SUCCESSOR-ERROR AMPLIFICATION IN TEMPORAL-DIFFERENCE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

With function approximation, a temporal-difference (TD) update at one state can change the prediction at its successor, an effect we call *successor shift*. The successor is special because its prediction also appears in the bootstrap target used to form the update. We show that this creates a feedback mechanism in which the successor's value error can amplify itself through the TD update, which we call *successor-error amplification*. Under simple conditions on deterministic chains, this amplification is strong enough that the TD update increases the successor's squared value error in expectation. To mitigate this amplification, we derive Successor-Shift-Controlled TD (SSC-TD), which suppresses successor shift while preserving the norm of the TD update. An adaptive version estimates the shift's average effect on the successor's squared value error, and activates suppression when the estimate indicates harm. Experiments show adaptive SSC-TD lowers lifetime MSE relative to TD on a majority of Atari environments.

1 INTRODUCTION

Function approximation uses shared parameters to predict values at different states. An update at one state can therefore change predictions at others. We refer to these changes as *generalization*. When the affected predictions need similar corrections, generalization lets one update improve several predictions, which is an advantage over updating each state separately in a table. Consecutive states often share features, so an update affects the prediction at the next state (the successor) . We call the change at the successor value *successor shift*. In temporal-difference (TD) learning (Sutton 1988), the successor is special: it's predicted value also helps determine the target for the current update.

Our central observation is that using the successor's prediction to form the TD target can cause the update to amplify the error in that same successor's prediction. Suppose the successor's value is overestimated, as in Figure 1. Using the overestimate rather than the true successor value raises the TD target, contributing an upward change to the prediction of current state's value. With positive feature overlap, generalization carries this upward contribution back to the successor's prediction, increasing the original overestimate. An underestimate is amplified in the same way, with the signs reversed. We call this mechanism *successor-error amplification*.

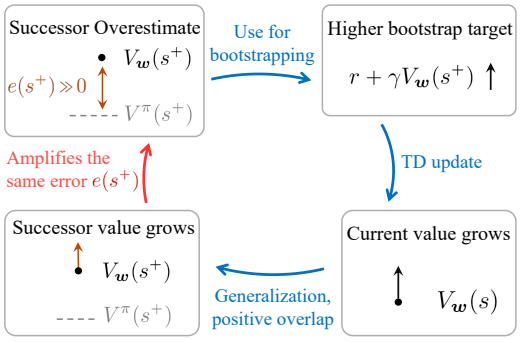

Figure 1: Successor-error amplification. s and s^+ denote the current and successor states, respectively. Dots denote value estimates $V_{\boldsymbol{w}}$, dashed lines true values V^π, and $e(s^+) = V_{\boldsymbol{w}}(s^+) - V_\pi(s^+)$ the successor's value error. An overestimate at the successor contributes to an update that increases the same overestimate.

Successor-error amplification results from combination of generalization and bootstrapping. A Monte Carlo update can also generalize to the successor, but its target does not contain the successor prediction and therefore provides no such self-amplifying feedback for the successor's value error.

Successor-error amplification can make the complete TD update increase the successor's squared value error on average. We make this observation precise on stationary deterministic chains with

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

constant positive feature overlap between consecutive states. On such chains, we derive the expected change in the successor's squared value error caused by a TD update. In the average-reward setting, this change is always nonnegative, so generalization to the successor never helps on average. Under the same chain assumptions, we also show that generalization to the successor is never more beneficial than generalization to the predecessor (the previous state), measured by the same expected error change. For general environments however, successor shift is not universally harmful: useful transfer between states needing similar corrections can outweigh amplification. We therefore seek to control successor shift rather than remove it unconditionally.

We present Successor-Shift Controlled TD (SSC-TD) to control successor shift while preserving the norm of the TD update. SSC-TD removes a chosen fraction of the update component aligned with the successor feature, then rescales the remainder to TD update's norm. SSC-TD and TD therefore move the parameters equally far and differ only in direction.

Whether successor shift helps or harms depends on the transition dynamics, discount factor, representation, and current value estimates. Its effect can therefore change during learning even when the environment and representation remain fixed. We adapt the suppression according to the average effect of TD's successor shift on the successor's value error: an online estimate of this effect turns suppression on when the effect is positive and off otherwise.

Empirically, we find that successor-error amplification can be substantial, while the net effect of successor shift can change from helpful to harmful during learning. Adaptive SSC-TD responds to this changing utility and reduces lifetime MSE relative to TD on a majority of Atari games. SSC-TD also remains stable on Baird's counterexample, where TD diverges.

