set $\mathcal{X}$ is defined as

$$\|f\|_{p,\rho}^p \triangleq \int f(x)^p \rho(\mathrm{d}x).$$

If ρ is the uniform distribution, we may drop it for simplicity. In the special case of $p = \infty$, we have

$$\|f\|_\infty \triangleq \sup_x |f(x)|.$$

For two distributions $p, q \in \mathcal{M}(\mathcal{X})$, the χ^2-divergence is defined as

$$\chi^2(p \,||\, q) \triangleq \int \frac{(p(\mathrm{d}x) - q(\mathrm{d}x))^2}{q(\mathrm{d}x)}$$

B.3 Other examples of matrix splitting

In addition to the example of $M = \mathbf{I}$ and $N = \mathbf{I} - A$ in Section 2.2, there are several other commonly used choices for matrix splitting.

If we decompose A by its diagonal part D, its strictly lower triangular part $-L$, and its strictly upper triangular part $-U$ (so $A = D - L - U$), the choice of $M = D$ and $N = L + U$ leads to the Jacobi iteration. Clearly, the computation of $M^{-1} = D^{-1}$ is easy.

If we select $M = D - L$ and $N = U$ (or $M = D - U$ and $N = L$), we get the forward (or backward) Gauss-Seidel iteration.

In all these cases, solving $Mz_k = Nz_{k-1} + b$ is easy. The convergence of these methods can be established too. For instance, if A is strictly diagonally dominated, the Jacobi iteration is convergent (Theorem 11.2.2 of Golub and Van Loan [2013]). These examples, as well as other choices available in the numerical linear algebra literature such as the Successive Over Relaxation method, show that there are multiple ways to split A to M and N, each with their own convergence properties.

B.4 Relation between $\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$, the Total Variation error, and the KL divergence

The model error $\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$ appeared in Section 4, and we argued that it is a reasonable choice to measure the distances between distributions. We expand on it here.

For countable state spaces, the norm $\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$ is the maximum over states of the Total Variation (TV) distance between $\mathcal{P}^\pi(\cdot|x)$ and $\hat{\mathcal{P}}^\pi(\cdot|x)$. The TV distance itself can be upper bounded by the KL-divergence between the distributions. So we get

$$\left\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right\|_\infty = \max_{x\in\mathcal{X}} \left\|\mathcal{P}^\pi(\cdot|x) - \hat{\mathcal{P}}^\pi(\cdot|x)\right\|_1 \le \max_{x\in\mathcal{X}} \sqrt{2\mathsf{KL}\left(\mathcal{P}^\pi(\cdot|x)||\hat{\mathcal{P}}^\pi(\cdot|x)\right)}.$$

The KL-divergence is the population version of the negative-logarithm loss used in the Maximum Likelihood Estimation (MLE). Admittedly, the maximum over $\mathcal{X}$ in $\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$ is perhaps strict. Usually there are states in the state space that are irrelevant to the problem or even unreachable. If the model is learned from samples, it will be very inaccurate in these states.

C More details on OS-Dyna

This appendix offers more detail on how Algorithm 1 can be implemented for the special case of finite MDPs using a Maximum-Likelihood estimate model learning.

Here, we provide a more detailed version of Algorithm 1 for finite MDPs that uses MLE for model learning. The expanded version is given in Algorithm 2. It is important to note that we are only presenting a special case of OS-Dyna's implementation. For example, the state-action pairs we observe may come from the trajectories of the agent's interaction with the environment instead of the fixed distribution ρ in Algorithm 2. Also many other techniques in RL such as Fitted Q-Iteration with a replay buffer can be used instead of the stochastic approximation method we used in line 10 of Algorithm 2.

Algorithm 2 uses Maximum-Likelihood Estimation (MLE) for model learning. In finite MDPs, where $\hat{\mathcal{P}}(\cdot|x,a)$ is a $|\mathcal{X}|$-dimensional vector of probabilities for every x, a, MLE takes the form

$$\hat{\mathcal{P}}(x'|x,a) = \frac{N(x,a,x')}{\sum_{x''} N(x,a,x'')}.$$

Here, $N(x,a,x')$ is the number of times x' is reached from x and action a. This gives the model update in Algorithm 2. In the general case, $\hat{\mathcal{P}}$ may be represented by some parametric distribution p_θ. Nonetheless, the same principle of MLE can be applied to define the loss function for model learning. For example by moving towards the gradient of log-likelihood

$$\hat{\theta} \leftarrow \hat{\theta} + \alpha \nabla_{\hat{\theta}} \sum_{i=1}^{t} \log p_\theta(X_i'|X_i, A_i).$$

In addition to using the conventional approach of using MLE for model learning, more recent research has studied decision-aware model learning, in which the loss function incorporates some aspects of the decision problem itself. Some examples are Joseph et al. [2013], Farahmand et al. [2017], Silver et al. [2017], Oh et al. [2017], Farahmand [2018], Grimm et al. [2020], Schrittwieser et al. [2020], D'Oro et al. [2020], Abachi et al. [2020], Lambert et al. [2020], Ayoub et al. [2020], Nikishin et al. [2022], Voelcker et al. [2022].

D Proofs and other theoretical results

D.1 Basic properties of the Varga operators S^π and S^* and MDPs

Lemma 3. *For any policy π, we have*

$$S^\pi V^\pi = V^\pi.$$

Also for the optimal value function V^* and the optimal policy π^*, we have

$$S^* V^* = S^{\pi^*} V^* = V^*.$$

Proof. For the first part we write

$$\begin{aligned}
S^\pi V^\pi &= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1}(r^\pi + \gamma(\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi)V^\pi) \\
&= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1}(r^\pi + \gamma \mathcal{P}^\pi V^\pi - \gamma \hat{\mathcal{P}}^\pi V^\pi) \\
&= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1}(V^\pi - \gamma \hat{\mathcal{P}}^\pi V^\pi) \\
&= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1}(\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi) V^\pi \\
&= V^\pi,
\end{aligned}$$

where we used the Bellman equation $V^\pi = r^\pi + \gamma \mathcal{P}^\pi V^\pi$ in the third equality.

For the second part, note that the second equality is a consequence of the first part and $V^* = V^{\pi^*}$. For the first equality, we prove $S^* V^* = V^*$ by showing that V^* is the optimal value function in MDP $(\mathcal{X}, \mathcal{A}, \bar{r}_{V^*}, \hat{\mathcal{P}})$. To see this, we show that it satisfies the optimal Bellman equations for this MDP. For any state x,

$$\begin{aligned}
\max_a \bar{r}_{V^*}(x,a) + \gamma \hat{\mathcal{P}}(\cdot|x,a)V^* &= \max_a r(x,a) + \gamma \mathcal{P}(\cdot|x,a)V^* - \gamma \hat{\mathcal{P}}(\cdot|x,a)V^* + \gamma \hat{\mathcal{P}}(\cdot|x,a)V^* \\
&= \max_a r(x,a) + \gamma \mathcal{P}(\cdot|x,a)V^* \\
&= V^*(x),
\end{aligned}$$

where in the last step we used the optimal Bellman equation in original MDP. □

Algorithm 2 OS-Dyna (Detailed Version)

```
1: Input: Sampling distribution $\rho$ over $\mathcal{X} \times \mathcal{A}$. Learning rate schedule $(\alpha_t)_{t=0}^{\infty}$. Policy $\pi$ (for PE).
   Inner loop iterations $L$. Environment steps $T$.
2: Initialize
     • Value function $V \colon \mathcal{X} \to \mathbb{R}$ with $V(x) = 0$ for all $x$,
     • Auxiliary reward function $\bar{r} \colon \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ with $\bar{r}(x,a) = 0$ for all $x, a$,
     • Visitation counts $N \colon \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \mathbb{Z}$ with $N(x,a,x') = 0$ for all $x, a, x'$.
     • Transition Model $\hat{\mathcal{P}} \colon \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to [0,1]$ with $\hat{\mathcal{P}}(x'|x,a) = \frac{1}{|\mathcal{X}|}$ for all $x, a, x'$.
3: for $t = 1, 2, \ldots, T$ do
4:     Sample $(X_t, A_t)$ from sampling distribution $\rho$.
5:     Take action $A_t$ at $X_t$ in the environment
6:     Observe $X'_t \sim \mathcal{P}(\cdot|X_t, A_t)$ and $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$
7:     Set $N(X_t, A_t, X'_t) \leftarrow N(X_t, A_t, X'_t) + 1$
8:     Update the model $\forall x' \in \mathcal{X}$ :   $\hat{\mathcal{P}}(x'|X_t, A_t) \leftarrow \frac{N(X_t, A_t, x)}{\sum_{x'' \in \mathcal{X}} N(X_t, A_t, x'')}$
9:     Let
           $$Y_t = R_t + \gamma V(X'_t) - \gamma \sum_{x' \in \mathcal{X}} \hat{\mathcal{P}}(x'|X_t, A_t) \cdot V(x')$$
10:    Set $\bar{r}(X_t, A_t) \leftarrow \bar{r}(X_t, A_t) + \alpha_t \cdot (Y_t - \bar{r}(X_t, A_t))$
11:    Set $U_0 \leftarrow V$
12:    for $i = 1, 2, \ldots, L$ do
13:        For every $x \in \mathcal{X}$, set
           $$U_i(x) \leftarrow \begin{cases} \bar{r}(x, \pi(x)) + \gamma \sum_{x' \in \mathcal{X}} \hat{\mathcal{P}}(x'|x, \pi(x)) \cdot U_{i-1}(x') & \text{(for PE)} \\ \max_{a \in \mathcal{A}} \Big[ \bar{r}(x,a) + \gamma \sum_{x' \in \mathcal{X}} \hat{\mathcal{P}}(x'|x,a) \cdot U_{i-1}(x') \Big] & \text{(for Control)} \end{cases}$$
           $$\hat{\pi}^*(x) \leftarrow \operatorname*{argmax}_{a \in \mathcal{A}} \left[ \bar{r}(x,a) + \gamma \sum_{x' \in \mathcal{X}} \hat{\mathcal{P}}(x'|x,a) \cdot U_{i-1}(x') \right] \qquad \text{(for Control)}$$
14:    end for
15:    Set $V \leftarrow U_L$.
16: end for
17: For PE, output $V$. For control, output $V$ and $\hat{\pi}^*$.
```

Lemma 4. *Let $G^\pi = (M^\pi)^{-1} N^\pi = (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \gamma (\mathcal{P} - \hat{\mathcal{P}})$. For any two functions V_1 and V_2 and policy π we have*

$$S^\pi V_1 - S^\pi V_2 = G^\pi (V_1 - V_2)$$

Proof. We have

$$\begin{aligned} S^\pi V_1 - S^\pi V_2 &= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} (r^\pi + \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi) V_1) - (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} (r^\pi + \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi) V_2) \\ &= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} (r^\pi + \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi) V_1 - r^\pi - \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi) V_2) \\ &= (\mathbf{I} - \gamma \hat{\mathcal{P}}^\pi)^{-1} \gamma (\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi)(V_1 - V_2) \\ &= G^\pi (V_1 - V_2) \end{aligned}$$

□

Lemma 5. *For any value function V and policy π, we have $S^* V \succcurlyeq S^\pi V$ where $\succcurlyeq$ is componentwise inequality.*

