Source Traces for Temporal Difference Learning

Silviu Pitis
Georgia Institute of Technology
Atlanta, GA, USA 30332
spitis@gatech.edu

Abstract

This paper motivates and develops source traces for temporal difference (TD) learning in the tabular setting. Source traces are like eligibility traces, but model potential histories rather than immediate ones. This allows TD errors to be propagated to potential causal states and leads to faster generalization. Source traces can be thought of as the model-based, backward view of successor representations (SR), and share many of the same benefits. This view, however, suggests several new ideas. First, a TD(λ)-like source learning algorithm is proposed and its convergence is proven. Then, a novel algorithm for learning the source map (or SR matrix) is developed and shown to outperform the previous algorithm. Finally, various approaches to using the source/SR model are explored, and it is shown that source traces can be effectively combined with other model-based methods like Dyna and experience replay.

1 Introduction

When we, as humans, first learn that some state of the world is valuable (or harmful), we adjust our behavior. For instance, suppose we burn our hand on a hot pan. At least three types of generalization enable useful learning. First, we generalize to substantially similar causal states: we will be more careful handling hot pans on Tuesdays, even if the burn happened on a Monday. Second, we generalize back through time from direct causes to indirect ones. Rather than simply stopping at the last second, we may now instinctively reach for an oven mitt before approaching the pan. Finally, we generalize not only to actual causes, but also to *potential* causes. Our burn may make us less willing to keep our hand close to a campfire while roasting marshmallows.

Reinforcement learning mirrors human learning in many respects, including the three modes of generalization just discussed. The first—generalizing to substantially similar causes—is a direct result of function approximation (e.g., Mnih et al. 2015). The latter two—generalization to indirect causes and generalization to potential causes—though aided by function approximation, are driven primarily by temporal difference (TD) learning (Sutton 1988). TD learning uses a value function to bootstrap learning, allowing new information to flow backward, state-by-state, from effects to causes. The speed of one-step TD methods is limited, however, as they require many repeat experiences to generalize.

Several families of methods achieve faster generalization by propagating TD errors several steps per real experience. Eligibility trace methods keep a short-term memory vector that accumulates the (decaying) "eligibilities" of recently visited states, and use it to propagate TD errors to past states in proportion to their eligibility (Sutton 1988). Experience replay (ER) (Lin 1992) and Dyna (Sutton 1990) methods learn from replayed or generated experiences, so that several value function backups occur per real experience. Unlike eligibility traces, ER and Dyna employ a long-term model to speed up learning with respect to known potential causes.

This paper introduces source traces. Source traces are like eligibility traces, but model potential histories rather than immediate ones. This allows for propagation of TD errors to potential causal states and leads to faster learning. Source traces can be thought of as the model-based, backward view of successor representations (SR) (Dayan 1993). This backward view suggests several new ideas: a provably-convergent TD(λ)-like algorithm for learning and the concept of partial source traces and SR (Section 3), a novel algorithm for learning the source map (Section 4), the triple model learning algorithm (Section 4), and new perspectives on the source map, which allow, among other things, the use of source traces to enhance experience replay (Section 5).

For clarity and brevity, and as a building block to future work, this paper focuses on the tabular MRP-valuation setting. It is likely, however, given existing extensions of SR (e.g., Barreto et al. 2016), that source traces can be effectively extended to control settings requiring approximation, which is discussed briefly in the conclusion (Section 6).

2 Ideal Source Traces

We consider an n-state discounted Markov Reward Process (MRP) with infinite time horizon, formally described as the tuple (S, p, r, γ), where: S is the state space; p is the transition function, which takes two states, $s, s' \in S$, and returns the probability of transitioning to s' from s; r is the reward function, which takes a state, s, and returns a real-valued reward for that state; and $\gamma \in [0, 1)$ is the discount factor. If the size of the state space, $|S|$, is finite, then p can be described by the $|S|\times|S|$ probability matrix $\mathbf{P}$ whose ij-th entry is $p(s_i, s_j)$. Similarly, r can be described by the $|S|$-

dimensional vector $\mathbf{r}$ whose i-th entry is $\mathbb{E}\{r(s_i)\}$. We seek the value of each state, $v(s)$ for $s \in S$, which is equal to the sum of expected discounted future rewards, starting in state s. Letting $\mathbf{v}$ be the vector whose i-th entry is $v(s_i)$:

$$\begin{aligned}\mathbf{v} &= \mathbf{r} + \gamma\mathbf{P}\mathbf{r} + \gamma^2\mathbf{P}^2\mathbf{r}\dots \\ &= \left(\mathbf{I} + \gamma\mathbf{P} + \gamma^2\mathbf{P}^2\dots\right)\mathbf{r} \\ &= \left(\mathbf{I} - \gamma\mathbf{P}\right)^{-1}\mathbf{r}\end{aligned} \tag{1}$$

where the third equality holds because $\gamma < 1$ and $\mathbf{P}$ is a transition matrix, so that $\|\gamma\mathbf{P}\| < 1$ and the well known matrix identity applies (see, e.g., Theorem A.1 of Sutton 1988). Alternatively, note that $\mathbf{v} = \mathbf{r} + \gamma\mathbf{P}\mathbf{v}$, from which the result follows by arithmetic, given that $\mathbf{I} - \gamma\mathbf{P}$ is invertible.

In practice, $\mathbf{v}$ is often solved for iteratively, rather than by inverting $\mathbf{I} - \gamma\mathbf{P}$ (see Section 4.1 of Sutton and Barto 1998). For the moment, however, let us assume that we already have the *source map* $\mathbf{S} = (\mathbf{I} - \gamma\mathbf{P})^{-1}$. Then, given any $\mathbf{r}$, we can solve for $\mathbf{v}$ in a single matrix multiplication.