2 BACKGROUND

We study value prediction for a fixed policy. The policy and environment induce a Markov reward process with transition kernel $P(s^+ \mid s)$ and reward distribution $R(\cdot \mid s, s^+)$, from which we observe a stream of states and rewards $s_0, r_0, s_1, r_1, \ldots$ The value of a state is the expected discounted return, $V^\pi(s) = \mathbb{E}_\pi\big[\sum_{k\geqslant 0} \gamma^k r_k \mid s_0 = s\big]$, with discount factor $\gamma \in [0, 1)$. The goal is to estimate V^π from the observed stream.

We write d^π for an on-policy stationary state distribution, and s^-, s, s^+ for the predecessor, current state, and successor. States are denoted by lower case letters, including sampled states. Under d^π the chain is stationary, and we write $\mathbb{E}_\pi$ for expectation over a stationary segment $(s^-, s, r, s^+, r^+, s^{++})$, with $s \sim d^\pi$, successors drawn from P, and the predecessor from the same stationary chain.

We use linear function approximation so that prediction changes can be computed exactly. Each state s has a feature vector $\boldsymbol{x}_s \stackrel{\text{def}}{=} \boldsymbol{x}(s) \in \mathbb{R}^n$, and $V_{\boldsymbol{w}}(s) = \boldsymbol{x}_s^\top \boldsymbol{w}$. When describing a single update, all predictions and errors are evaluated at the same pre-update weights $\boldsymbol{w}$. All analyses and algorithms are extendable to differentiable nonlinear value function by replacing features by value gradients, which gives the corresponding prediction changes to first order in the step size.

TD(0) moves the current estimate toward a bootstrap target formed from the estimate at the next state. For an observed transition (s, r, s^+),

$$\delta_s \stackrel{\text{def}}{=} r + \gamma V_{\boldsymbol{w}}(s^+) - V_{\boldsymbol{w}}(s), \qquad \Delta \boldsymbol{w}_s^{\mathrm{TD}} = \alpha \delta_s \boldsymbol{x}_s, \tag{1}$$

where $\alpha > 0$ is the step size and the new weights are $\boldsymbol{w} + \Delta \boldsymbol{w}_s^{\mathrm{TD}}$. When there is no ambiguity in notation, we write $\Delta \boldsymbol{w}_s$ instead of $\Delta \boldsymbol{w}_s^{\mathrm{TD}}$. The subscript on δ_s identifies the starting state of the observed transition. Its reward and successor are supplied by that transition.

Successor shift. Through generalization, the update also changes predictions at other states. For any parameter increment $\Delta \boldsymbol{w}$ and any state u, define

$$\Delta_{\Delta \boldsymbol{w}} V(u) \stackrel{\text{def}}{=} V_{\boldsymbol{w}+\Delta \boldsymbol{w}}(u) - V_{\boldsymbol{w}}(u) = \boldsymbol{x}_u^\top \Delta \boldsymbol{w}. \tag{2}$$

Then, for TD update at state s, $\Delta_{\Delta \boldsymbol{w}_s} V(u) = \alpha \delta_s \boldsymbol{x}_s^\top \boldsymbol{x}_u$. The feature inner product therefore determines the signed generalization from s to u. Writing $c_s \stackrel{\text{def}}{=} \boldsymbol{x}_s^\top \boldsymbol{x}_{s^+}$ for the successor feature

overlap, the changes at the current state and its successor are

$$\Delta_{\Delta \boldsymbol{w}_s} V(s) = \alpha \delta_s \|\boldsymbol{x}_s\|^2, \qquad \Delta_{\Delta \boldsymbol{w}_s} V(s^+) = \alpha \delta_s c_s. \tag{3}$$

Successor shift is the change $\Delta_{\Delta \boldsymbol{w}_s} V(s^+)$ caused by the update at s, shown in the last equation. The corresponding predecessor shift is $\Delta_{\Delta \boldsymbol{w}_s} V(s^-) = \alpha \delta_s c_{s^-}$, where $c_{s^-} = \boldsymbol{x}_{s^-}^\top \boldsymbol{x}_s$. In a tabular representation, different states have orthogonal features, so these shifts are absent between distinct states. With shared features, both are typically non-zero.