Proof. This is direct consequence of definition $S^* V = \max_\pi S^\pi V$ □

Lemma 6. *For any policy π, initial state $x \in \mathcal{X}$, and a measurable set B, we have that*

$$(M^\pi)^{-1}(B|x) = \frac{1}{1-\gamma} \hat{\eta}^\pi (B|x).$$

Proof. Recall that $(M^\pi)^{-1} = (\mathbf{I} - \gamma\hat{\mathcal{P}}^\pi)^{-1}$. As $\|\gamma\hat{\mathcal{P}}^\pi\|_\infty = \gamma < 1$, we can use Neumann expansion

$$(\mathbf{I} - \gamma\hat{\mathcal{P}}^\pi)^{-1} = \sum_{m\geq 0}(\gamma\hat{\mathcal{P}}^\pi)^{(m)}.$$

Therefore, the probability of starting from state x and reaching a measurable set B is $\sum_{m\geq 0}(\gamma\hat{\mathcal{P}}^\pi)^{(m)}(B|x)$, which is $\frac{1}{1-\gamma}\hat{\eta}^\pi(B|x)$ by the definition of $\hat{\eta}^\pi$. □

Lemma 7. *For any policy π, initial state $x \in \mathcal{X}$, and a measurable set B, we have*

$$\int_y \hat{\eta}^\pi(\mathrm{d}y|x)\hat{\mathcal{P}}^\pi(B|y) \leq \frac{1}{\gamma}\hat{\eta}^\pi(B|x).$$

Proof.

$$\begin{aligned}
\int_y \hat{\eta}^\pi(\mathrm{d}y|x)\hat{\mathcal{P}}^\pi(B|y) &= (1-\gamma)\int_y \left(\sum_{t=0}^{\infty}\gamma^t \hat{\mathcal{P}^\pi}^{(t)}(\mathrm{d}y|x)\right)\hat{\mathcal{P}}^\pi(B|y) \\
&= (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \int_y \hat{\mathcal{P}^\pi}^{(t)}(\mathrm{d}y|x)\hat{\mathcal{P}}^\pi(B|y) \\
&= (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \hat{\mathcal{P}^\pi}^{(t+1)}(B|x) \\
&= \frac{(1-\gamma)}{\gamma}\sum_{t=0}^{\infty}\gamma^{t+1}\hat{\mathcal{P}^\pi}^{(t+1)}(B|x) \\
&= \frac{1}{\gamma}(\hat{\eta}^\pi(B|x) - (1-\gamma)\hat{\mathcal{P}^\pi}^{(0)}(B|x)) \\
&\leq \frac{1}{\gamma}\hat{\eta}^\pi(B|x).
\end{aligned}$$

□

D.2 Proofs for convergence of OS-VI for policy evaluation

Proof of Theorem 1 for $\star = \infty$

Proof. From Lemma 3, we have $S^\pi V^\pi = V^\pi$. By definition $V_k = S^\pi V_{k-1} + \epsilon_k^{\text{value}}$. Using Lemma 4, we get

$$\begin{aligned}
\|V^\pi - V_k\|_\infty &= \left\|S^\pi V^\pi - S^\pi V_{k-1} - \epsilon_k^{\text{value}}\right\|_\infty \\
&= \left\|G^\pi(V^\pi - V_{k-1}) - \epsilon_k^{\text{value}}\right\|_\infty \\
&\leq \|G^\pi\|_\infty \|V^\pi - V_{k-1}\|_\infty + \left\|\epsilon_k^{\text{value}}\right\|_\infty
\end{aligned}$$

Now, we have that

$$\|G^\pi\|_\infty = \left\|(\mathbf{I} - \gamma\hat{\mathcal{P}}^\pi)^{-1}\gamma(\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi)\right\|_\infty \leq \frac{\gamma}{1-\gamma}\left\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right\|_\infty,$$

where we used the fact that for a linear operator F with $\|F\| < 1$, it holds that $\left\|(\mathbf{I} - F)^{-1}\right\| \leq \frac{1}{1-\|F\|}$ (when F is a square matrix, this is Lemma 2.3.3 of Golub and Van Loan 2013). As $\hat{\mathcal{P}}^\pi$ is a transition matrix, we can choose the supremum norm, which has the property that $\|\hat{\mathcal{P}}^\pi\|_\infty = 1$.

By defining $\gamma' = \frac{\gamma}{1-\gamma}\left\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right\|_\infty$ and combining the above two inequalities, we get

$$\|V^\pi - V_k\|_\infty \leq \gamma' \cdot \|V^\pi - V_{k-1}\|_\infty + \left\|\epsilon_k^{\text{value}}\right\|_\infty.$$

Expanding this recursive inequality gives

$$
\begin{aligned}
\|V^\pi - V_k\|_\infty &\le \gamma'^k \cdot \|V^\pi - V_0\|_\infty + \sum_{i=1}^{k} \gamma'^{k-i} \|\epsilon_i^{\text{value}}\|_\infty \\
&\le \gamma'^k \cdot \|V^\pi - V_0\|_\infty + \epsilon^{\text{value}} \sum_{i=1}^{k} \gamma'^{k-i} \\
&= \gamma'^k \cdot \|V^\pi - V_0\|_\infty + \frac{1-\gamma'^k}{1-\gamma'} \cdot \epsilon^{\text{value}},
\end{aligned}
$$

which completes the proof. □

Before proving the L_p norm result, we present a key lemma.

Lemma 8. *Let ρ be an arbitrary distribution over state space. Assume that for any $x \in \mathcal{X}$, $\hat{\eta}^\pi(\cdot|x) \ll \rho$, i.e., $\hat{\eta}^\pi(\cdot|x)$ is absolutely continuous w.r.t. ρ. For any policy π and a function $v\colon \mathcal{X} \to \mathbb{R}$, we have*

$$
\|G^\pi v\|_{4,\rho} \le \frac{\gamma}{1-\gamma} \sqrt{\hat{C}^\pi(\rho) \cdot \chi^2_\rho(\mathcal{P}^\pi \,||\, \hat{\mathcal{P}}^\pi)} \cdot \|v\|_{4,\rho}
$$

Proof. Let $\Delta\mathcal{P}^\pi = \left|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\right|$. Using Lemma 6, we expand $\|G^\pi v\|^4_{4,\rho}$

$$
\begin{aligned}
\|G^\pi v\|^4_{4,\rho} &= \int_x \rho(\mathrm{d}x) \left[\iint_{y,z} \frac{1}{1-\gamma} \hat{\eta}^\pi(\mathrm{d}y|x) \cdot \gamma(\mathcal{P}^\pi(\mathrm{d}z|y) - \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)) \cdot v(z) \right]^4 \\
&\le \int_x \rho(\mathrm{d}x) \left[\iint_{y,z} \frac{1}{1-\gamma} \hat{\eta}^\pi(\mathrm{d}y|x) \cdot \gamma \Delta\mathcal{P}^\pi(\mathrm{d}z|y) \cdot |v(z)| \right]^4 \\
&= \frac{\gamma^4}{(1-\gamma)^4} \int_x \rho(\mathrm{d}x) \left[\iint_{y,z} \left(\sqrt{\rho(\mathrm{d}y)} \cdot \frac{\Delta\mathcal{P}^\pi(\mathrm{d}z|y)}{\sqrt{\hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}} \right) \left(\frac{\hat{\eta}^\pi(\mathrm{d}y|x) \cdot |v(z)| \cdot \sqrt{\hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}}{\sqrt{\rho(\mathrm{d}y)}} \right) \right]^4 \\
&\le \frac{\gamma^4}{(1-\gamma)^4} \int_x \rho(\mathrm{d}x) \left(\iint_{y,z} \rho(\mathrm{d}y) \cdot \frac{\Delta\mathcal{P}^\pi(\mathrm{d}z|y)^2}{\hat{\mathcal{P}}^\pi(\mathrm{d}z|y)} \right)^2 \cdot \left(\iint_{y,z} \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot v(z)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\rho(\mathrm{d}y)} \right)^2 \\
&= \frac{\gamma^4}{(1-\gamma)^4} \cdot \chi^2_\rho(\mathcal{P}^\pi \,||\, \hat{\mathcal{P}}^\pi)^2 \cdot \int_x \rho(\mathrm{d}x) \left(\iint_{y,z} \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot v(z)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\rho(\mathrm{d}y)} \right)^2 \\
&= \frac{\gamma^4}{(1-\gamma)^4} \cdot \chi^2_\rho(\mathcal{P}^\pi \,||\, \hat{\mathcal{P}}^\pi)^2 \cdot \int_x \rho(\mathrm{d}x) \left[\int_z \left(\sqrt{\rho(\mathrm{d}z)} \cdot v(z)^2 \right) \cdot \left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\sqrt{\rho(\mathrm{d}z)} \cdot \rho(\mathrm{d}y)} \right) \right]^2 \\
&\le \frac{\gamma^4}{(1-\gamma)^4} \cdot \chi^2_\rho(\mathcal{P}^\pi \,||\, \hat{\mathcal{P}}^\pi)^2 \cdot \int_x \rho(\mathrm{d}x) \left[\int_z \rho(\mathrm{d}z) v(z)^4 \right] \left[\int_z \left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\sqrt{\rho(\mathrm{d}z)} \cdot \rho(\mathrm{d}y)} \right)^2 \right] \\
&= \frac{\gamma^4}{(1-\gamma)^4} \cdot \chi^2_\rho(\mathcal{P}^\pi \,||\, \hat{\mathcal{P}}^\pi)^2 \cdot \|v\|^4_{4,\rho} \cdot \iint_{x,z} \rho(\mathrm{d}x) \left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\sqrt{\rho(\mathrm{d}z)} \cdot \rho(\mathrm{d}y)} \right)^2,
\end{aligned}
\tag{D.1}
$$

where the second and the third inequalities are from the Cauchy-Schwarz inequality. We now write

$$
\begin{aligned}
&\iint_{x,z} \rho(\mathrm{d}x)\left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\sqrt{\rho(\mathrm{d}z)} \cdot \rho(\mathrm{d}y)}\right)^2 \\
&\quad = \int_x \rho(\mathrm{d}x) \int_z \rho(\mathrm{d}z)\left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\rho(\mathrm{d}z) \cdot \rho(\mathrm{d}y)}\right)^2 \\
&\quad = \int_x \rho(\mathrm{d}x) \int_z \rho(\mathrm{d}z)\left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)}{\rho(\mathrm{d}y)} \cdot \frac{\hat{\eta}^\pi(\mathrm{d}y|x) \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\rho(\mathrm{d}z)}\right)^2 \\
&\quad \le \int_x \rho(\mathrm{d}x)\left(\max_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)}{\rho(\mathrm{d}y)}\right)^2 \int_z \rho(\mathrm{d}z) \cdot \left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x) \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\rho(\mathrm{d}z)}\right)^2.
\end{aligned}
$$