We define the *ideal source trace* for state s_j as the j-th column of $\mathbf{S}$, $[\mathbf{S}]_{\cdot j}$. If we have already computed $\mathbf{v}$ for some $\mathbf{r}$, and $\mathbb{E}\{r(s_j)\}$ changes, we can use the source trace to precisely propagate the expected reward delta, $\Delta r_j = \mathbb{E}\{r'(s_j)\} - \mathbb{E}\{r(s_j)\}$, and update $\mathbf{v}$ in $O(|S|)$ time:

$$\mathbf{v}' = \mathbf{v} + \Delta r_j [\mathbf{S}]_{\cdot j}. \tag{2}$$

This "source backup" operation is easily extended to *expected* temporal differences. Given some model $\mathbf{v}_0$:

$$\begin{aligned}\mathbf{v} &= \mathbf{S}\mathbf{r} + \mathbf{v}_0 - \mathbf{v}_0 \\ &= \mathbf{v}_0 + \mathbf{S}\left(\mathbf{r} - \mathbf{S}^{-1}\mathbf{v}_0\right) \\ &= \mathbf{v}_0 + \mathbf{S}\left(\mathbf{r} + \gamma\mathbf{P}\mathbf{v}_0 - \mathbf{v}_0\right).\end{aligned} \tag{3}$$

In words: to arrive at the correct value of $\mathbf{v}$ from $\mathbf{v}_0$, the expected temporal difference upon leaving each state s_j, $[\mathbf{r} + \gamma\mathbf{P}\mathbf{v}_0 - \mathbf{v}_0]_{j\cdot}$, needs to be added to $\mathbf{v}_0$ in proportion to the values of $[\mathbf{S}]_{\cdot j}$, the ideal source trace for s_j.

Related Work—SR and LSTD

The source map is well known and goes by several names. When speaking of undiscounted absorbing Markov chains, the source map $(\mathbf{I}-\mathbf{P})^{-1}$ is known as the *fundamental matrix* (Kemeny and Snell 1976). In the RL setting, it is equivalent to the matrix of *successor representations* (Dayan 1993).

Whereas source traces look *backward* to causes, successor representations (SR) look *forward* to effects: the SR of state s_i, defined as the i-th *row* of $\mathbf{S}$, is a $|S|$-dimensional vector whose j-th element can be understood as the cumulative discounted expected number of visits to s_j when starting in s_i. While TD learning with source traces updates $\mathbf{v}_0$ directly, TD learning with SR learns the model $\mathbf{r}_0$ and uses it to compute $v_0(s_i)$ as the dot product $[\mathbf{S}_0]_{i\cdot}\mathbf{r}_0$ (models are denoted with the subscript 0 throughout). The two perspectives are illustrated in figure 1. Though different in motivation and interpretation, source traces and successor representations share the matrix $\mathbf{S}$ and are useful for similar reasons.

Much like the backward and forward views of TD(λ) offer different insights (Sutton and Barto 1998), so too do the backward view (source traces) and the forward view (SR) here. For instance, the forward view suggests the following useful characterization: we can think of $\mathbf{S}_{ij}$ as the solution $v(s_i)$ of a derivative MRP with the same dynamics as the underlying MRP, but with zero reward everywhere *except* state s_j, where the reward is 1. This suggests that source traces may be found by TD learning as the solutions to a set of derivative MRPs (one for each state). This algorithm was proposed by Dayan 1993. By contrast, the backward view prompts an alternative algorithm based on TD(λ) that offers a faster initial learning curve at the price of increased variance. We combine the two algorithms in Section 4 to get the best of both worlds. This shift in perspective produces several other novel ideas, which we explore below.

Source learning is also related to Least-Squares TD (LSTD) methods (Bradtke and Barto 1996). In the tabular case, LSTD models $\mathbf{S}^{-1}$ (i.e., $\mathbf{I} - \gamma\mathbf{P}$) and $\mathbf{r}$ (or scalar multiples thereof), and computes $\mathbf{v}_0$ as $(\mathbf{S}_0^{-1})^{-1}\mathbf{r}_0$. Incremental LSTD (iLSTD) (Geramifard, Bowling, and Sutton 2006) is similar, but learns $\mathbf{v}_0$ incrementally, thus avoiding matrix inversion. Finally, recursive LSTD (RLS TD) (Bradtke and Barto 1996) computes $\mathbf{v}_0$ as does LSTD, but does so by recursive least squares instead of matrix inversion. Of these algorithms, RLS TD—despite being an algorithmic trick—is closest in spirit to source learning; in the tabular case, equation 15 of Bradtke and Barto and equation 4 (below) have comparable structure. However, RLS TD maintains its estimate of $\mathbf{v}$ precisely using $O(|S|^2)$ time per step, whereas the TD Source algorithm presented learns $\mathbf{S}_0$ and $\mathbf{v}_0$ using $O(|S|)$ time per step. As iLSTD has the same time complexity as source learning in the tabular case, we compare them empirically in Section 4.

3 Convergence

Equation 3 shows that a synchronous source backup of all expected TD errors converges after a single iteration. But what about in the reinforcement learning setting, where rewards and transitions are sampled asynchronously? And what if our model of $\mathbf{S}$ is only approximate?

Unlike the Q-learning backup operator, the asynchronous source backup operator is not generally a contraction in any weighted maximum norm of the value space, even with known $\mathbf{S}$. As an example, consider the two state MRP defined by $\mathbf{r} = [0, 0]^T$, $\mathbf{P} = [[0.5, 0.5], [0.5, 0.5]]$, and $\gamma = 0.5$, so that $\mathbf{S} = [[1.5, 0.5], [0.5, 1.5]]$. The optimal $\mathbf{v}$ is clearly $[0, 0]^T$, but starting at $\mathbf{v}_0 = [a, -a]^T$, the asynchronous backup from the first state (state i) results in a TD error of $-a$, so that $\mathbf{v}_1 = \mathbf{v}_0 + [\mathbf{S}]_{\cdot i}\left([r]_i + \gamma[\mathbf{P}]_{\cdot i}^T\mathbf{v}_0 - [\mathbf{v}_0]_i\right) = [a, -a]^T - a[1.5, 0.5]^T = [-0.5a, -1.5a]^T$. It is easy to see that the value of state j has been pushed away from its optimal value by $0.5a$: $|[\mathbf{v}_1 - \mathbf{v}]_j| > |[\mathbf{v}_0 - \mathbf{v}]_j|$. By symmetry, the same expansion in $v(s_i)$ occurs for the asynchronous backup from state j. Since a is arbitrary, the conclusion follows.

Nevertheless, by working in reward space, instead of value space, proof of convergence is reduced to a straightforward application of the general convergence theorem of Jaakkola, Jordan, and Singh 1994. Notably, convergence is guaranteed even with sufficiently approximated $\mathbf{S}$.