Measuring the effect of generalization. The benefits of generalization need not be the same at the successor and predecessor. We measure them separately by asking how much an update at s changes the squared value error at each of s^+ and s^-. For each state u, let $e(u) \stackrel{\text{def}}{=} V_{\boldsymbol{w}}(u) - V^\pi(u)$ be the value error before the update. Since the true value does not change,

$$\Delta_{\Delta \boldsymbol{w}_s} e(u) \stackrel{\text{def}}{=} \big(V_{\boldsymbol{w}+\Delta \boldsymbol{w}_s}(u) - V^\pi(u)\big) - e(u) = \Delta_{\Delta \boldsymbol{w}_s} V(u). \tag{4}$$

The update subscript identifies the parameter change, and the argument identifies where its effect is measured. We define the successor and predecessor generalization measures by

$$\begin{aligned} \mathcal{G}_{\text{succ}}(\boldsymbol{w}) &\stackrel{\text{def}}{=} \frac{1}{2}\mathbb{E}_\pi\Big[\big(e(s^+) + \Delta_{\Delta \boldsymbol{w}_s} e(s^+)\big)^2 - e(s^+)^2\Big], \\ \mathcal{G}_{\text{pre}}(\boldsymbol{w}) &\stackrel{\text{def}}{=} \frac{1}{2}\mathbb{E}_\pi\Big[\big(e(s^-) + \Delta_{\Delta \boldsymbol{w}_s} e(s^-)\big)^2 - e(s^-)^2\Big]. \end{aligned} \tag{5}$$

Every update inside these expectations starts from the same $\boldsymbol{w}$. A positive measure means that generalization increases the affected state's squared value error on average; a negative measure means that it reduces the error. $\mathcal{G}_{\text{succ}}$ and $\mathcal{G}_{\text{pre}}$ are changes in error, rather than losses to be minimized. In contrast to the usual overall value-error loss, $\mathbb{E}_\pi[e(s)^2]$, $\mathcal{G}_{\text{succ}}$ evaluates the successor value of each sampled state after a different state-specific update $\Delta \boldsymbol{w}_s$. A positive $\mathcal{G}_{\text{succ}}$ therefore does not imply that the overall value error increases or that learning diverges.

Strong successor overlap is common in deep reinforcement learning. Consecutive observations differ little, and function approximators map them to nearby points in feature space. Bengio et al. (2020) report strong alignment between TD(λ) update gradients at nearby states in MsPacman, extending over tens of steps. Kumar et al. (2022) observe growing dot products between current and successor state-action features during offline TD and Q-learning on Atari. Such overlap allows an update to change the successor prediction substantially. Whether that change improves the prediction at successor is the question addressed by $\mathcal{G}_{\text{succ}}$.

3 Successor-Error Amplification

We show how bootstrapping and generalization together can amplify the successor's value error. The error enters the bootstrap target and contributes to the TD update at the current state. Generalization then carries this contribution back to the successor. With positive feature overlap, the contribution increases an overestimate or deepens an underestimate, pushing the successor's prediction farther from its true value.

Hold $\boldsymbol{w}$ fixed and consider the sampled TD update $\Delta \boldsymbol{w}_s = \alpha \delta_s \boldsymbol{x}_s$ for a transition (s, r, s^+). Substituting $V_{\boldsymbol{w}} = V^\pi + e$ into (1) gives

$$\delta_s = r + \gamma V_{\boldsymbol{w}}(s^+) - V_{\boldsymbol{w}}(s) = \underbrace{r + \gamma V^\pi(s^+) - V^\pi(s)}_{\text{zero in expectation given } s} + \gamma e(s^+) - e(s). \tag{6}$$

An overestimate at the current state ($e(s) > 0$) contributes a downward correction to the current state's prediction. In contrast, an overestimate at the successor ($e(s^+) > 0$) raises the bootstrap target and contributes an upward change to that same prediction.

Through generalization, the update at the current state also changes the successor prediction. Substituting (6) into the successor shift (3), we obtain

$$\begin{aligned} \Delta_{\Delta \boldsymbol{w}_s} e(s^+) &= \Delta_{\Delta \boldsymbol{w}_s} V(s^+) = \alpha \delta_s c_s \\ &= \alpha c_s \big[r + \gamma V^\pi(s^+) - V^\pi(s)\big] - \alpha c_s e(s) + \underbrace{\alpha \gamma c_s e(s^+)}_{\text{successor-error amplification}}. \end{aligned} \tag{7}$$

The last term traces the error's return to the successor. The error enters the bootstrap target as $\gamma e(s^+)$ and returns through generalization as $\alpha\gamma c_s e(s^+)$. For $\gamma > 0$ and $c_s > 0$, this returned contribution increases an overestimate and deepens an underestimate. Figure 1 traces the same path. We call this mechanism *successor-error amplification*. Its contribution to the first-order change in $\frac{1}{2}e(s^+)^2$ is $\alpha\gamma c_s e(s^+)^2$, which is positive whenever the successor error is nonzero. At the current state, the contribution from its own error is corrective instead, giving $-\alpha\|\boldsymbol{x}_s\|^2 e(s)^2$ in the first-order change in $\frac{1}{2}e(s)^2$.