Using Lemma 7, we can continue as

$$
\begin{aligned}
&\iint_{x,z} \rho(\mathrm{d}x)\left(\int_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)^2 \cdot \hat{\mathcal{P}}^\pi(\mathrm{d}z|y)}{\sqrt{\rho(\mathrm{d}z)} \cdot \rho(\mathrm{d}y)}\right)^2 \\
&\quad \le \int_x \rho(\mathrm{d}x)\left(\max_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)}{\rho(\mathrm{d}y)}\right)^2 \int_z \rho(\mathrm{d}z) \cdot \left(\frac{\hat{\eta}^\pi(\mathrm{d}z|x)}{\gamma\rho(\mathrm{d}z)}\right)^2 \\
&\quad = \frac{1}{\gamma^2} \int_x \rho(\mathrm{d}x)\left(\max_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)}{\rho(\mathrm{d}y)}\right)^2 \int_z \hat{\eta}^\pi(\mathrm{d}z|x) \cdot \frac{\hat{\eta}^\pi(\mathrm{d}z|x)}{\rho(\mathrm{d}z)} \\
&\quad \le \frac{1}{\gamma^2} \int_x \rho(\mathrm{d}x)\left(\max_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)}{\rho(\mathrm{d}y)}\right)^2 \left(\max_z \frac{\hat{\eta}^\pi(\mathrm{d}z|x)}{\rho(\mathrm{d}z)}\right) \int_z \hat{\eta}^\pi(\mathrm{d}z|x) \\
&\quad = \frac{1}{\gamma^2} \int_x \rho(\mathrm{d}x)\left(\max_y \frac{\hat{\eta}^\pi(\mathrm{d}y|x)}{\rho(\mathrm{d}y)}\right)^3 \\
&\quad = \hat{C}^\pi(\rho)^2.
\end{aligned}
$$

Substituting in (D.1) gives

$$
\|G^\pi v\|_{4,\rho}^4 \le \frac{\gamma^4}{(1-\gamma)^4} \cdot \chi_\rho^2(\mathcal{P}^\pi \mid\mid \hat{\mathcal{P}}^\pi)^2 \cdot \hat{C}^\pi(\rho)^2 \cdot \|v\|_{4,\rho}^4,
$$

which concludes the proof. □

Proof of Theorem 1 for the $L_4(\rho)$ norm

Proof. From Lemma 3, we have $S^\pi V^\pi = V^\pi$. By definition $V_k = S^\pi V_{k-1} + \epsilon_k^{\text{value}}$. Using Lemma 4, we get

$$
\begin{aligned}
\|V^\pi - V_k\|_{4,\rho} &= \left\|S^\pi V^\pi - S^\pi V_{k-1} - \epsilon_k^{\text{value}}\right\|_{4,\rho} \\
&= \left\|G^\pi(V^\pi - V_{k-1}) - \epsilon_k^{\text{value}}\right\|_{4,\rho} \\
&\le \|G^\pi(V^\pi - V_{k-1})\|_{4,\rho} + \left\|\epsilon_k^{\text{value}}\right\|_{4,\rho} \\
&\le \gamma'\|V^\pi - V_{k-1}\|_{4,\rho} + \left\|\epsilon_k^{\text{value}}\right\|_{4,\rho},
\end{aligned}
$$

where we used Lemma 8 in the last step. Expanding this recursive inequality gives

$$\begin{aligned}\|V^\pi - V_k\|_{4,\rho} &\le \gamma'^k \cdot \|V^\pi - V_0\|_{4,\rho} + \sum_{i=1}^{k} \gamma'^{k-i} \|\epsilon_k^{\text{value}}\|_{4,\rho} \\ &\le \gamma'^k \cdot \|V^\pi - V_0\|_{4,\rho} + \epsilon^{\text{value}} \sum_{i=1}^{k} \gamma'^{k-i} \\ &\le \gamma'^k \cdot \|V^\pi - V_0\|_{4,\rho} + \frac{1-\gamma'^k}{1-\gamma'} \cdot \epsilon^{\text{value}},\end{aligned}$$

which completes the proof. □

D.3 Proofs for convergence of OS-VI for Control

We prove Theorem 2 for the L_∞ and $L_4(\rho)$ cases separately.

For the L_∞ part, we break its proof into two lemmas.

Lemma 9. *Assume that $k \ge 1$. Let γ' be the effective discounted factor defined in Theorem 2 for the $\star = \infty$ case. Then,*

$$\|V^{\pi_k} - V^*\|_\infty \le \frac{2\gamma'}{1-\gamma'} \|V_{k-1} - V^*\|_\infty + \frac{1}{1-\gamma'} \left\|\epsilon_k^{policy}\right\|_\infty.$$

Proof. Let $\preccurlyeq$ be the componentwise inequality. Using Lemma 4 and Lemma 5 we write

$$\begin{aligned}V^* - V^{\pi_k} &= S^{\pi^*}V^* - S^{\pi^*}V_{k-1} + S^{\pi^*}V_{k-1} - S^*V_{k-1} + S^*V_{k-1} - S^{\pi_k}V_{k-1} + S^{\pi_k}V_{k-1} - S^{\pi_k}V^{\pi_k} \\ &\preccurlyeq G^{\pi^*}(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V_{k-1} - V^{\pi_k}) \\ &= G^{\pi^*}(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V_{k-1} - V^*) + G^{\pi_k}(V^* - V^{\pi_k}) \\ &= (G^{\pi^*} - G^{\pi_k})(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V^* - V^{\pi_k})\end{aligned}$$

Note that by definition $V^* - V^{\pi_k} \succcurlyeq 0$. Thus, we get

$$\begin{aligned}\|V^* - V^{\pi_k}\|_\infty &\le \left\|(G^{\pi^*} - G^{\pi_k})(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V^* - V^{\pi_k})\right\|_\infty \\ &\le \left(\left\|G^{\pi^*}\right\|_\infty + \|G^{\pi_k}\|_\infty\right)\|V^* - V_{k-1}\|_\infty + \left\|\epsilon_k^{\text{policy}}\right\|_\infty + \|G^{\pi_k}\|_\infty \|V^* - V^{\pi_k}\|_\infty \\ &\le 2\gamma'\|V^* - V_{k-1}\|_\infty + \left\|\epsilon_k^{\text{policy}}\right\|_\infty + \gamma'\|V^* - V^{\pi_k}\|_\infty.\end{aligned}$$

We conclude

$$\|V^* - V^{\pi_k}\|_\infty \le \frac{2\gamma'}{1-\gamma'}\|V^* - V_{k-1}\|_\infty + \frac{1}{1-\gamma'}\left\|\epsilon_k^{\text{policy}}\right\|_\infty.$$

□

Lemma 10. *Assume that $k \ge 1$. Let γ' be the effective discounted factor defined in Theorem 2 for the $\star = \infty$ case. Then for any $1 \le i \le k-1$ we have*

$$\|V_i - V^*\|_\infty \le \gamma'\|V_{i-1} - V^*\|_\infty + \left\|\epsilon_i^{value}\right\|_\infty.$$

Proof. Let $\pi'_i = \pi_V(V_{i-1})$. We have by Lemma 4 and Lemma 5

$$\begin{aligned}V^* - S^*V_{i-1} &= S^*V^* - S^{\pi'_i}V^* + S^{\pi'_i}V^* - S^{\pi'_i}V_{i-1} \succcurlyeq G^{\pi'_i}(V^* - V_{i-1}), \\ V^* - S^*V_{i-1} &= S^{\pi^*}V^* - S^{\pi^*}V_{i-1} + S^{\pi^*}V_{i-1} - S^*V_{i-1} \preccurlyeq G^{\pi^*}(V^* - V_{i-1}),\end{aligned}$$

where we used $S^*V^* \succcurlyeq S^{\pi'_i}V^*$ and $S^{\pi^*}V_{i-1} \preccurlyeq S^*V_{i-1}$.

Let $|\cdot|$ and $\max$ be componentwise functions. We get

$$\begin{aligned}
|V^* - S^* V_{i-1}| &\preccurlyeq \max\Big(\Big|G^{\pi'_i}(V^* - V_{i-1})\Big|, \Big|G^{\pi^*}(V^* - V_{i-1})\Big|\Big) \\
\Rightarrow \|V^* - S^* V_{i-1}\|_\infty &\leq \max\Big(\Big\|G^{\pi'_i}(V^* - V_{i-1})\Big\|_\infty, \Big\|G^{\pi^*}(V^* - V_{i-1})\Big\|_\infty\Big) \\
&\leq \max(\gamma'\|V^* - V_{i-1}\|_\infty, \gamma'\|V^* - V_{i-1}\|_\infty) \\
&= \gamma'\|V^* - V_{i-1}\|_\infty.
\end{aligned}$$

Finally, we write

$$\begin{aligned}
\|V_i - V^*\|_\infty &\leq \|V_i - S^* V_{i-1}\|_\infty + \|S^* V_{i-1} - V^*\|_\infty \\
&\leq \left\|\epsilon_i^{\text{value}}\right\|_\infty + \gamma'\|V^* - V_{i-1}\|_\infty,
\end{aligned}$$

as desired. □

Proof of Theorem 2 – the L_∞ case

Proof. Expanding the recursive result of Lemma 10, we get

$$\begin{aligned}
\|V_{k-1} - V^*\|_\infty &\leq \gamma'^{k-1}\|V_0 - V^*\|_\infty + \sum_{i=1}^{k-1} \gamma'^{k-1-i}\left\|\epsilon_i^{\text{value}}\right\|_\infty \\
&\leq \gamma'^{k-1}\|V_0 - V^*\|_\infty + \epsilon^{\text{value}} \sum_{i=1}^{k-1} \gamma'^{k-1-i} \\
&\leq \gamma'^{k-1}\|V_0 - V^*\|_\infty + \epsilon^{\text{value}} \cdot \frac{1 - \gamma'^{k-1}}{1 - \gamma'}.
\end{aligned}$$

Substituting this in Lemma 9, we get

$$\begin{aligned}
\|V^{\pi_k} - V^*\|_\infty &\leq \frac{2\gamma'}{1-\gamma'} \cdot \left[\gamma'^{k-1}\|V_0 - V^*\|_\infty + \epsilon^{\text{value}} \cdot \frac{1 - \gamma'^{k-1}}{1-\gamma'}\right] + \frac{1}{1-\gamma'}\left\|\epsilon_k^{\text{policy}}\right\|_\infty \\
&= \frac{2\gamma'^k}{1-\gamma'}\|V_0 - V^*\|_\infty + \frac{2\gamma'(1-\gamma'^{k-1})}{(1-\gamma')^2} \cdot \epsilon^{\text{value}} + \frac{1}{1-\gamma'} \cdot \left\|\epsilon_k^{\text{policy}}\right\|_\infty.
\end{aligned}$$

□

We introduce similar lemmas for the proof of the $L_4(\rho)$ part of Theorem 2.