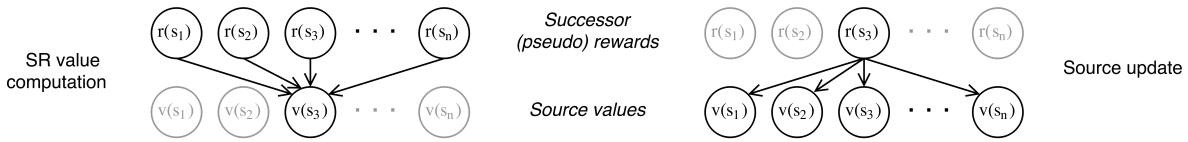

Figure 1: SR value computation (left) vs source update (right). When using the SR, one computes $v_0(s_i)$ (as $[\mathbf{S}_0]_{i\cdot}\mathbf{r}_0$) by looking *forward* to the (pseudo) reward values of its successors, which are learned according to equation 6 or equation 12. With source traces, one updates $\mathbf{v}_0$ directly by distributing the (pseudo) reward delta at state s_j *backward* to its sources in proportion to $[\mathbf{S}_0]_{\cdot j}$ (equation 4). Given a fixed source map, it follows from the derivation of equation 6 that using the SR with equation 6 and source learning with equation 4 are equivalent. The difference that arises when the source map changes during learning is explored in Section 4—Direct Learning vs SR-Reward Decomposition.

Theorem. The sampled, asynchronous source learning algorithm given by:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \alpha_n[\mathbf{S}_0]_{\cdot\langle n\rangle}\Big(r(s_{\langle n\rangle}) + \gamma v_n(s_{\langle n+1\rangle}) - v_n(s_{\langle n\rangle})\Big) \quad (4)$$

where subscript $\langle n\rangle$ denotes the index corresponding to the state at time n and $\mathbf{S}_0$ is an approximation of $\mathbf{S}$, converges to the true $\mathbf{v}$ with probability 1 if:

(a) the state space is finite,

(b) $\forall$ state indices x, $\sum_{\langle n\rangle=x} \alpha_n = \infty$ and $\sum_{\langle n\rangle=x} \alpha_n^2 < \infty$,

(c) $\mathrm{Var}\{r(s_i)\}$ is bounded for all i, and

(d) $\|(\mathbf{I} - \mathbf{S}^{-1}\mathbf{S}_0)\| \leq \gamma$.

Proof. Let $\mathbf{d}_n$ and $\boldsymbol{\alpha}_n$ be $|S|$-dimensional vectors, where the i-th slot of $\mathbf{d}_n$ corresponds to a sampled TD error starting in state s_i, $r(s_i) + \gamma v_n(s_{next(i)}) - v_n(s_i)$ (all samples but the one corresponding to $s_{\langle n\rangle}$ are hypothetical), and $\boldsymbol{\alpha}_n$ is zero everywhere except in the position corresponding to $s_{\langle n\rangle}$, where it is α_n. Then, we can rewrite equation 4 as:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathbf{S}_0(\boldsymbol{\alpha}_n \odot \mathbf{d}_n) \quad (5)$$

where $\odot$ is the element-wise product.

Since $\|(\mathbf{I} - \mathbf{S}^{-1}\mathbf{S}_0)\| \leq \gamma < 1$, the matrix identity used in equation 1 above implies that $\mathbf{S}^{-1}\mathbf{S}_0$ is invertible, so that $\mathbf{S}_0$ is invertible. Letting $\mathbf{r}_n = \mathbf{S}_0^{-1}\mathbf{v}_n$ and left-multiplying both sides of (5) by $\mathbf{S}_0^{-1}$ we obtain the following equivalent update in (pseudo) reward space:

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \boldsymbol{\alpha}_n \odot \mathbf{d}_n. \quad (6)$$

Subtracting $\mathbf{S}_0^{-1}\mathbf{S}\mathbf{r}$ from both sides and rearranging, we obtain the iterative process $\{\boldsymbol{\Delta}_n\}$:

$$\boldsymbol{\Delta}_{n+1} = (1 - \boldsymbol{\alpha}_n) \odot \boldsymbol{\Delta}_n + \boldsymbol{\alpha}_n \odot \mathbf{F}_n \quad (7)$$

where $\boldsymbol{\Delta}_n = \mathbf{r}_n - \mathbf{S}_0^{-1}\mathbf{S}\mathbf{r}$ and $\mathbf{F}_n = \mathbf{d}_n + \boldsymbol{\Delta}_n$. To this process, we apply Theorem 1 of Jaakkola, Jordan, and Singh 1994. Conditions (1) and (2) of Jaakkola, Jordan, and Singh are satisfied by the assumptions above. Condition (3) is satisfied because:

$$\begin{aligned} \|\mathbb{E}\{\mathbf{F}_n\}\| &= \|\mathbf{r} - \mathbf{S}^{-1}\mathbf{S}_0\mathbf{r}_n + \boldsymbol{\Delta}_n\| \\ &= \|(\mathbf{I} - \mathbf{S}^{-1}\mathbf{S}_0)\boldsymbol{\Delta}_n\| \\ &\leq \gamma\|\boldsymbol{\Delta}_n\|. \end{aligned} \quad (8)$$

Condition (4) is satisfied because $[\mathbf{F}_n]_k$ depends at most linearly on $[\mathbf{r}_n]_k$ and $\mathrm{Var}\{r(s_k)\}$ is bounded by assumption. Thus, Theorem 1 of Jaakkola, Jordan, and Singh applies, $\{\boldsymbol{\Delta}_n\}$ converges to $\mathbf{0}$, and so too does $\{\mathbf{v}_n - \mathbf{v}\}$.

Partial Source Traces and Speed of Convergence

The theorem admits a variety of approximate models, so long as $\|(\mathbf{I} - \mathbf{S}^{-1}\mathbf{S}_0)\| \leq \gamma$. One such family is that of n-*step* source traces (cf. the n-step methods of Sutton and Barto 1998), derived from the source maps $\mathbf{S}_n = \sum_{k=0}^{n-1}(\gamma\mathbf{P})^k$ for $n \geq 1$. Using $\mathbf{S}_1 = \mathbf{I}$ corresponds to TD(0), so that $n > 1$ interpolates between TD(0) and full source learning. Another way to interpolate between TD(0) and full source learning is to introduce a parameter $\lambda \in [0, 1]$ (cf. λ in TD(λ)) and use λ source traces, derived from $\mathbf{S}^\lambda = \sum_{k=0}^{\infty}(\gamma\lambda\mathbf{P})^k$.