The amplification term $\alpha\gamma c_s e(s^+)$ in (7) comes from using the successor prediction in the bootstrap target. A Monte Carlo return does not depend on that prediction, so the corresponding update has no such term. Feature overlap still allows generalization to the successor, but without this error-amplifying contribution.

Successor-error amplification can outweigh the remaining terms in (7) and make the complete successor shift harmful on average. We now quantify this effect and compare it with generalization to the predecessor. As an example, consider a stationary chain with deterministic transitions and rewards, and constant positive consecutive overlap $c_s = c > 0$. The bracketed term in (7) is then zero on every transition, so $\delta_s = \bar{\delta}_s = \gamma e(s^+) - e(s)$. All expectations hold the weights fixed, and we assume finite second moments. Expanding the square in (5) gives

$$\mathcal{G}_{\text{succ}} = \mathbb{E}_\pi\Big[e(s^+)\Delta_{\Delta \boldsymbol{w}_s} e(s^+) + \frac{1}{2}\big(\Delta_{\Delta \boldsymbol{w}_s} e(s^+)\big)^2\Big] = \alpha c\, \mathbb{E}_\pi\Big[\underbrace{\gamma e(s^+)^2}_{\text{amplification}} - e(s)e(s^+) + \frac{\alpha c}{2}\delta_s^2\Big]. \tag{8}$$

Stationarity gives $\mathbb{E}_\pi[e(s)^2] = \mathbb{E}_\pi[e(s^+)^2]$. Completing the square inside the expectation yields

$$\mathcal{G}_{\text{succ}} = \alpha c\, \mathbb{E}_\pi\Big[\underbrace{\frac{1}{2}\big(e(s^+) - e(s)\big)^2 + \frac{\alpha c}{2}\delta_s^2}_{\geqslant 0} - (1-\gamma)e(s)^2\Big]. \tag{9}$$

The successor generalization measure, $\mathcal{G}_{\text{succ}}$, is positive when the error-difference and finite-step terms outweigh the last term, in which case the complete shift is harmful. Consecutive errors that are close can instead make the shift helpful. The balance therefore depends on the current value errors, not only on the chain and representation.

Generalization to the predecessor is more helpful under the same assumptions, although it need not reduce the predecessor's error. We prove in Appendix A that

$$\mathcal{G}_{\text{succ}} - \mathcal{G}_{\text{pre}} = \frac{\alpha\gamma c}{2}\mathbb{E}_\pi\Big[\big(e(s^+) - e(s^-)\big)^2\Big] \geqslant 0. \tag{10}$$

The ordering holds for every $\alpha > 0$ and $\gamma \in [0, 1)$ under these assumptions. For $\gamma > 0$, the inequality is strict unless $e(s^+) = e(s^-)$ almost surely. The fact that generalization is more favorable to the predecessor than to the successor is consistent with the direction in which the update uses information. The shift at s^+ comes from an update whose target depends on $V_{\boldsymbol{w}}(s^+)$, but not on any information from its future. The shift at s^- instead carries information from farther its future, s^+.

The average-reward case gives a particularly clear consequence. For differential TD (e.g., see Section 10.3 of Sutton & Barto 2018) on this deterministic chain, with the true average reward subtracted and a fixed differential-value reference, the same identities hold with $\gamma = 1$. The discount term in (9) vanishes, so $\mathcal{G}_{\text{succ}} \geqslant 0$ for every $\alpha > 0$, with equality exactly when consecutive errors agree almost surely. Therefore, successor generalization is always harmful (or neutral at best) in such average-reward setting. The predecessor–successor ordering also remains valid.

Two questions follow. How can successor shift be suppressed? And how can suppression adapt during learning to retain useful generalization and suppress harmful shift? The next two sections address these questions.