Lemma 11. *Assume $k \geq 1$, and ρ is a distribution over state space. Let γ' be the effective discounted factor defined in in Theorem 2 for the $\star = 4, \rho$ case. Then*

$$\|V^{\pi_k} - V^*\|_{4,\rho} \leq \frac{2\gamma'}{1-\gamma'}\|V_{k-1} - V^*\|_{4,\rho} + \frac{1}{1-\gamma'}\left\|\epsilon_k^{\textit{policy}}\right\|_{4,\rho}.$$

Proof. Let $\preccurlyeq$ be the componentwise inequality. Exactly similar to proof of Lemma 9 we have by Lemma 4 and Lemma 5 that

$$\begin{aligned}
V^* - V^{\pi_k} &= S^{\pi^*}V^* - S^{\pi^*}V_{k-1} + S^{\pi^*}V_{k-1} - S^* V_{k-1} + S^* V_{k-1} - S^{\pi_k}V_{k-1} + S^{\pi_k}V_{k-1} - S^{\pi_k}V^{\pi_k} \\
&\preccurlyeq G^{\pi^*}(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V_{k-1} - V^{\pi_k}) \\
&= G^{\pi^*}(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V_{k-1} - V^*) + G^{\pi_k}(V^* - V^{\pi_k}) \\
&= (G^{\pi^*} - G^{\pi_k})(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V^* - V^{\pi_k}).
\end{aligned}$$

Note that by definition $V^* - V^{\pi_k} \succcurlyeq 0$. Thus, using Lemma 8 we can write

$$\begin{aligned}
\|V^* - V^{\pi_k}\|_{4,\rho} &\leq \left\|(G^{\pi^*} - G^{\pi_k})(V^* - V_{k-1}) - \epsilon_k^{\text{policy}} + G^{\pi_k}(V^* - V^{\pi_k})\right\|_{4,\rho} \\
&\leq \left\|G^{\pi^*}(V^* - V_{k-1})\right\|_{4,\rho} + \|G^{\pi_k}(V^* - V_{k-1})\|_{4,\rho} + \|G^{\pi_k}(V^* - V^{\pi_k})\|_{4,\rho} + \left\|\epsilon_k^{\text{policy}}\right\|_{4,\rho} \\
&\leq 2\gamma'\|V^* - V_{k-1}\|_{4,\rho} + \left\|\epsilon_k^{\text{policy}}\right\|_{4,\rho} + \gamma'\|V^* - V^{\pi_k}\|_{4,\rho}
\end{aligned}$$

We conclude that

$$\|V^* - V^{\pi_k}\|_{4,\rho} \leq \frac{2\gamma'}{1-\gamma'}\|V^* - V_{k-1}\|_{4,\rho} + \frac{1}{1-\gamma'}\left\|\epsilon_k^{\text{policy}}\right\|_{4,\rho}$$

□

Lemma 12. *Assume that $k \geq 1$. Let γ' be the effective discounted factor defined in Theorem 2 for the $\star = 4, \rho$ case. Then for any $1 \leq i \leq k-1$ we have*

$$\|V_i - V^*\|_{4,\rho} \leq \gamma'\|V_{i-1} - V^*\|_{4,\rho} + \left\|\epsilon_i^{\textit{value}}\right\|_{4,\rho}$$

Proof. Let $\pi_i' = \pi_V(V_{i-1})$. We have by Lemma 4 and Lemma 5

$$\begin{aligned} V^* - S^*V_{i-1} &= S^*V^* - S^{\pi_i'}V^* + S^{\pi_i'}V^* - S^{\pi_i'}V_{i-1} \succcurlyeq G^{\pi_i'}(V^* - V_{i-1}) \\ V^* - S^*V_{i-1} &= S^{\pi^*}V^* - S^{\pi^*}V_{i-1} + S^{\pi^*}V_{i-1} - S^*V_{i-1} \preccurlyeq G^{\pi^*}(V^* - V_{i-1}) \end{aligned}$$

Let $|\cdot|$ and $\max$ be componentwise functions. We get from Lemma 8

$$\begin{aligned} |V^* - S^*V_{i-1}| &\preccurlyeq \max\left(\left|G^{\pi_i'}(V^* - V_{i-1})\right|, \left|G^{\pi^*}(V^* - V_{i-1})\right|\right) \\ \Rightarrow \|V^* - S^*V_{i-1}\|_{4,\rho}^4 &\leq \left\|G^{\pi_i'}(V^* - V_{i-1})\right\|_{4,\rho}^4 + \left\|G^{\pi^*}(V^* - V_{i-1})\right\|_{4,\rho}^4 \\ &\leq 2\max\left(\frac{\gamma^4}{(1-\gamma)^4}\hat{C}^{\pi_i'}(\rho)^2 \cdot \chi_\rho^2(\mathcal{P}^{\pi_i'} \mid\mid \hat{\mathcal{P}}^{\pi_i'})^2,\right. \\ &\qquad \left.\frac{\gamma^4}{(1-\gamma)^4}\hat{C}^{\pi^*}(\rho)^2 \cdot \chi_\rho^2(\mathcal{P}^{\pi^*} \mid\mid \hat{\mathcal{P}}^{\pi^*})^2\right)\|V^* - V_{i-1}\|_{4,\rho}^4 \\ &\leq \gamma'^4\|V^* - V_{i-1}\|_{4,\rho}^4. \end{aligned}$$

Finally we write

$$\begin{aligned} \|V_i - V^*\|_{4,\rho} &\leq \|V_i - S^*V_{i-1}\|_{4,\rho} + \|S^*V_{i-1} - V^*\|_{4,\rho} \\ &\leq \left\|\epsilon_i^{\text{value}}\right\|_{4,\rho} + \gamma'\|V^* - V_{i-1}\|_{4,\rho} \end{aligned}$$

□

Proof of Theorem 2 – $L_4(\rho)$ case

Proof. Using Lemma 11 and Lemma 12 the proof follows exactly like the proof of the L_∞ case. Expanding the recursive result of Lemma 12, we get

$$\begin{aligned} \|V_{k-1} - V^*\|_{4,\rho} &\leq \gamma'^{k-1}\|V_0 - V^*\|_{4,\rho} + \sum_{i=1}^{k-1}\gamma'^{k-1-i}\|\epsilon_i^{\text{value}}\|_{4,\rho} \\ &\leq \gamma'^{k-1}\|V_0 - V^*\|_{4,\rho} + \epsilon^{\text{value}}\sum_{i}^{k-1}\gamma'^{k-1-i} \\ &\leq \gamma'^{k-1}\|V_0 - V^*\|_{4,\rho} + \epsilon^{\text{value}} \cdot \frac{1-\gamma'^{k-1}}{1-\gamma'}. \end{aligned}$$

Substituting this in Lemma 11, we get

$$\begin{aligned} \|V^{\pi_k} - V^*\|_{4,\rho} &\leq \frac{2\gamma'}{1-\gamma'} \cdot \left[\gamma'^{k-1}\|V_0 - V^*\|_{4,\rho} + \epsilon^{\text{value}} \cdot \frac{1-\gamma'^{k-1}}{1-\gamma'}\right] + \frac{1}{1-\gamma'}\left\|\epsilon_k^{\text{policy}}\right\|_{4,\rho} \\ &= \frac{2\gamma'^k}{1-\gamma'}\|V_0 - V^*\|_{4,\rho} + \frac{2\gamma'(1-\gamma'^{k-1})}{(1-\gamma')^2} \cdot \epsilon^{\text{value}} + \frac{1}{1-\gamma'} \cdot \left\|\epsilon_k^{\text{policy}}\right\|_{4,\rho} \end{aligned}$$

□

E Additional experiments

In this section, we present further experiments on our algorithms. We consider multiple environments and settings. Section E.1 introduces the environments used. In the next sections three sets of experiments are presented.

1. We further verify OS-VI acceleration compared to VI in three environments for both PE and control problems.
2. We investigate the effect of model error on OS-VI. Two model error forumlations are tested in all environments.
3. OS-Dyna is compared to Dyna and model-free algorithms using two learning schedules.

E.1 Environments

We do experiments in three different environments. The details of them are as follows.

Modified Cliffwalk. We design a modified cliffwalk environment to better show the differences among algorithms. The environment is a 6×6 gridworld shown in Figure 3. The starting state is the top-left corner. The agent receives reward of 20 in the top-right corner i.e. every action taken from this state gives reward of $r(x, a) = 20$. There are three holes in the middle four cells of the first, third and the fifth row that if fallen into, the agent gets stuck and receives reward of -32, -16, and -8 on every step, respectively. There is a penalty of -1 in all other states to encourage finding the shortest route to the goal. The agent has four actions: UP, RIGHT, DOWN, and LEFT. Each action has 90% chance to successfully move the agent in the chosen direction. With probability of 10% one of the other three directions is randomly chosen and the agent moves in that direction. If the agent attempts to go out of the environment, it will stay in place.

The discount factor of this environment is $\gamma = 0.9$. For policy evaluation experiments, we evaluate the optimal policy. The optimal policy is to take the safest route among (the two, or the only) closest paths to the right side of environment, as shown in Figure 3. Note that the smoothed models introduced in (6.1) will over-estimate the danger of cliffs and may take a suboptimal longer path to be safer. This will make the solution of inaccurate models suboptimal.

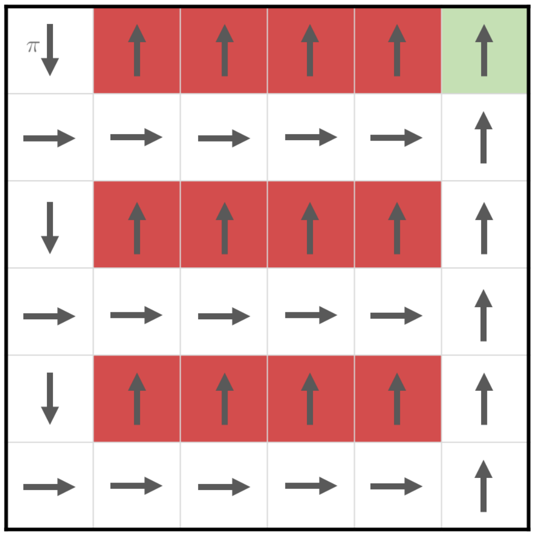


Figure 3: Modified Cliffwalk environment. The red cells in the first, third, and the third rows are holes with penalty of -32, -16, -8, respectively. The top-right corner is the goal state with reward of 20. All other states have penalty of -1. There is 10% failure probability in actions. Arrows show the policy used for policy evaluation experiments.

Random MDP (Garnet). We use the Garnet environment used by Bhatnagar et al. [2009], Farahmand and Ghavamzadeh [2021]. The name is originally chosen as the acronym of *Generic Average*

Reward Non-stationary Environment Testbed. We use the same name, even though we implement it for discounted stationary MDPs similar to Farahmand and Ghavamzadeh [2021].

Our Garnet problem is parametrized by the tuple $(|\mathcal{X}|, |\mathcal{A}|, b_P, b_r)$. Here, $|\mathcal{X}|$ and $|\mathcal{A}|$ are the number of states and actions. The value b_p is the branching factor of the environment, which is the number of possible next states for each state-action pair. When generating an instance, for each state-action pair, we randomly select b_p states without replacement as the possible next states. Then, the transition distribution is generated by randomly choosing $b_p - 1$ points on the $(0, 1)$ interval. These points will partition the interval into b_p parts, each corresponding to one of the transition probabilities. The reward function is only state-dependent. We select b_r states without replacement, and for each chosen state x, we assign $r(x)$ a uniformly sampled value in $(0, 1)$.

We generate 100 randomly generated instances of the Garnet problem with $|\mathcal{X}| = 50$, $|\mathcal{A}| = 4$, $b_P = 3$, and $b_r = 5$. The discount factor is chosen as $\gamma = 0.99$. For policy evaluation experiments, we use the optimal policy of the instance. The plots for Garnet show the average values on the 100 problem instances along with a shaded area showing one standard error.

Maze Environment. This is a simple 3×3 maze shown in Figure 4. The top-left corner is the initial state, and the top-right corner is the goal state with reward of 1. Similar to the modified clifffwalk, the agent has four actions: UP, RIGHT, DOWN, and LEFT. Each action has 90% chance to successfully move the agent in the chosen direction. With probability of 10% one of the other three directions is randomly chosen and the agent moves in that direction. If the agent attempts to go out of the environment or hits a wall, it will stay in place. The discount factor is $\gamma = 0.9$. We use the optimal policy shown by arrows in Figure 4 for policy evaluation experiments.