We may combine $\mathbf{S}_n$ and $\mathbf{S}^\lambda$ to form:

$$\mathbf{S}_n^\lambda = \sum_{k=0}^{n-1}(\gamma\lambda\mathbf{P})^k. \quad (9)$$

This gives us the more general family of *partial* source traces, for which convergence holds at all λ and n. The case $n = 1$ is easy to verify. For $n > 1$, we have:

$$\begin{aligned} \|\mathbf{I} - \mathbf{S}^{-1}\mathbf{S}_n^\lambda\| &= \|\mathbf{I} - (\mathbf{I} - \gamma\mathbf{P})\sum_{k=1}^{n}(\gamma\lambda\mathbf{P})^{k-1}\| \\ &= \|(1-\lambda)\gamma\mathbf{P}\sum_{k=0}^{n-2}(\lambda\gamma\mathbf{P})^k + \lambda^{n-1}\gamma^n\mathbf{P}^n\| \\ &\leq \gamma\left[(1-\lambda)\sum_{k=0}^{n-2}(\lambda\gamma)^k + (\lambda\gamma)^{n-1}\right] \\ &= \gamma\left[(1-\lambda\gamma)\sum_{k=0}^{n-2}(\lambda\gamma)^k + (\lambda\gamma-\lambda)\sum_{k=0}^{n-2}(\lambda\gamma)^k + (\lambda\gamma)^{n-1}\right] \\ &= \gamma\left[1 - \lambda(1-\gamma)\sum_{k=0}^{n-2}(\lambda\gamma)^k\right] < \gamma \end{aligned} \quad (10)$$

where the first inequality holds by a combination of the triangle inequality, the identity $\|a\mathbf{A}\| \leq |a|\|\mathbf{A}\|$, and the fact that all terms are positive. The bracketed quantity on the final line is in $[0, 1]$ and strictly decreasing in both n and λ, which suggests that the speed of convergence increases as we move from TD(0) (either $n = 1$ or $\lambda = 0$) to full source traces ($n = \infty$ and $\lambda = 1$).

Experiments in two environments confirm this, with an important caveat. The first environment is a Random MRP with 100 states, 5 random transitions per state with transition probabilities sampled from $\mathcal{U}(0, 1)$ and then normalized

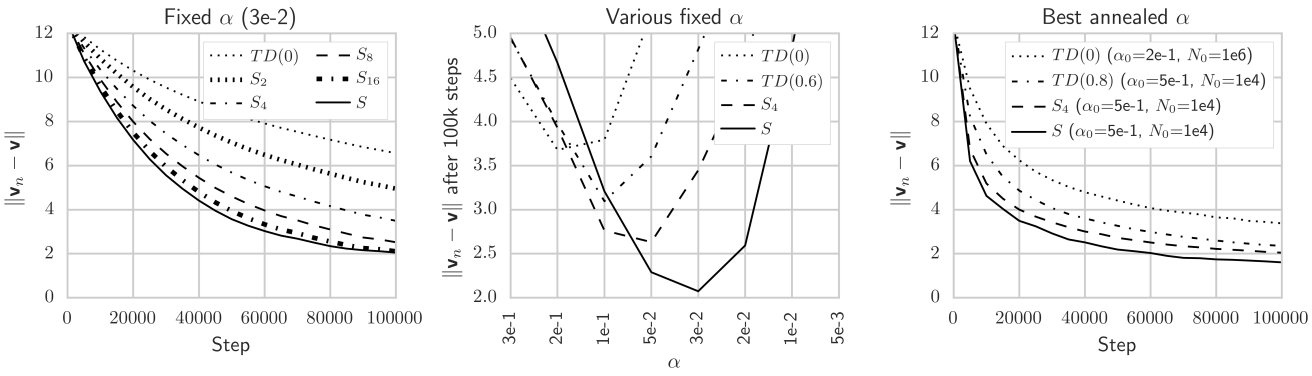

Figure 2: Convergence in 3D Gridworld using known partial and full source traces for various learning rate schedules.

(transitions are re-sampled if the resulting $\mathbf{P}$ is not invertible), a random reward for each state sampled from $\mathcal{N}(0,1)$, and $\gamma = 0.9$. The second is a 1000-state 3D Gridworld with wraparound: states are arranged in a $10\times10\times10$ 3D grid, with state transitions between adjacent states (6 per state, including edge states, which wraparound). Transition probabilities are sampled from $\mathcal{U}(0,1)$ and then normalized. 50 random states are assigned a random reward sampled from $\mathcal{N}(0,1)$ and γ is set to 0.95.

Unless otherwise noted, similar results held in both environments, here and throughout. All experiments, unless otherwise noted, reflect average results on 30 Random MRP or 3D Gridworld environments. The same set of randomly generated environments was used when comparing algorithms.

Figure 2 (left), which reflects the 3D Gridworld, plots learning curves after 100,000 steps for the source learning algorithm given by equation 4 for $\mathbf{S}_1$ (TD(0)) through $\mathbf{S}_4$ and $\mathbf{S}$. A similar pattern appears for $\mathbf{S}^\lambda$ with increasing λ. In each case, $\mathbf{v}_0$ was initialized to $\mathbf{0}$, and $\|\mathbf{v}_n - \mathbf{v}\|$ was averaged across the MRPs ($\mathbf{v}$ was computed by matrix inversion).

The curves in figure 2 (left) are not representative of all learning rates. Figure 2 (center) shows the final error achieved by TD(0), TD(λ) at the best λ (tested in 0.1 increments), $\mathbf{S}_4$ and $\mathbf{S}$ at various fixed α. Although the advantage of full source traces is clear, both TD(λ) and partial source learning do better at higher αs. In the Random MRP environment (not shown), partial traces provide the best results after only 50,000 steps and experiments suggest that TD(0) eventually overtakes full source learning at *all* fixed α.

This highlights a weakness of using source traces: they amplify the per-step transition variance by propagating it multiple steps. We propose two strategies for managing this. The first is to use a forward model, which we explore in the "triple model learning" algorithm of Section 4. The second is to anneal the learning rate. For consistent comparison with iLSTD in Section 4, we tested the following set of annealing schedules, adapted from Geramifard et al. 2007:

$$\alpha_n = \alpha_0(N_0 + 1)/(N_0 + n^{1.1}) \tag{11}$$

for $\alpha \in \{$5e-1, 2e-1, 1e-1, 5e-2, 2e-2, 1e-2, 5e-3$\}$ and $N_0 \in \{0,$ 1e2, 1e4, 1e6$\}$. Figure 2 (right) plots the best results for TD(0), TD(λ) at the best λ, $\mathbf{S}_4$ and $\mathbf{S}$ when annealing the learning rate, which improved both the convergence speed and final results for all methods.