4 Suppressing Successor Shift

We now derive an update that reduces successor shift while preserving the norm of the TD update. The TD update is itself the solution of a proximal problem,

$$\Delta \boldsymbol{w}_s^{\text{TD}} = \arg\min_{\Delta \boldsymbol{w}} \; -\delta_s \Delta_{\Delta \boldsymbol{w}} V(s) + \frac{1}{2\alpha}\|\Delta \boldsymbol{w}\|^2.$$

The first term is the first-order reduction of $\frac{1}{2}\delta_s^2$ with the bootstrap target held fixed, and the second favors a small parameter change. We add a penalty on successor shift. For nonzero current and successor features and a suppression strength $\eta \in [0, 1)$, consider

$$\min_{\Delta \boldsymbol{w}} \; -\delta_s \Delta_{\Delta \boldsymbol{w}} V(s) + \frac{1}{2\alpha}\left(\|\Delta \boldsymbol{w}\|^2 + \frac{\eta}{1-\eta}\left(\frac{\Delta_{\Delta \boldsymbol{w}} V(s^+)}{\|\boldsymbol{x}_{s^+}\|} \right)^2 \right). \tag{11}$$

The added term penalizes movement along the successor's unit feature direction. Normalizing by $\|\boldsymbol{x}_{s^+}\|$ makes this cost independent of successor feature scale. The factor $\eta/(1-\eta)$ maps $\eta = 0$ to no penalty and makes the penalty arbitrarily strong as η approaches one.

The objective is strictly convex, and its unique minimizer is

$$\alpha\delta_s \left(\boldsymbol{x}_s - \eta \frac{c_s}{\|\boldsymbol{x}_{s^+}\|^2} \boldsymbol{x}_{s^+} \right), \tag{12}$$

which is the TD update with the fraction η of its projection onto $\boldsymbol{x}_{s^+}$ removed. This turns the TD update away from $\boldsymbol{x}_{s^+}$, but also shrinks its size by the factor $n_s \overset{\text{def}}{=} \sqrt{1 - \eta(2-\eta)\rho_s^2} \in (0, 1]$, where $\rho_s \overset{\text{def}}{=} c_s/(\|\boldsymbol{x}_s\|\|\boldsymbol{x}_{s^+}\|)$ is the cosine similarity of the two features. We rescale (12) to the length of the TD update, so that SSC-TD and TD move the parameters equally far and differ only in direction:

$$\Delta \boldsymbol{w}_s^{\text{SSC}-\text{TD}} = \frac{\alpha\delta_s}{n_s} \left(\boldsymbol{x}_s - \eta \frac{c_s}{\|\boldsymbol{x}_{s^+}\|^2} \boldsymbol{x}_{s^+} \right). \tag{13}$$

Figure 2 shows both steps, and Appendix B gives the derivation.

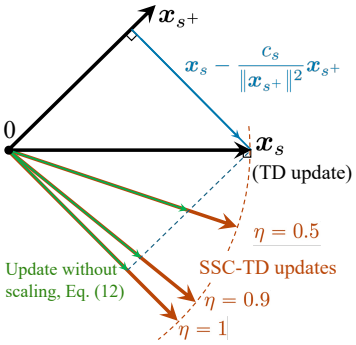

Figure 2: SSC-TD geometry for $\alpha\delta_s = 1$. Green arrows show solutions of (11); brown arrows show SSC-TD update (13).

Taking inner products with the two feature vectors gives the resulting prediction changes,

$$\Delta_{\Delta \boldsymbol{w}_s^{\text{SSC}-\text{TD}}} V(s) = \frac{1-\eta\rho_s^2}{n_s} \Delta_{\Delta \boldsymbol{w}_s^{\text{TD}}} V(s), \qquad \Delta_{\Delta \boldsymbol{w}_s^{\text{SSC}-\text{TD}}} V(s^+) = \frac{1-\eta}{n_s} \Delta_{\Delta \boldsymbol{w}_s^{\text{TD}}} V(s^+). \tag{14}$$

Both fractions in (14) lie in $(0, 1]$, so neither prediction change is enlarged or reversed. For fixed $|\rho_s| < 1$, the successor fraction decreases continuously to zero as η approaches one, so suppression depends on both η and the feature alignment. For small η, the successor fraction decreases at first order in η, and the current-state fraction decreases more slowly at second order.

When $\eta = 0$, SSC-TD is TD. It also reduces to TD when $c_s = 0$, since there is no successor shift to suppress, and when $|\rho_s| = 1$, since the two predictions then depend on the same parameter direction and cannot be controlled separately. For a terminal successor or a zero feature vector, we use TD.