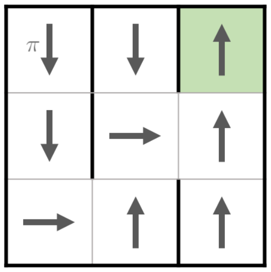


Figure 4: Maze environment. The top-right corner is the goal state with reward of 1. There is 10% failure probability in actions. Arrows show the policy used for policy evaluation experiments.

E.2 Convergence rate of OS-VI

We empirically compare OS-VI with VI in all three environments. In this section and Section E.3, we evaluate the normalized error of V_k, which is

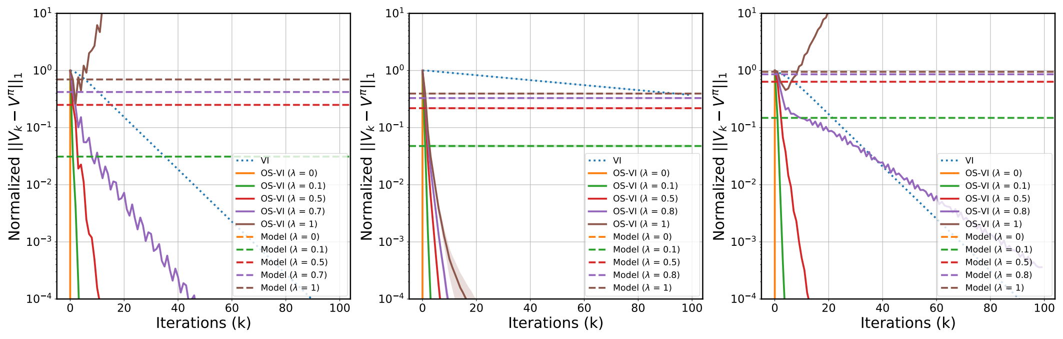
$$\frac{\|V_k - V^\pi\|_1}{\|V^\pi\|_1} \tag{E.1}$$

for the policy evaluation problem, and

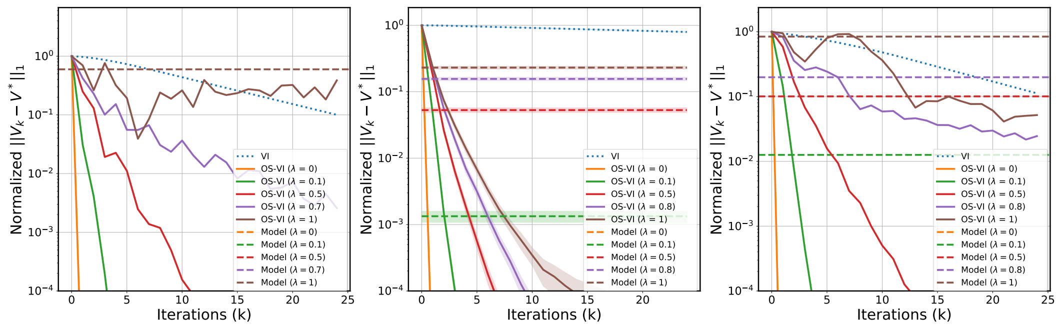
$$\frac{\|V_k - V^*\|_1}{\|V^*\|_1} \tag{E.2}$$

for the control problem. In control problem, while V^{π_k} has qualitatively same behaviors as V_k, it usually converges too fast and has instabilities. Therefore, we consider the normalized error of V_k instead of V^{π_k} to better see the convergence behaviors..

The results are shown in Figure 5. The dashed lines show the error values obtained by just using the model $\hat{\mathcal{P}}$. We see that OS-VI converges much faster than VI across model errors and environments for both the policy evaluation and control problems. Also note that only using the model $\hat{\mathcal{P}}$ would not be enough since the solution of models with error is not correct.



(a) Policy Evaluation



(b) Control

Figure 5: Comparison of OS-VI with VI and the solution of the model, in policy evaluation *(a)* and control *(b)* problems. The comparison is done in maze *(left)*, Garnet *(middle)*, and modified cliffwalk *(right)* environments. Garnet plots are average of 100 instances. The shaded area is one standard error.

E.3 Effect of model error on OS-VI

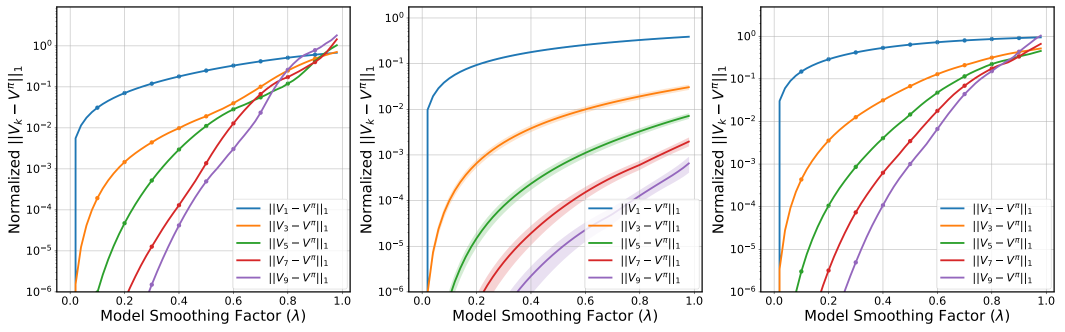
As shown in the theory, the convergence rate of OS-VI is affected by the accuracy of $\hat{\mathcal{P}}$. In this section we further investigate this effect. To do so, we run OS-VI with smoothed models obtained by a range of smoothing parameters. The smoothed models are defined in (6.1). A higher smoothing parameter λ will make the transition distributions more uniform and less accurate.

The normalized error of OS-VI in few of initial iterations is plotted against the smoothing parameter λ in Figure 6. The difference among the iterations shows the convergence rate of OS-VI. It can be seen that for larger errors, the errors in different iterations become more similar, which means the algorithm has not progressed much towards the correct solution. In very large values of λ there are cases that the order of iterations has changed. More specifically, later iterations have larger error than earlier ones. This means that algorithm is diverging.

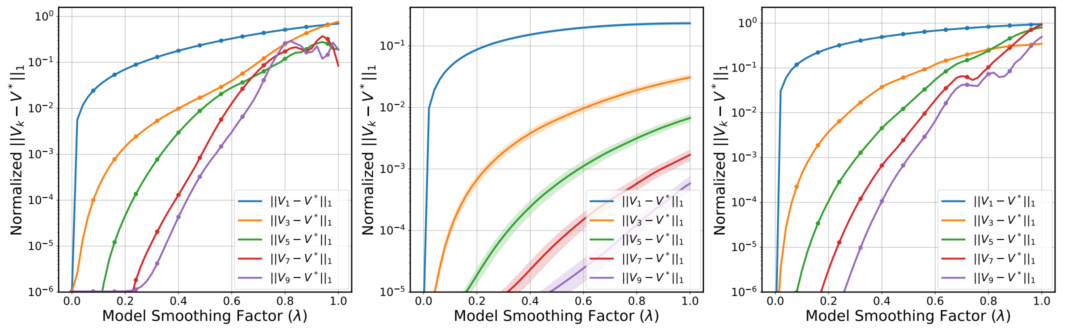
To better show the divergence scenario of OS-VI, we introduce a new notion of model error. The *self-loop perturbed model* is defined as

$$\hat{\mathcal{P}}(\cdot|x, a; \mathcal{P}, \lambda) = (1 - \lambda)\mathcal{P}(\cdot|x, a) + \lambda I(\cdot|x) \tag{E.3}$$

where $I(\cdot|x)$ is the distribution of deterministically staying in state x. Larger values of λ push the model transitions towards an MDP where no transitions occur. Similar to smoothing, larger values of λ lead to a more inaccurate model. The effect of this new model error is shown in Figure 7. We see that a clear divergence happens for large values of λ. The error of OS-VI increases with each iteration.



(a) Policy Evaluation



(b) Control

Figure 6: Effect of smoothing on convergence of OS-VI in policy evaluation *(a)* and control *(b)* problems. Models are obtained by smoothing as in (6.1). The comparison is done in maze *(left)*, Garnet *(middle)*, and modified cliffwalk *(right)* environments. Garnet plots are average of 100 instances. The shaded area is one standard error.

E.4 Additional Experiments on OS-Dyna

In this section we compare OS-Dyna with Dyna and model-free algorithms. We focus on the modified cliffwalk environment and smoothed MLE models defined in Section 6.

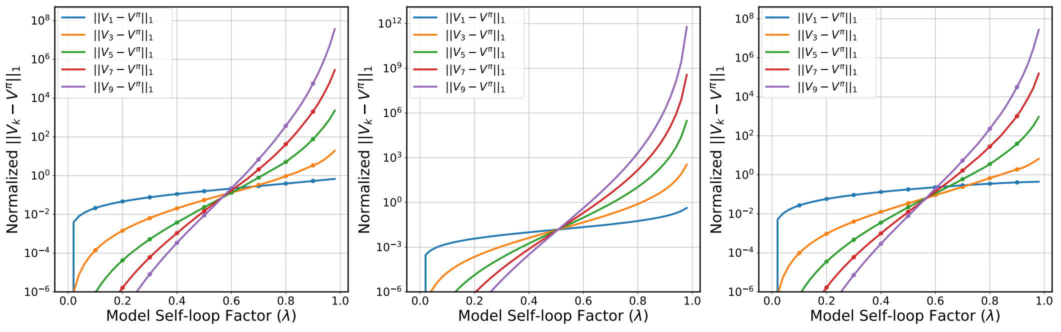
In the implementation of OS-Dyna, V_k is calculated from $\bar{r}_k$ and the $\hat{\mathcal{P}}$ through exact dynamic programming to reduce the noise. Specifically, we find the optimal value function $V^*(\hat{\mathcal{P}}, \bar{r}_k)$ and the value function $V^\pi(\hat{\mathcal{P}}, \bar{r}_k)$ by performing VI on MDP $(\mathcal{X}, \mathcal{A}, \hat{\mathcal{P}}, \bar{r}_k)$. The same is true for Dyna. The value function is updated to the exact solution of the model on every iteration.

In policy evaluation, we compare to TD-Learning, which is to update the value function in the following way with each sample $(X_t, \pi(X_t), R_t, X'_t)$:

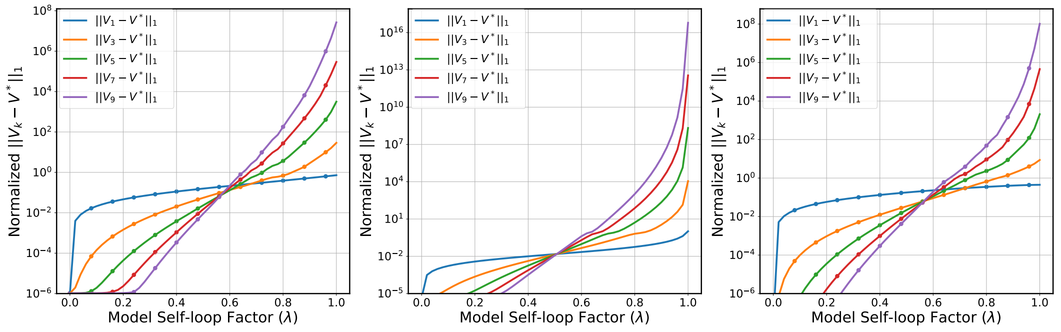
$$V(X_t) \leftarrow V(X_t) + \alpha_t(R_t + \gamma \cdot V(X'_t) - V(X_t).) \tag{E.4}$$

Here, α_t is the learning rate at step t. We use constant and rescaled linear [Wainwright, 2019] learning rate schedules in PE experiments. The rescaled linear schedule sets $\alpha_t = \frac{\alpha}{1+(1-u)\cdot t}$. We fine tune the learning rate schedule for each algorithm independently such that 0.1 normalized error (E.1) is achieved as fast as possible. In constant learning rate, the value of α is 0.2 for TD-Learning and 0.05 for all OS-Dyna instances. In rescaled linear schedule, $\alpha, u = 1, 0.999$ for TD-Learning, and $\alpha, u = 0.8, 0.995$ for OS-Dyna.