Relating the Convergence Results to SR

It is tempting to interpret the convergence theorem in terms of SR by noting that $\mathbf{S}_0$ maps to the SR of a discrete state space ($\mathbf{S}_0 \Leftrightarrow \mathbf{\Phi}$), so that the pseudo-reward space of equation 6 maps to the weight space when using the SR with linear value function approximation ($\mathbf{r}_n \Leftrightarrow \boldsymbol{\theta}_n$). NB, however, that equation 6 updates only a single element of $\mathbf{r}_n$, whereas semi-gradient TD(0) over $\boldsymbol{\theta}$, which was the rule used in Dayan 1993 and which we term *pseudo-reward descent*, updates (potentially) all elements of $\boldsymbol{\theta}_n$ with the rule:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n + \alpha_n \delta_n \boldsymbol{\phi}(s_{\langle n \rangle}) \tag{12}$$

where $\delta_n = [\mathbf{d}_n]_{\langle n \rangle}$ is the temporal difference at step n and $\boldsymbol{\phi}(s_{\langle n \rangle})$ is the SR of $s_{\langle n \rangle}$. Although the convergence of (12) does not follow from our theorem, it is guaranteed, under slightly different conditions, by the theorem of Tsitsiklis and Van Roy 1997. A notable difference is that the theorem of Tsitsiklis and Van Roy requires updates to be made according to the invariant distribution of the MRP. We compare the performance of these two update rules in Section 4.

Setting aside the above differences, we note that $\mathbf{S}_n^\lambda$ may be used to define partial successor representations (i.e., SR with n-step or decaying lookahead) and conjecture that the speed of pseudo-reward descent will increase with n and λ, as it did for source learning. This idea can be related to the work of Gehring 2015, which shows that using low-rank approximations of $\mathbf{S}$ can obtain good results. Note that Gehring's low-rank approximations are quite different from the invertible approximations presented here and that the two methods for approximation might be combined.

4 Learning and Using the Source Map

Source traces are not useful without a practical method for computing them. In this section we draw inspiration from TD(λ) to show that the source model can be learned. While

our convergence theorem does not formally apply to this scenario (since $\mathbf{S}_0$ is changing), we conjecture that convergence will occur if the changes to $\mathbf{S}_0$ are sufficiently "small".

The Link Between Source and Eligibility Traces

The TD(λ) algorithm (Sutton 1988), defined by the update:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \alpha_n \mathbf{e}_n \Big(r(s_{\langle n \rangle}) + \gamma v_n(s_{\langle n+1 \rangle}) - v_n(s_{\langle n \rangle}) \Big) \quad (13)$$

propagates the TD error of each experience according to the *eligibility trace* at time n, $\mathbf{e}_n$, defined recursively as:

$$\mathbf{e}_n = [\mathbf{I}]_{\cdot \langle n \rangle} + \gamma \lambda \mathbf{e}_{n-1} \quad (14)$$

where subscript $\langle n \rangle$ denotes the index corresponding to $s_{\langle n \rangle}$.

It notable that, but for replacing the current eligibility trace with the source trace for the current state, the asynchronous source learning update (equation 4) is identical to the TD(λ) update (equation 13). This suggests a link between source traces and eligibility traces.

Indeed, it is *almost* correct to suppose that the average eligibility trace given that the current state is s_j is the source trace for s_j. But recall from equation 1 that $\mathbf{S} = \mathbf{I} + \gamma \mathbf{P} + \gamma^2 \mathbf{P}^2 \cdots = \mathbf{I} + \gamma \mathbf{P} \mathbf{S}$, so that the source trace for s_j is:

$$[\mathbf{S}]_{\cdot j} = [\mathbf{I}]_{\cdot j} + \gamma \sum_i p(s_i, s_j) [\mathbf{S}]_{\cdot i} \quad (15)$$

In contrast, averaging the eligibility traces for state s_j would reverse the roles of s_j and s_i; letting $\bar{\mathbf{e}}_j$ represent the average eligibility trace in state s_j, it is easy to see that:

$$\begin{aligned} \bar{\mathbf{e}}_j &= \mathbb{E}\{[\mathbf{I}]_{\cdot j} + \gamma \lambda \mathbf{e}_i\} \\ &= [\mathbf{I}]_{\cdot j} + \gamma \lambda \sum_i P(s_i | s_j) \bar{\mathbf{e}}_i. \end{aligned} \quad (16)$$

Importance sampling at each step of the accumulation corrects for this and turns what would otherwise be the average eligibility trace into an estimator of the source trace. It is reflected in the algorithm below (line 12).

The Tabular TD Source Algorithm

The TD Source algorithm is similar to TD(λ), except that it uses the observed history to update the model $\mathbf{S}_0$ rather than to directly distribute TD errors. This adds extra computation per step, and requires storing an $|S| \times |S|$ matrix in memory. Line 12 of the algorithm corresponds to equation 15 above. β is the learning rate used in the stochastic approximation of $\mathbf{S}$ on line 13, and may be fixed or annealed according to some schedule. Lines 14 and 15 are commented out and are not part of the TD Source algorithm (see next subsection).

The λ parameter is included should partial source traces be more desirable than full ones (e.g., because they propagate less variance, or because they are easier to approximate). One might also consider annealing λ to 1 in order to minimize the effect of inaccuracies during learning whilst still obtaining a full source model, which we investigate briefly in the next subsection.

Algorithm 1 Tabular TD learning with source traces

1: **procedure** TD SOURCE[-SR](*episodes*, γ, λ, α, β)
2: $\quad$ $\mathbf{v} \leftarrow \mathbf{0}$
3: $\quad$ $\mathbf{S} \leftarrow \mathbf{I}$ ($|S| \times |S|$ identity matrix)
4: $\quad$ $\mathbf{c} \leftarrow \mathbf{0}$ (counts vector)
5: $\quad$ **for** episode in 1 .. *n* **do**
6: $\quad\quad$ $j \leftarrow$ index of initial state
7: $\quad\quad$ $c(s_j) \leftarrow c(s_j) + 1$
8: $\quad\quad$ $[\mathbf{S}]_{\cdot j} \leftarrow (1 - \beta)[\mathbf{S}]_{\cdot j} + \beta [\mathbf{I}]_{\cdot j}$
9: $\quad\quad$ **for** pair (s_i, s_j) and reward r in episode **do**
10: $\quad\quad\quad$ $\mathbf{v} \leftarrow \mathbf{v} + \alpha [\mathbf{S}]_{\cdot i} (r + \gamma v_j - v_i)$
11: $\quad\quad\quad$ $c(s_j) \leftarrow c(s_j) + 1$
12: $\quad\quad\quad$ $\mathbf{e} \leftarrow [\mathbf{I}]_{\cdot j} + \gamma \lambda \frac{c(s_j)}{c(s_i)} [\mathbf{S}]_{\cdot i}$
13: $\quad\quad\quad$ $[\mathbf{S}]_{\cdot j} \leftarrow (1 - \beta)[\mathbf{S}]_{\cdot j} + \beta \mathbf{e}$
$\quad\quad\quad$ /*
14: $\quad\quad\quad$ $\mathbf{e} \leftarrow [\mathbf{I}]_{i \cdot} + \gamma \lambda [\mathbf{S}]_{j \cdot}$ ▷ TD SR
15: $\quad\quad\quad$ $[\mathbf{S}]_{i \cdot} \leftarrow (1 - \beta)[\mathbf{S}]_{i \cdot} + \beta \mathbf{e}$ ▷ TD SR
$\quad\quad\quad$ */
16: **return** $\mathbf{v}$, $\mathbf{S}$, $\mathbf{c}$