5 Adapting the Suppression

Successor shift is not always harmful. Depending on the problem and on the current weights, the shift can reduce the successor's value error or can increase it. For example, the term $-\alpha c_s e(s)$ in (7) can outweigh amplification term when $c_s > 0$ and the two value errors have the same sign. The suppression of Section 4 can therefore remove useful generalization, and its benefit can change during learning. We therefore adapt the suppression, η, during learning, based on whether TD's complete successor shift increases or decreases the successor's squared value error on average.

The successor measure $\mathcal{G}_{\text{succ}}$ in (5) gives the criterion for the impact of an update on the successor error: positive values indicate harm, and negative values imply helpful generalization to the successor. Expanding the square and substituting TD's shift from (3) gives

$$\mathcal{G}_{\text{succ}} = \alpha\, \mathbb{E}_\pi\big[c_s\delta_s e(s^+)\big] + \frac{\alpha^2}{2}\, \mathbb{E}_\pi\big[(c_s\delta_s)^2\big]. \tag{15}$$

The first term in (15) determines the sign of $\mathcal{G}_{\text{succ}}$ for small step sizes. We therefore suppress successor shift when $\mathbb{E}_\pi[c_s\delta_s\, e(s^+)]$ is positive.

To estimate $\mathbb{E}_\pi[c_s\delta_s\, e(s^+)]$ without true values, we use observed returns. Let $G_{t+1} = \sum_{k \geqslant t+1} \gamma^{t-k-1} r_k$ denote the observed return from s_{t+1}, with the sum ending at termination in an

episodic problem. Let δ_t and c_t refer to the transition at time t. Because $\mathbb{E}_\pi[G_{t+1}|s_{t+1}] = V^\pi(s_{t+1})$, the sample $c_t\delta_t\big(V_{\boldsymbol{w}}(s_{t+1}) - G_{t+1}\big)$ has mean $\mathbb{E}_\pi[c_s\delta_s e(s^+)]$ at fixed weights under stationary sampling. For online estimation, the following lemma shows that the sample sum can be accumulated causally, using an idea analogous to the backward view of TD(λ) (Sutton & Barto 2018).

Lemma 1. *At fixed weights, let $h_0 = 0$ and $h_{k+1} = \gamma h_k + c_k\delta_k$. Then, over any episode,*

$$\sum_{t\geqslant 0} c_t\delta_t\big(V_{\boldsymbol{w}}(s_{t+1}) - G_{t+1}\big) = -\sum_{k\geqslant 1}\delta_k h_k.$$

Each term $-\delta_k h_k$ is available at time k, and their sum equals the total of the return-based samples. We use an exponential moving average of $-\delta_k h_k$, with the changing-weight correction of Appendix C, to track the criterion $\mathbb{E}_\pi[c_s\delta_s e(s^+)]$ during learning. Denoting this moving-average estimate by $\hat{\mathcal{G}}$, we set

$$\eta = \begin{cases} \eta_{\max}, & \hat{\mathcal{G}} > 0, \\ 0, & \hat{\mathcal{G}} \leqslant 0, \end{cases} \qquad 0 \leqslant \eta_{\max} < 1. \tag{16}$$

Because the value estimates change during learning, the suppression chosen by (16) changes as well. Appendix C extends Lemma 1 to changing weights, gives proof, and provides a complete algorithm.

6 Experiments

We measure how large the successor-error amplification is, and test whether SSC-TD removes the harm while keeping the useful transfer. We then evaluate SSC-TD on Atari prediction and on problems where TD is unstable.

6.1 How Large Is Successor-Error Amplification?

Section 3 argued that successor shift carries an error-amplifying component, and that the complete shift can nonetheless help. It helps in some problems and hurts in others, and a single problem can change from one to the other during training. We measure how large the harmful component is. In particular, we measured successor-error amplification against the rest of TD's successor shift in (7), and compared that shift with two shifts that contain no amplification: TD's predecessor shift and Monte Carlo's successor shift.

Successor-measure depends on the value error, so we used deterministic problems with fixed policies to compute the true values exactly. We used Mountain Car (e.g., see Sutton & Barto 2018) and CartPole (Barto et al. 1983) with deterministic policies mentioned in Appendix D.1. We trained TD at a small step size, froze the weights at checkpoints, and evaluated all measures at the frozen TD weights, so that TD and Monte Carlo were compared on the same value errors. For more details please see Appendix D.1.