The results for PE are shown in Figure 8. It can be seen that OS-Dyna converges faster than TD-Learning in both learning rate schedules. Also note that unlike OS-Dyna, Dyna does not converge to true values in presence of model error. It is worth mentioning that Dyna without model error is the best one can do in policy evaluation problem without any addition assumptions on the environment. Thus it is expected to outperform all other algorithms.



(a) Policy Evaluation



(b) Control

Figure 7: Effect of self-loop error on convergence of OS-VI in policy evaluation *(a)* and control *(b)* problems. Models are obtained by self-loop perturbation as in (E.3). The comparison is done in maze *(left)*, Garnet *(middle)*, and modified cliffwalk *(right)* environments. Garnet plots are average of 100 instances. The shaded area is one standard error.

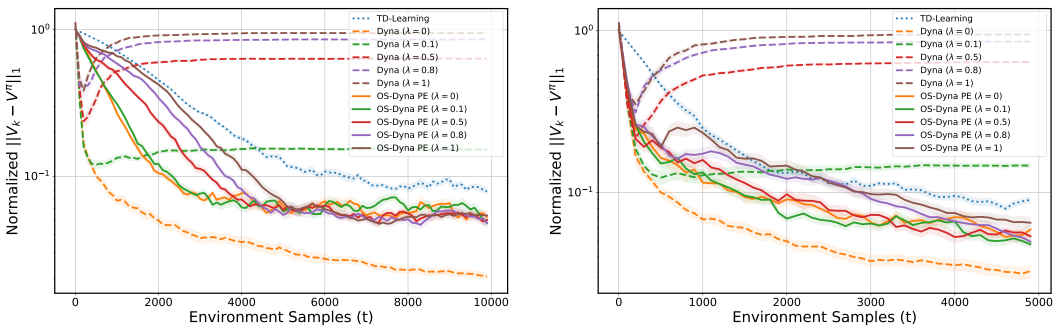


Figure 8: Comparison of OS-Dyna with Dyna and TD-learning in PE using constant *(Left) and rescaled linear (Right) learning rates. This is average over 20 runs. The shaded area is one standard error.*

In control, OS-Dyna is compared with Dyna and Q-Learning using the delayed decay [Sutton and Barto, 2019] and rescaled linear [Wainwright, 2019] learning rate schedules. The delayed decay sets $\alpha_t = \alpha$ for $t \leq N$ and $\alpha_t = \alpha/(t - N)$ otherwise. The learning rates for each algorithm is fine tuned to achieve the optimal policy as fast as possible and stay stable on it. The results are shown in Figure 9.

In delayed decay learning rate, we have $\alpha, N = 0.02, 68000$ for Q-Learning. For instances of OS-Dyna we have

- OS-Dyna $(\lambda = 0)$: $\alpha, N = 0.02, 30000$

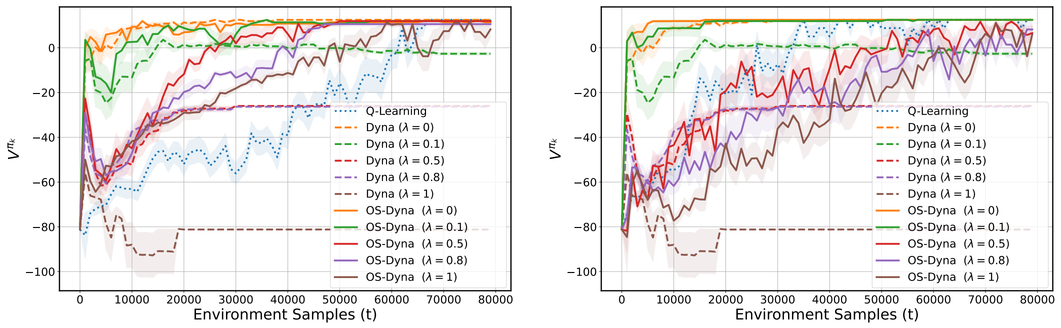


Figure 9: Comparison of OS-Dyna with Dyna and Q-learning in control using delayed decay *(Left) and rescaled linear (Right) learning rates. This is average over 20 runs. The shaded area is one standard error.*

- OS-Dyna ($\lambda = 0.1$): $\alpha, N = 0.02, 35000$
- OS-Dyna ($\lambda = 0.5$): $\alpha, N = 0.02, 50000$
- OS-Dyna ($\lambda = 0.8$): $\alpha, N = 0.02, 48000$
- OS-Dyna ($\lambda = 1$): $\alpha, N = 0.02, 80000$

In rescaled linear learning rate, we have $\alpha, u = 0.1, 0.9999$ for Q-Learning. For instances of OS-Dyna we have

- OS-Dyna ($\lambda = 0$): $\alpha, u = 1, 0.9$
- OS-Dyna ($\lambda = 0.1$): $\alpha, u = 1, 0.9$
- OS-Dyna ($\lambda = 0.5$): $\alpha, u = 1, 0.9995$
- OS-Dyna ($\lambda = 0.8$): $\alpha, u = 1, 0.9995$
- OS-Dyna ($\lambda = 1$): $\alpha, u = 1, 0.9995$

F Extended related work

In this section, we provide a comparative analysis of the convergence behaviour of OS-VI (Appendix F.1). We also point to some work where the Jacobi and Gauss-Seidel iterations or some form of matrix splitting have been studied in the context of dynamic programming (Appendix F.2).

F.1 Comparison of convergence behaviours of OS-VI, value iteration, policy iteration, and modified policy iteration

We briefly compared the convergence behaviour of OS-VI with the convergence of VI in Section 4 after stating Theorem 2. Here, we expand that discussion, and include comparison with PI and MPI as well. We consider three aspects:

1. *(Model Error)* How does the model error affect the convergence limit?
2. *(Transient Error)* How fast the initial error in the approximation of value function diminishes as the iteration number k grows?
3. *(Approximation Error Amplification)* How are the errors at each step of these algorithms amplified and do affect the outcome policy?

Let us discuss the *Model Error* first, as it is a crucial difference between OS-VI and other methods such as VI, PI, and MPI. Suppose that we do not know $\mathcal{P}$, but only have access to $\hat{\mathcal{P}} \neq \mathcal{P}$, and we use the approximate model with VI (2.2) (that is, we perform $V_k \leftarrow \max_\pi\{r^\pi + \gamma\hat{\mathcal{P}}V_{k-1}\}$), or PI or MPI, as shall be recalled soon. These methods then converge to the optimal value function/policy w.r.t. the dynamics $\hat{\mathcal{P}}$. Let us denote the optimal value function w.r.t. $\hat{\mathcal{P}}$ by $\hat{V}^*$, and the optimal policy $\hat{\pi}^*$. These are, in general, different from the optimal value function V^* and policy π^* w.r.t. the dynamics $\mathcal{P}$ – they are biased.

If we execute $\hat{\pi}^*$ in the true environment with dynamics $\mathcal{P}$, its performance will be lower than the performance of the optimal policy π^*, measured according to their corresponding value functions. The performance can be upper bounded as follows [Ávila Pires and Szepesvári, 2016, Theorem 7]:

$$\left\|V^{\pi^*} - V^{\hat{\pi}^*}\right\|_\infty \leq \frac{2\gamma}{1-\gamma}\left\|(\mathcal{P} - \hat{\mathcal{P}})\hat{V}^*\right\|_\infty \leq \frac{2\gamma V_{\max}}{1-\gamma}\left\|\mathcal{P} - \hat{\mathcal{P}}\right\|_\infty.$$

For the methods that only rely on the approximate model $\hat{\mathcal{P}}$, this performance deterioration is in general inevitable. This is the essence of sim2real problem.

OS-VI does not have this issue. By using both $\hat{\mathcal{P}}$ and $\mathcal{P}$, it brings the potential benefit of querying an approximate model (which is supposedly computationally cheaper), while guarding against converging to a biased solution. Of course, this is under the condition that it converges, which is the case if the model is accurate enough. Note that OS-VI uses more information (both $\hat{\mathcal{P}}$ and $\mathcal{P}$) than VI, PI, or MPI, which only use either $\mathcal{P}$ or $\hat{\mathcal{P}}$ – they cannot benefit from both.

Since OS-VI needs access to $\mathcal{P}$, one may wonder if it is beneficial to use OS-VI after all, as opposed to using VI, PI, or MPI with the true model $\mathcal{P}$. The answer to this question depends on the convergence rate of these algorithms. A faster algorithm requires fewer queries to the true model $\mathcal{P}$. The rest of this subsection is dedicated to studying their convergence rates, focusing on the effect of transient error and the approximation error amplification.

To set the stage, let us introduce the approximate models of VI, PI, and MPI. These should be compared with (4.1) and (4.5)-(4.6) in Section 4.

Recall that VI iteratively applies the Bellman operator T to the previous value function V_{k-1} in order to obtain the new approximation V_k of the value function, cf. (2.2). As discussed for OS-VI in the beginning of Section 4.1, there might be an error in each step, which we formalize by considering that an error function $\epsilon_k^{\text{value}}$ is added to the operation of the exact VI:

$$V_k = \begin{cases} T^\pi V_{k-1} + \epsilon_k^{\text{value}}, & \text{(Policy Evaluation)} \\ T^* V_{k-1} + \epsilon_k^{\text{value}}. & \text{(Control)} \end{cases}$$

When $\varepsilon_k^{\text{value}} = 0$, we get the exact VI (2.2).

At each iteration of PI, we first compute the greedy policy $\pi_k \leftarrow \pi_g(V_{k-1})$ and then perform PE in order to compute V^{π_k}. In approximate PI, we might have error at computing the greedy policy or computing its value function. These errors are modelled as

$$T^{\pi_k} V_{k-1} = T^* V_{k-1} + \epsilon_k^{\text{policy}}, \qquad \text{(policy improvement)} \tag{F.1}$$

$$V_k = V^{\pi_k} + \epsilon_k^{\text{value}} \; [= (\mathbf{I} - \mathcal{P}^{\pi_k})^{-1} r^{\pi_k} + \epsilon_k^{\text{value}}], \text{(policy evaluation)} \tag{F.2}$$

The Modified PI is similar to PI with the difference that instead of aiming to compute V^{π_k} at each step exactly (ignoring the $\epsilon_k^{\text{value}}$ term for the moment), it only partially moves towards it by applying T^{π_k} for $m \geq 1$ times. That is, $V_k \leftarrow (T^{\pi_k})^m V_{k-1}$. When $m \to \infty$, by the contraction property of the Bellman operator, $V_k \to V^{\pi_k}$. This is exactly the same as PI. When $m = 1$, it is the same as VI. The approximate MPI is modelled as

$$\begin{aligned} T^{\pi_k} V_{k-1} &= T^* V_{k-1} + \epsilon_k^{\text{policy}}, && \text{(policy improvement).} \\ V_k &= (T^{\pi_k})^m V_{k-1} + \epsilon_k^{\text{value}}. && \text{(partial policy evaluation).} \end{aligned}$$

For simplicity of comparison, we focus on the supremum norm-based analysis for each of these methods. Some of the existing results are not exactly in the form that we need. For example, they take $k \to \infty$, which loses the information about the transient error. Whenever possible, we re-use them in order to obtain error bounds for VI, PI, and MPI.