TD Source vs Dayan's Algorithm ("TD SR")

As noted in Section 2, the source map is equivalent to the matrix of successor representations, for which there is an existing learning algorithm due to Dayan 1993 (when combined with equation 4, "TD SR"). Whereas TD Source learns the source map using the column-wise recurrence of equation 15, TD SR uses the row-wise recurrence:

$$[\mathbf{S}]_{i \cdot} = [\mathbf{I}]_{i \cdot} + \gamma \sum_j p(s_i, s_j) [\mathbf{S}]_{j \cdot}. \quad (17)$$

TD SR uses lines 14 and 15 of the algorithm in place of lines 12 and 13. And since it does not require importance sampling, lines 4, 6, 7, 8 and 11 can be removed.

As importance sampling often increases variance and slows learning, we might expect TD SR to outperform TD Source. However, the column-wise algorithm has one major advantage: it immediately puts to use the most recent information. For this reason, we should expect TD Source to outperform TD SR algorithm at the start of learning.

Experiment confirms this hypothesis. Figure 3 tracks the quality of the approximation over the first 50,000 steps of the Random MRP environment. Four different algorithms were run at various minimum learning rates. The learning curves at the best learning rate (in terms of final error) are shown. For all algorithms, α was annealed to the minimum, on a per-state basis according to the harmonic series, as this performed better in all cases than using fixed α. Error was computed as $\sqrt{\sum ([\mathbf{S}_0 - \mathbf{S}]_{ij})^2}$.

The results suggest that we might improve upon TD Source by either (1) finding a better method for annealing α in response to large importance sampling ratios, or (2) switching to TD SR after an initial training period (neither shown). Alternatively, we might combine TD Source and TD SR and update both $[\mathbf{S}]_{\cdot j}$ and $[\mathbf{S}]_{i \cdot}$ at each step by using lines 12-15 simultaneously ("TD Source-SR"). This outperforms either algorithm individually by a sizable mar-

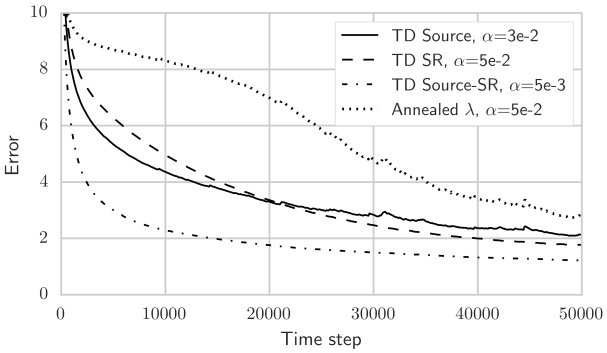

Figure 3: TD Source vs TD SR on Random MRPs

gin. To check that this is not simply coincidence resulting from a higher effective learning rate, we tested several other learning rate schedules for both TD Source and TD SR individually—none were as effective as TD Source-SR.

Finally, we tested the proposition in the previous subsection that annealing λ to 1 might allow for better approximation of $\mathbf{S}$. Starting at $\lambda = 0.5$, λ was annealed linearly to 1 over the first 25,000 steps. This approach failed.

In the 3D Gridworld environment (not shown), similar relationships held after 200,000 steps, although the impact of importance sampling was not as great (due to the two-way local structure) and TD SR did not overtake TD Source. TD Source-SR once again performed best.

Direct Learning vs SR-Reward Decomposition

In Section 3 we identified three update rules: source learning (equation 4), the equivalent update in pseudo-reward space (equation 6), and pseudo-reward descent (equation 12). When $\mathbf{S}_0$ changes during learning, equations 4 and 6 lead to slightly different results, since the former accumulates $\mathbf{v}_0$ and is less sensitive to changes in $\mathbf{S}_0$. The three rules thus define distinct algorithms, which we term *direct methods*. We call the algorithm corresponding to equation 6 *White's algorithm*, as it was first proposed by White 1996 (§6.2.1, with $\lambda = 0$).

There exists a fourth algorithm. Rather than learn $\mathbf{v}$ directly, one may decompose learning into two independent problems: learning $\mathbf{S}_0$ and learning $\mathbf{r}_0$. Then $\mathbf{v} \approx \mathbf{S}_0 \cdot \mathbf{r}_0$. This LSTD-like approach is taken by Kulkarni et al. 2016. We refer to it as *SR-reward decomposition*.

Though White 1996 proposed both White's algorithm and SR-reward decomposition, a comparison of direct learning and decomposition has been lacking. Which is better?

On one hand, White noted that decomposition "will be out of sync until learning is complete" and "as a result, the estimate [of $\mathbf{v}$] may be quite poor." One may formalize this statement by letting $\boldsymbol{\mathcal{E}} = \mathbf{S}_0 - \mathbf{S}$, $\boldsymbol{\epsilon} = \mathbf{r}_0 - \mathbf{r}$, and putting:

$$\begin{aligned} \mathbf{v}_0 - \mathbf{v} &= [\mathbf{S} + \boldsymbol{\mathcal{E}}][\mathbf{r} + \boldsymbol{\epsilon}] - \mathbf{S}\mathbf{r} \\ &= \boldsymbol{\mathcal{E}}\mathbf{r} + \mathbf{S}\boldsymbol{\epsilon} + \boldsymbol{\mathcal{E}}\boldsymbol{\epsilon}. \end{aligned} \tag{18}$$

Then note that the errors compound in the final term.

On the other hand, $\mathbf{r}$ may be easy to learn, so that the latter two terms go to zero and only the error introduced by $\boldsymbol{\mathcal{E}}\mathbf{r}$ is left. But if $\boldsymbol{\mathcal{E}}$ does not go to zero, as in the case of learning a partial source map, then the decomposition approach will not converge to the correct $\mathbf{v}$.