TD and Monte Carlo updates have the form $\Delta\boldsymbol{w}_s = \alpha\varepsilon_s\boldsymbol{x}_s$. On a deterministic problem, $\varepsilon_s^{\mathrm{TD}} = \delta_s = \gamma e(s^+) - e(s)$ and $\varepsilon_s^{\mathrm{MC}} = G_s - V_{\boldsymbol{w}}(s) = -e(s)$, where G_s is the return from s. The measure $\mathcal{G}_{\mathrm{succ}}$ in (5) is the change in the squared value error at s^+ caused by the update at the current state s. We normalize it by the change caused by the update at s^+ itself,

$$\mathcal{N}_{\mathrm{succ}} \stackrel{\text{def}}{=} \tfrac{1}{2}\mathbb{E}_\pi\Big[\big(e(s^+) + \Delta_{\Delta\boldsymbol{w}_{s^+}} e(s^+)\big)^2 - e(s^+)^2\Big],$$

and report

$$C_{\mathrm{succ}} \stackrel{\text{def}}{=} -\lim_{\alpha\to 0}\frac{\mathcal{G}_{\mathrm{succ}}}{\mathcal{N}_{\mathrm{succ}}} = -\lim_{\alpha\to 0}\frac{\alpha\mathbb{E}_\pi[c_s e(s^+)\varepsilon_s] + O(\alpha^2)}{\alpha\mathbb{E}_\pi[\|\boldsymbol{x}_{s^+}\|^2 e(s^+)\varepsilon_{s^+}] + O(\alpha^2)} = \frac{\mathbb{E}_\pi[c_s e(s^+)\varepsilon_s]}{D_{\mathrm{succ}}}, \tag{17}$$

where $D_{\mathrm{succ}} \stackrel{\text{def}}{=} -\mathbb{E}_\pi[\|\boldsymbol{x}_{s^+}\|^2 e(s^+)\varepsilon_{s^+}]$. Both $\mathcal{G}_{\mathrm{succ}}$ and $\mathcal{N}_{\mathrm{succ}}$ are computed with the same update, TD or Monte Carlo, which we mark by a superscript. Similarly, we normalize $\mathcal{G}_{\mathrm{pre}}$ by the change caused by the update at s^- itself, which gives $C_{\mathrm{pre}} = \mathbb{E}_\pi[c_{s^-}e(s^-)\varepsilon_s]/D_{\mathrm{pre}}$ with $D_{\mathrm{pre}} \stackrel{\text{def}}{=} -\mathbb{E}_\pi[\|\boldsymbol{x}_{s^-}\|^2 e(s^-)\varepsilon_{s^-}]$. A measure of -1 means that the update at s reduces the neighbour's squared error as much as the neighbour's own update does.

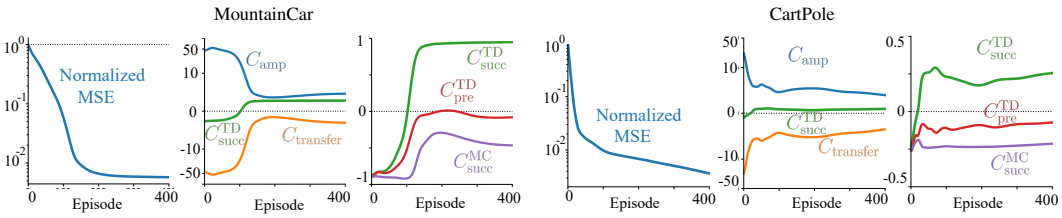

Figure 3: Successor-error amplification in Mountain Car and CartPole. In each triple, the first panel shows the root mean squared value error of the TD run, divided by its initial value. The second splits TD's successor measure $C^{\text{TD}}_{\text{succ}}$ into amplification C_{amp} and transfer C_{transfer} (rendered on a non-linear y-axis, $\text{sign}(y)\log_2(1+|y|)$). The third compares $C^{\text{TD}}_{\text{succ}}$ with TD's predecessor measure $C^{\text{TD}}_{\text{pre}}$ and the Monte Carlo successor measure $C^{\text{MC}}_{\text{succ}}$. Positive measures indicate harm.