Convergence of value iteration. We consider VI (PE) and VI (Control) separately. For VI (PE), we derive the bound as follows:

$$\begin{aligned} V^\pi - V_k &= T^\pi V^\pi - (T^\pi V_{k-1} + \epsilon_k^{\text{value}}) = \gamma \mathcal{P}^\pi (V^\pi - V_{k-1}) + \epsilon_k^{\text{value}} = \cdots \\ &= \sum_{i=0}^{k-1} (\gamma \mathcal{P}^\pi)^i \epsilon_{k-i}^{\text{value}} + (\gamma \mathcal{P}^\pi)^k (V^\pi - V_0). \end{aligned}$$

Assume that $\|\epsilon_i^{\text{value}}\|_\infty \leq \epsilon^{\text{value}}$ for all $i = 1, \ldots, k$. We then have

$$\|V^\pi - V_k\|_\infty \leq \frac{1-\gamma^k}{1-\gamma} \epsilon^{\text{value}} + \gamma^k \|V^\pi - V_0\|_\infty. \tag{F.3}$$

This upper bound shows the effect of transient error and the approximation error at each iteration. The transient error decays with the rate of $O(\gamma^k)$. This can go to zero quite slowly when the discount factor is close to one. The approximation errors $\epsilon_i^{\text{value}}$ of the approximate VI procedure, upper bounded by ϵ^{value}, are amplified by a factor of $(1-\gamma)^{-1}$. Asymptotically, we have $\frac{\epsilon^{\text{value}}}{1-\gamma}$ behaviour.

This result should be compared with Theorem 1 with the choice of $\star = \infty$, which shows that OS-VI (PE) behaves as

$$\|V^\pi - V_k\|_\infty \leq \frac{1-\gamma'^k}{1-\gamma'} \epsilon^{\text{value}} + \gamma'^k \|V^\pi - V_0\|_\infty, \tag{F.4}$$

with $\gamma' = \frac{\gamma}{1-\gamma} \|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty$. When the model is accurate enough ($\|\mathcal{P}^\pi - \hat{\mathcal{P}}^\pi\|_\infty < 1-\gamma$), the effective discount factor γ' is smaller than the discount factor γ of the original MDP. Consequently, the transient error of OS-VI can decay significantly faster than VI's. Moreover, the error amplification of ϵ^{value} is by a factor of $(1-\gamma')^{-1}$, which is smaller than that of approximate VI.

We also have a similar result for VI (Control). We follow the proof of Equation (2.2) of Munos [2007] to get that for the greedy policy $\pi_k \leftarrow \pi_g(V_{k-1})$, we have

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma}{1-\gamma} \|V^* - V_{k-1}\|_\infty. \tag{F.5}$$

To upper bound $\|V^* - V_{k-1}\|_\infty$, we add and subtract $T^* V_{k-2}$ to $V^* - V_{k-1}$, and benefit from $V^* = T^* V^*$ and the triangle inequality to get

$$\begin{aligned} \|V^* - V_{k-1}\|_\infty &\leq \|T^* V^* - T^* V_{k-2}\|_\infty + \|T^* V_{k-2} - V_{k-1}\| \\ &\leq \gamma \|V^* - V_{k-2}\|_\infty + \|T^* V_{k-2} - V_{k-1}\| \\ &= \gamma \|V^* - V_{k-2}\|_\infty + \|\epsilon_{k-1}^{\text{value}}\|_\infty. \end{aligned}$$

Repeating this argument, we obtain

$$\|V^* - V_{k-1}\|_\infty \leq \sum_{i=0}^{k-1} \gamma^i \left\|\epsilon_{k-i}^{\text{value}}\right\|_\infty + \gamma^k \|V^* - V_0\|_\infty.$$

Plugging this inequality in (F.5) and using the same assumption that $\|\epsilon_i^{\text{value}}\|_\infty \leq \epsilon^{\text{value}}$ for all $i = 1, 2, \dots$ lead to

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma}{1-\gamma}\left[\frac{1-\gamma^{k-1}}{1-\gamma}\epsilon^{\text{value}} + \gamma^{k-1}\|V^* - V_0\|_\infty\right]. \tag{F.6}$$

The transient behaviour is $O(\gamma^k)$, as in VI (PE). The amplification of the approximation errors is by a factor of $(1-\gamma)^{-2}$. The result for VI (Control) should be compared with Theorem 2 with $\star = \infty$, which is

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma'^k}{1-\gamma'}\|V_0 - V^*\|_\infty + \frac{2\gamma'(1-\gamma'^{k-1})}{(1-\gamma')^2}\,\epsilon^{\text{value}} + \frac{1}{1-\gamma'}\left\|\epsilon_k^{\text{policy}}\right\|_\infty. \tag{F.7}$$

As in the OS-VI (PE) case, the transient behaviour is $O(\gamma'^k)$, which can be much faster than VI's whenever the approximate model is accurate enough. The error amplification is $(1-\gamma')^{-2}$, which is smaller under the same condition. We have an extra $\|\epsilon_k^{\text{policy}}\|_\infty$ term, which is the possible error in the computation of the S-improved policy. The parallel for VI would be the error in the computation of the greedy policy. In the VI model considered above, we did not consider such a source of error.

Convergence of policy iteration. Considering that $\epsilon_i^{\text{policy}} = 0$ in (F.1), we use Lemma 4 of Munos [2003], which states that

$$V^* - V^{\pi_k} \preccurlyeq \gamma\mathcal{P}^{\pi^*}(V^* - V^{\pi_{k-1}}) + \gamma\left[\mathcal{P}^{\pi_k}(\mathbf{I} - \gamma\mathcal{P}^{\pi_k})^{-1}(\mathbf{I} - \gamma\mathcal{P}^{\pi_{k-1}}) - \mathcal{P}^{\pi^*}\right](V_{k-1} - V^{\pi_{k-1}}).$$

Noticing that $0 \preccurlyeq V^* - V^{\pi_k}$, by taking the absolute values of both sides, and using Jensen's inequality, we get that

$$\begin{aligned}|V^* - V^{\pi_k}| \preccurlyeq &\gamma\mathcal{P}^{\pi^*}|V^* - V^{\pi_{k-1}}| + \\ &\gamma\left[\mathcal{P}^{\pi_k}(\mathbf{I} - \gamma\mathcal{P}^{\pi_k})^{-1} + \gamma\mathcal{P}^{\pi_k}(\mathbf{I} - \gamma\mathcal{P}^{\pi_k})^{-1}\mathcal{P}^{\pi_{k-1}} + \mathcal{P}^{\pi^*}\right]|V_{k-1} - V^{\pi_{k-1}}|.\end{aligned}$$

Taking the supremum of both sides over the state space, and benefitting from $\|\mathcal{P}^\pi\|_\infty = 1$ and that $\left\|(\mathbf{I} - \gamma\mathcal{P}^\pi)^{-1}\right\|_\infty \leq \frac{1}{1-\gamma}$ (for any π), we obtain

$$\|V^* - V^{\pi_k}\|_\infty \leq \gamma\|V^* - V^{\pi_{k-1}}\|_\infty + \frac{2\gamma}{1-\gamma}\left\|\epsilon_{k-1}^{\text{value}}\right\|_\infty.$$

Expanding this, we get

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma}{1-\gamma}\sum_{i=0}^{k-1}\gamma^i\left\|\epsilon_{k-1-i}^{\text{value}}\right\|_\infty + \gamma^{k-1}\|V^* - V^{\pi_0}\|_\infty.$$

Assuming that $\|\epsilon_i^{\text{value}}\|_\infty \leq \epsilon^{\text{value}}$ for all $i = 1, \dots, k-1$, we get

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma(1-\gamma^k)}{(1-\gamma)^2}\epsilon^{\text{value}} + \gamma^{k-1}\|V^* - V^{\pi_0}\|_\infty. \tag{F.8}$$

This shows that approximate PI has the transient behaviour of $O(\gamma^k)$, and it amplifies the PE error ϵ^{value} by a factor of $(1-\gamma)^{-2}$. This is the same as VI, and the comparison with OS-VI is exactly the same: whenever the model error is small enough, approximate OS-VI benefits from the approximate model $\hat{\mathcal{P}}$ and improves both the transient error rate and the error amplification.

The results of Munos [2003] does not consider the possibility of $\epsilon_i^{\text{policy}}$ being non-zero. For that, we report the asymptotic result of Proposition 6.2 Bertsekas and Tsitsiklis [1996], which states that

$$\limsup_{k\to\infty}\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma\epsilon^{\text{value}} + \epsilon^{\text{policy}}}{(1-\gamma)^2}, \tag{F.9}$$

Table 1: The transient and error amplification effects on $\|V^* - V^{\pi_k}\|_\infty$ for various method

Method	Transient Error	Error Amplification
VI (PE) (F.3)	$\gamma^k \|V^\pi - V_0\|_\infty$	$\frac{\epsilon^{\text{value}}}{1-\gamma}$
VI (Control) (F.6)	$\frac{2\gamma^k}{1-\gamma} \|V^* - V_0\|_\infty$	$\frac{2\gamma\epsilon^{\text{value}}}{(1-\gamma)^2}$
PI (F.8)-(F.9)	$\gamma^{k-1} \|V^* - V^{\pi_0}\|_\infty$	$\frac{2\gamma\epsilon^{\text{value}}+\epsilon^{\text{policy}}}{(1-\gamma)^2}$
MPI (F.10)	$\frac{2\gamma^k}{1-\gamma} \|V^* - V_0\|_\infty$	$\frac{2\gamma\epsilon^{\text{value}}+\epsilon^{\text{policy}}}{(1-\gamma)^2}$
OS-VI (PE) (F.4)	$\gamma'^k \|V^\pi - V_0\|_\infty$	$\frac{\epsilon^{\text{value}}}{1-\gamma'}$
OS-VI (Control) (F.7)	$\frac{2\gamma'^k}{1-\gamma'} \|V^* - V_0\|_\infty$	$\frac{2\gamma'\epsilon^{\text{value}}}{(1-\gamma')^2} + \frac{\|\epsilon_k^{\text{policy}}\|_\infty}{1-\gamma'}$

in which $\|\epsilon_i^{\text{policy}}\|_\infty \leq \epsilon^{\text{policy}}$ for all $i \geq 1$.

Convergence of modified policy iteration. Lemma 4 of Scherrer et al. [2015] leads to

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma(1-\gamma^{k-1})\epsilon^{\text{value}} + (1-\gamma^k)\epsilon^{\text{policy}}}{(1-\gamma)^2} + \frac{2\gamma^k}{1-\gamma}\|V^* - V_0\|_\infty. \tag{F.10}$$

The transient behaviour is $O(\gamma^k)$, and the error amplification is $(1-\gamma)^{-2}$ for both PE error ϵ^{value} and greedification error ϵ^{policy}. The comparison with OS-VI is as before, and shows that OS-VI can improve the convergence rate of the transient error as well as reducing the error amplification effect, if the model is accurate enough.