Since direct methods are forgiving of inaccurate $\mathbf{S}_0$, we may hypothesize that their performance will improve relative to decomposition as the source map becomes harder to learn. We tested this in the 3D Gridworld environment, and also took the opportunity to compare the three direct methods (no parallel experiments were done on Random MRPs).

To modulate the difficulty of learning $\mathbf{S}$, we varied the γ parameter. This increases the following bound on $\|\mathbf{S}\|$:

$$\|\mathbf{S}\| = \|\mathbf{I} + \gamma\mathbf{P} + \gamma^2\mathbf{P}^2 \ldots\| \leq 1/(1-\gamma). \tag{19}$$

Intuitively, larger γ means that rewards are propagating back further, which means larger $\mathbf{S}$ and more room for error $\boldsymbol{\mathcal{E}}$.

Figure 4 plots $\|\mathbf{v}_n - \mathbf{v}\|$ over the first 200,000 steps in 3D Gridworld at two γ values for the four algorithms discussed, each at their best tested fixed α, with higher error lines corresponding to higher γ. For each algorithm, the model of $\mathbf{S}$ was learned in the style of TD Source-SR, as described in the previous subsection, so that only the method for computing and updating $\mathbf{v}$ was varied.

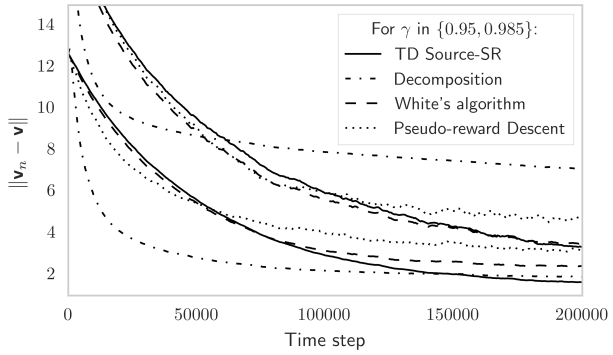

Figure 4: Source learning vs SR-reward decomposition vs pseudo-reward descent (higher lines corresp. to higher γ)

The results show that decomposition results in faster initial learning in all cases and maintains its advantage for longer the smaller γ is. As hypothesized, however, it is overtaken by direct methods as learning progresses.

Triple Model Learning

The results of the previous subsection suggest the following hybrid strategy: use decomposition at the start of learning and then switch to a direct method. While not unreasonable, it is unclear how to design a switching criterion for such an approach. Instead, we propose the following alternative hybrid, which might be viewed as an instance of Dyna: in addition to learning the backward source model and the reward model, learn a forward transition model; then propagate expected temporal differences rather than sampled ones. This entails the following update:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + [\mathbf{S}_0]_{\cdot\langle n\rangle} \left([\mathbf{r}_0]_{\langle n\rangle} + \gamma([\mathbf{P}_0]_{\langle n\rangle\cdot})^T \mathbf{v}_n - v_n(s_{\langle n\rangle}) \right)$$

where $\mathbf{P}_0$ models the transition matrix $\mathbf{P}$.

This update uses three separate models and takes O($|S|$) time per step. Learning $\mathbf{P}_0$ is straightforward in the tabular setting and, like learning $\mathbf{S}_0$ and $\mathbf{r}_0$, takes O($|S|$) time per step. We refer to the resulting algorithm as *triple model learning*. Note that since experiences are used solely for learning the models, and learning $\mathbf{v}_0$ is strictly model-based, updates may be strategically ordered a la prioritized sweeping (Moore and Atkeson 1993). Note further that since $\mathbf{P}_0$ models $p(s_i, s_j)$, it may be used in place of experience when learning $\mathbf{S}_0$, which should reduce variance and eliminate the need for importance sampling.

TD Source-SR vs Triple Model Learning vs iLSTD

In this section we compare TD Source-SR, triple model learning and iLSTD in the 3D Gridworld environment, using the same setup and set of annealing schedules (plus fixed αs) used in figure 2 (right). iLSTD was chosen for comparison because, like TD Source-SR and triple model learning, it is a model-based method that runs in O($|S|$) time per step in the tabular setting. We ran both the random and greedy variants of iLSTD, with $m = 1$ dimensions updated per iteration, as in Geramifard et al. 2007.[1]

The results are shown in figure 5, which is directly comparable to figure 2 (right). All four model-based approaches ended up in approximately the same place. Surprisingly, they performed comparably to source learning with ideal source traces, which tells us that a precisely accurate source map is not so important. Triple model learning had the fastest learning curve by a respectable margin, and both it and TD Source-SR learned faster than the two iLSTD algorithms (but note that only the best *final* results are shown). By reducing the variance of updates, triple model learning outperforms even ideal source traces (final error of 1.45 vs 1.61).

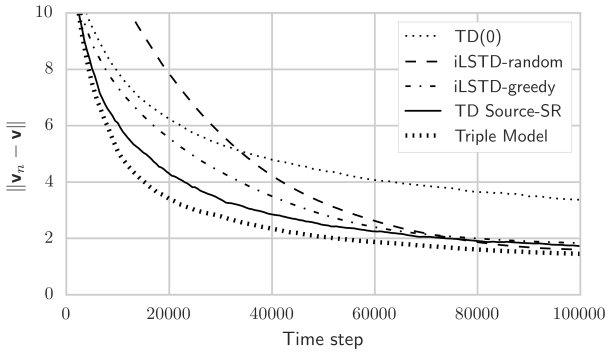

Figure 5: Performance in 3D Gridworld using best tested annealing schedule. Compare to figure 2 (right).

[1]We note that iLSTD, primarily its greedy variant, "blew up" for several annealing schedules. This was the result of accumulating a large $\mathbf{A}$. In these cases, we used our knowledge of the true $\mathbf{v}$ to terminate learning early.

5 Useful Properties of Source Traces

Setting aside any potential gains in learning speed, there are at least three properties that make the source model useful.

Time Scale Invariance

Given some continuous process that can be modeled with a discrete-time MRP, there is a choice of how granular time steps should be; we may choose to model a time step as one second or as one tenth of a second. Source traces are invariant to the granularity in the following sense: if the credit that state s_i receives for state s_j is $[\mathbf{S}]_{ij}$, this value will not change if the steps between s_i and s_j are defined with larger or smaller intervals (assuming γ is adjusted appropriately). As noted by Dayan 1993, the source map "effectively factors out the entire temporal component of the task."

Time scale invariance means that source traces capture something about the underlying process that is independent of how we structure the state and transition model for that process. In particular, an agent equipped with a source map $\mathbf{S}$ can use it to answer the following questions about cause and effect: "What is likely to cause X?" and "What are the likely effects of Y?" The answers could be used, in turn, for justification; i.e., as a step toward answering questions like: "Why did you do X?" or "Why did Y happen?"

We leave further development on this topic to future work.

Reward Invariance

Source traces are independent of the reward model. This makes them useful for transfer learning, for learning in non-stationary environments, and for evaluating hypotheticals.

The latter point—evaluating hypotheticals—is central to general intelligence. As this application is only relevant in control settings, however, we defer it to future work.

On the former points—transfer learning and learning in non-stationary environments—source traces provide a method of quickly adapting to changing reward values. This was investigated from the lens of SR in the work of Dayan 1993, as well as in the follow-up works of Barreto et al. 2016, Kulkarni et al. 2016, Zhang et al. 2016, and Lehnert, Tellex, and Littman 2017. The only novelty offered by the backward-view in this context is the ability to propagate the impact of a reward change at a single state to the entire value space in $O(|S|)$ time. This ability may be useful in cases involving communication; for example, if a teacher informs the agent that they are misjudging the value of state s_j, that information can be used to directly update $\mathbf{v}_0$.

Trajectory Independence

Like eligibility traces, source traces distribute TD errors more than one step, thereby increasing sample efficiency. But unlike eligibility traces, source traces are *independent of the current trajectory*. This means they can be used in cases where isolated experiences are preferred to trajectories.

In particular, there are at least two reasons why we would prefer to sample experiences instead of trajectories from an experience replay or Dyna model. First, random sampling can alleviate problems caused by correlated data (Mnih et al. 2015). Second, methods for prioritizing samples improve

	Target $\|\mathbf{v}_0 - \mathbf{v}\|$		
	3.0	2.0	1.5
TD(0)	154	465	979
TD(0) w/ ER	58	161	319
TD Source-SR	58	158	335
TD Source-SR w/ ER	41	100	189

Table 1: Thousands of steps to reach target error values in 3D Gridworld when using experience replay and source traces

the efficiency by sampling experiences out of order (Schaul et al. 2015; Moore and Atkeson 1993). In these cases, source traces may be used where eligibility traces are inapplicable.

To demonstrate this benefit, we ran each of TD(0), TD(0) with ER, TD Source-SR, and TD Source-SR with ER at various *fixed* learning rates in the 3D Gridworld environment. The replay memory had infinite capacity, and was invoked on every step to replay 3 past steps. For three "target" levels of error, we computed the number of real steps that it took for the average error (over the set of 30 3D Gridworlds) to fall below the target (in each case, at the best tested α). The results in Table 1 show that switching from one-step backups to source backups during ER can speed up learning.

6 Conclusion

This paper has developed the theory of source traces in the context of valuing tabular finite-state MRPs. Our contributions include the proposed source learning algorithm, a theorem for its convergence, the concept of partial source traces (and SR), the TD-Source and TD Source-SR algorithms, a comparison of direct methods and decomposition, the triple model learning algorithm, and a demonstration of the effectiveness of combining ER and source traces. While this work serves as a necessary foundational step, in most practical scenarios, we will be concerned with continuous, uncountable-state Markov decision processes (MDPs). In anticipation of future work, we conclude with a brief comment on the challenges posed by these settings.

For purposes of control, we note that $\mathbf{S}$ changes as policy changes. To capture the same benefits in a control setting, one must either construct a higher order source function, learn a set of source maps (one for each policy), or rely on the error bound provided by the convergence theorem and the (not necessarily true) assumption that minor changes in policy entail only minor changes in the source map.

For purposes of approximation, note that extending source traces to propagate TD errors back to features entails the problem of agglomeration: since features, unlike states, are not isolated from each other (multiple features appear in the same state), collecting their historical accumulations into a single trace will erode the usefulness of its interpretability. It may therefore be more appropriate to construct a generative source model that captures a time-invariant distribution of past states. Such a model would maintain both its interpretability and usefulness for model-based learning.

References

Barreto, A.; Munos, R.; Schaul, T.; and Silver, D. 2016. Successor features for transfer in reinforcement learning. *arXiv preprint arXiv:1606.05312*.

Bradtke, S. J., and Barto, A. G. 1996. Linear least-squares algorithms for temporal difference learning. In *Recent Advances in Reinforcement Learning*. Springer. 33–57.

Dayan, P. 1993. Improving generalization for temporal difference learning: The successor representation. *Neural Computation* 5(4):613–624.

Gehring, C. 2015. Approximate linear successor representation (extended abstract). *The 2nd Multidisciplinary Conference on Reinforcement Learning and Decision Making*.

Geramifard, A.; Bowling, M.; Zinkevich, M.; and Sutton, R. S. 2007. iLSTD: Eligibility traces and convergence analysis. In *Advances in Neural Information Processing Systems*, 441–448.

Geramifard, A.; Bowling, M.; and Sutton, R. S. 2006. Incremental least-squares temporal difference learning. In *Proceedings of the 21st national conference on Artificial intelligence-Volume 1*, 356–361. AAAI Press.

Jaakkola, T.; Jordan, M. I.; and Singh, S. P. 1994. Convergence of stochastic iterative dynamic programming algorithms. In *Advances in neural information processing systems*, 703–710.

Kemeny, J. G., and Snell, J. L. 1976. *Finite markov chains*. Springer-Verlag, second edition.

Kulkarni, T. D.; Saeedi, A.; Gautam, S.; and Gershman, S. J. 2016. Deep successor reinforcement learning. *arXiv preprint arXiv:1606.02396*.

Lehnert, L.; Tellex, S.; and Littman, M. L. 2017. Advantages and limitations of using successor features for transfer in reinforcement learning. *arXiv preprint arXiv:1708.00102*.

Lin, L.-H. 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning* 8(3/4):69–97.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Moore, A. W., and Atkeson, C. G. 1993. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine learning* 13(1):103–130.

Schaul, T.; Quan, J.; Antonoglou, I.; and Silver, D. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An Introduction*. The MIT Press, Cambridge.

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

Sutton, R. S. 1990. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the seventh international conference on machine learning*, 216–224.

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

White, L. M. 1996. *Temporal difference learning: eligibility traces and the successor representation for actions.* Univ. of Toronto.

Zhang, J.; Springenberg, J. T.; Boedecker, J.; and Burgard, W. 2016. Deep reinforcement learning with successor features for navigation across similar environments. *arXiv preprint arXiv:1612.05533*.