All these error bounds are summarized in Table 1 for ease of comparison. For the error amplification terms, we only consider the asymptotic behaviour by letting $k \to \infty$ to simplify the presentation.

F.2 Matrix splitting, Jacobi, and Gauss-Seidel iterations for dynamic programming

Kushner and Kleinman [1971] is one of the earliest paper we could find that mentions the Jacobi and Gauss-Seidel procedures for computing the value function. The focus of that work, however, is to propose accelerated variants of the Jacobi and Gauss-Seidel procedures through an over-relaxation procedure (cf. Section 3.1 of Varga 2000).

Bacon and Precup [2016] provide a matrix splitting perspective on planning with options. Their use of planning does not refer to the problem of Control (finding the optimal policy), but refers to the PE problem given a set of options that are consistent with the policy that is evaluated. They show that the computation of the value function using a given set of options can be interpreted as a particular choice of matrix splitting. The splitting depends on the dynamics, intra-option policies, the policy over options, and the termination probability of options. They show that decreasing the probability of termination, which corresponds to longer execution of options, leads to faster convergence of the planning. Although this is one of a few work that makes the connection between a dynamic programming-based approach and matrix splitting in numerical linear algebra explicit, it is fundamentally different from ours. They use matrix splitting to shed light on what planning with option does, but do not suggest a new algorithm. Their studied algorithm (VI-like procedures using options) does not benefit from the existence of an approximate $\hat{\mathcal{P}}$ to accelerate. The source of acceleration is the multi-step behaviour of an option. On a more detailed note, the matrix splitting in their work is of the *regular splitting* type, which has nice properties but is not suitable for the analysis of the splitting in this work.

The connection between multi-step models and matrix splitting is further developed in Chapter 4 of Bacon [2018]. He starts from the n-step model, and its corresponding Bellman-like equation for policy evaluation, which would be $V^\pi = \sum_{t=0}^{n-1}(\gamma\mathcal{P}^\pi)^t r^\pi + (\gamma\mathcal{P}^\pi)^n V^\pi$ (when $n = 1$, this is the usual Bellman equation). The value of n determines the number of unrolling steps. When n is randomly selected through a process that at each step decides whether to terminate or continue the unrolling with a probability determined by a function $\lambda : \mathcal{X} \times \mathcal{X} \to [0, 1]$, where $\lambda(x, x')$ depends on two consecutive states x and x', this leads to the so-called λ-models. This is closely related to the β-models of [Sutton, 1995]. A λ-model leads to a generalized Bellman equation. Bacon interprets the generalized Bellman equation as a particular choice of matrix splitting. The termination function

λ leads to a matrix splitting $M^\pi(\lambda)$ and $N^\pi(\lambda)$. This in turn determines the convergence rate of the iterative VI-like procedure for the computation of the value function, as the convergence rate depends on the spectral radius of $(M^\pi(\lambda)^{-1})N^\pi(\lambda)$. Similar remarks as the case of options applies: Bacon [2018, Chapter 4] sheds light to why already existing methods work, but it does not introduce a new algorithm; the analyzed algorithms do not benefit from an existence of an approximate model $\hat{\mathcal{P}}$.

Porteus [1975] propose several transformations to the reward and the probability transition matrix with the goal of improving the computational cost of solving the transformed MDP. One of the transformations, called *pre-inverse transform*, has some similarities with the operator splitting of this work. The end result, however, is different. That work considers a matrix W^π and define $\tilde{r}^\pi = (\mathbf{I} - W^\pi)^{-1} r^\pi$ and $\tilde{P}^\pi = (\mathbf{I} - W^\pi)^{-1}(\mathcal{P}^\pi - W^\pi)$. It requires that for any π, the matrix W^π be a lower triangular and be dominated by $\mathcal{P}^\pi$ as $0 \le W^\pi \le \mathcal{P}^\pi$ (W^π does not need to be a stochastic matrix). The paper then suggests performing one step of the Value Iteration as

$$V_k \leftarrow \operatorname*{argmax}_\pi (\mathbf{I} - W^\pi)^{-1}[r^\pi + (\mathcal{P}^\pi - W^\pi)V_{k-1}].$$

If W^π was $\hat{\mathcal{P}}^\pi$, this would be the same as (3.5). But we consider a probabilistic model $\hat{\mathcal{P}}^\pi$, which does not satisfy the setup of that work, including being a lower triangular or dominated by $\mathcal{P}^\pi$. That paper in fact considers W^π to be the lower triangular part of $\mathcal{P}^\pi$ (i.e., $[W^\pi]_{x,x'} = [\mathcal{P}^\pi]_{x,x'}$ for $1 \le x' \le x \le |\mathcal{X}|$) and zero otherwise), and then benefits from the lower triangularity of W^π to re-derive the Gauss-Seidel variant of VI.

Porteus referred to Varga [1962] to motivate another variant of pre-inverse transformation, in which W^π is not only dominated by $\mathcal{P}^\pi$, but also is diagonal. In that case, larger W^π leads to smaller spectral radius, which determines the convergence rate. If W^π is selected to be the diagonal part of $\mathcal{P}^\pi$ (i.e., $[W^\pi]_{x,x} = [\mathcal{P}^\pi]_{x,x}$, and zero for other elements), one retrieves the Jacobi variant of VI.

Although it is difficult to be sure why the condition $0 \le W^\pi \le \mathcal{P}^\pi$ was imposed, the paper's reference to Varga [1962] suggests that he was influenced by the concept of regular splitting, which is satisfied under the aforementioned condition.

References

Romina Abachi, Mohammad Ghavamzadeh, and Amir-massoud Farahmand. Policy-aware model learning for policy gradient methods. *arXiv:2003.00030v2*, 2020. 14

András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71:89–129, 2008. 6

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. 28

Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin F. Yang. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning (ICML)*, 2020. 14

Pierre-Luc Bacon. *Temporal Representation Learning*. PhD thesis, McGill University, 2018. 5, 31, 32

Pierre-Luc Bacon and Doina Precup. A matrix splitting perspective on planning with options. In *Continual Learning and Deep Networks Workshop at NIPS*. 2016. 5, 31

Dimitri P. Bertsekas and Steven E. Shreve. *Stochastic Optimal Control: The Discrete-Time Case*. Academic Press, 1978. 12

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996. 2, 12, 30

Shalabh Bhatnagar, Richard S. Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, 45(11):2471–2482, 2009. 22

Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019. 6

Pierluca D'Oro, Alberto Maria Metelli, Andrea Tirinzoni, Matteo Papini, and Marcello Restelli. Gradient-aware model-based policy search. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 14

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 6:503–556, 2005. 1

Jianqing Fan, Zhaoran Wang, Yuchen Xie, and Zhuoran Yang. A theoretical analysis of deep q-learning. *arXiv:1901.00137v3*, 2019. 6

Amir-massoud Farahmand. Iterative value-aware model learning. In *Advances in Neural Information Processing Systems (NeurIPS - 31)*, pages 9072–9083, 2018. 14

Amir-massoud Farahmand and Mohammad Ghavamzadeh. PID accelerated value iteration algorithm. In *International Conference on Machine Learning (ICML)*, 2021. 22, 23

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 (NeurIPS - 23)*, pages 568–576. 2010. 7

Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesvári, and Shie Mannor. Regularized policy iteration with nonparametric function spaces. *Journal of Machine Learning Research (JMLR)*, 17(139):1–66, 2016. 6

Amir-massoud Farahmand, André M.S. Barreto, and Daniel N. Nikovski. Value-aware loss function for model-based reinforcement learning. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1486–1494, April 2017. 14

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The John Hopkins University Press, 4th edition, 2013. 1, 3, 7, 13, 16

Geoffrey Gordon. Stable function approximation in dynamic programming. In *International Conference on Machine Learning (ICML)*, 1995. 1

Christopher Grimm, André M.S. Barreto, Satinder Singh, and David Silver. The value equivalence principle for model-based reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 14

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

Joshua Joseph, Alborz Geramifard, John W. Roberts, Jonathan P. How, and Nicholas Roy. Reinforcement learning with misspecified model classes. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 939–946. IEEE, 2013. 14

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML)*, pages 267–274, 2002. 7

Harold J. Kushner and Allan J. Kleinman. Accelerated procedures for the solution of discrete Markov control problems. *IEEE Transactions on Automatic Control*, 16(2):147–152, April 1971. 5, 31

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In *Conference on Learning for Dynamics and Control*, 2020. 14

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. 1

Rémi Munos. Error bounds for approximate policy iteration. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 560–567, 2003. 6, 7, 30

Rémi Munos. Performance bounds in L_p norm for approximate value iteration. *SIAM Journal on Control and Optimization*, pages 541–561, 2007. 6, 7, 8, 29

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research (JMLR)*, 9:815–857, 2008. 1, 6

Evgenii Nikishin, Romina Abachi, Rishabh Agarwal, and Pierre-Luc Bacon. Control-oriented model-based reinforcement learning with implicit differentiation. In *AAAI Conference on Artificial Intelligence*, 2022. 14

Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network. In *Advances in Neural Information Processing Systems (NIPS - 30)*, pages 6118–6128. Curran Associates, Inc., 2017. 14

Evan L. Porteus. Bounds and transformations for discounted finite markov decision chains. *Operations Research*, 23(4):761–784, 1975. 5, 32

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics (SIAM), 2nd edition, 2003. 1, 3

Bruno Scherrer, Mohammad Ghavamzadeh, Victor Gabillon, Boris Lesner, and Matthieu Geist. Approximate modified policy iteration and its application to the game of tetris. *Journal of Machine Learning Research (JMLR)*, 16(49):1629–1676, 2015. 6, 7, 31

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering Atari, Go, Chess and Shogi by planning with a learned model. *Nature*, 588, 2020. 14

David Silver, Hado van Hasselt, Matteo Hessel, Tom Schaul, Arthur Guez, Tim Harley, Gabriel Dulac-Arnold, David Reichert, Neil Rabinowitz, André M.S. Barreto, and Thomas Degris. The predictron: End-to-end learning and planning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3191–3199, 2017. 14

Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer, 2008. 6

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

Richard S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the 7th International Conference on Machine Learning (ICML)*, 1990. 2

Richard S. Sutton. TD models: Modeling the world at a mixture of time scales. In *International Conference on Machine Learning (ICML)*, pages 531–539. Elsevier, 1995. 31

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2019. 2, 12, 26

Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Morgan Claypool Publishers, 2010. 2, 12

Richard S. Varga. *Matrix Iterative analysis*. Springer, 1962. 32

Richard S. Varga. *Matrix Iterative Analysis*. Springer-Verlag, 2nd edition, 2000. 1, 31

Claas A Voelcker, Victor Liao, Animesh Garg, and Amir massoud Farahmand. Value gradient weighted model-based reinforcement learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=4-D6CZkRXxI`. 14

Martin J. Wainwright. Stochastic approximation with cone-contractive operators: Sharp ∞-bounds for q-learning. *ArXiv*, abs/1905.06265, 2019. 25, 26

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] In the end of Section 4 we show that the convergence and speed up need an accurate enough model.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] There are no immediate potential negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The details are in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The details are in the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A] The experiments are simple and can be run in personal systems.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]