For TD, (7) splits the first-order term of $\mathcal{G}_{\text{succ}}$ into amplification, $\alpha\mathbb{E}_\pi[\gamma c_s e(s^+)^2]$, and transfer, $-\alpha\mathbb{E}_\pi[c_s e(s)e(s^+)]$. Normalizing each by $D^{\text{TD}}_{\text{succ}}$ gives

$$C^{\text{TD}}_{\text{succ}} = C_{\text{amp}} + C_{\text{transfer}}, \qquad C_{\text{amp}} \stackrel{\text{def}}{=} \frac{\mathbb{E}_\pi[\gamma c_s e(s^+)^2]}{D^{\text{TD}}_{\text{succ}}}, \quad C_{\text{transfer}} \stackrel{\text{def}}{=} \frac{-\mathbb{E}_\pi[c_s e(s)e(s^+)]}{D^{\text{TD}}_{\text{succ}}}. \tag{18}$$

Figure 3 shows the measures through learning, together with the value error of the TD run. C_{amp} was positive throughout on both problems, reaching 53.65 in Mountain Car and 21.96 in CartPole. C_{transfer} was negative and largest early in learning, when the initial weights made the value errors of consecutive states aligned, and TD's successor shift then helped. From episode 103 in Mountain Car and 20 in CartPole, amplification dominated and $C^{\text{TD}}_{\text{succ}}$ stayed positive. In contrast, $C^{\text{TD}}_{\text{pre}}$ and $C^{\text{MC}}_{\text{succ}}$ were negative at almost every checkpoint. TD update thus helped its predecessor while harming its successor, and a Monte Carlo update from the same value errors helped the same successor. Mean consecutive feature cosine similarity ρ_s was 0.848 in Mountain Car and 0.254 in CartPole.

6.2 When Does Successor Shift Help?

Whether the complete shift is useful depends on whether neighbouring states need similar corrections. We isolated that dependence by fixing the problem and the representation and changing only the initial value errors.

We used a deterministic cycle of $n = 20$ states with zero rewards, so $V^\pi = 0$, and each prediction is equal to its value error. We used fixed n-dimensional feature vectors, where state s_i is represented by three consecutive ones at coordinates $i, i+1, i+2$ modulo n. Consecutive states share two active features, so $\rho_{s_i} = 2/3$ at every transition.

We used three initial-error patterns,

$$V_{\boldsymbol{w}_0}(s_i) = \begin{cases} 1, & \text{aligned,} \\ \cos(3\pi i/5), & \text{alternating,} \\ 1 + 0.2\cos(3\pi i/5), & \text{mixed.} \end{cases} \tag{19}$$

In the aligned case, all states required downward corrections, so generalization helped. The alternating errors required opposite corrections at consecutive states, making successor shift harmful. The changing pattern combined a common offset with an oscillation that TD removed more slowly, allowing the effect of successor shift to change during learning, from initially helpful to later harmful.

All runs used $\gamma = 0.9$ and $\alpha = 0.05$, with $\eta_{\max} = 0.9$, $\tau = 0.0001$ for adaptive SSC-TD. We obtained the initial weights by solving (19) for $\boldsymbol{w}_0$.

Figure 4 shows that TD learned faster than fixed SSC-TD from aligned errors, whereas suppression helped with alternating errors. In the changing case, the ordering reversed during learning, so neither fixed choice was best throughout. Adaptive SSC-TD reduced error faster late in training.

6.3 Atari Prediction

We evaluated larger-scale prediction on the Atari Prediction Benchmark (Javed et al. 2023), using fixed pretrained Rainbow-DQN policies (Hessel et al. 2018). We followed the setting of Section 4

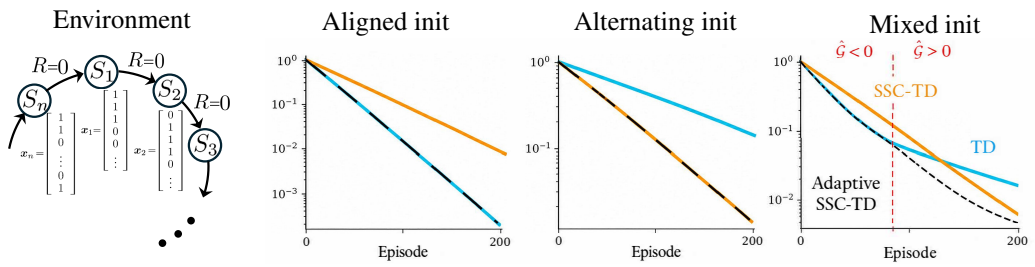

Figure 4: One chain and representation with three initial error patterns. Columns correspond to aligned, alternating, and changing initial errors. The upper panels compare root mean squared value error for TD, SSC-TD, and adaptive SSC-TD.

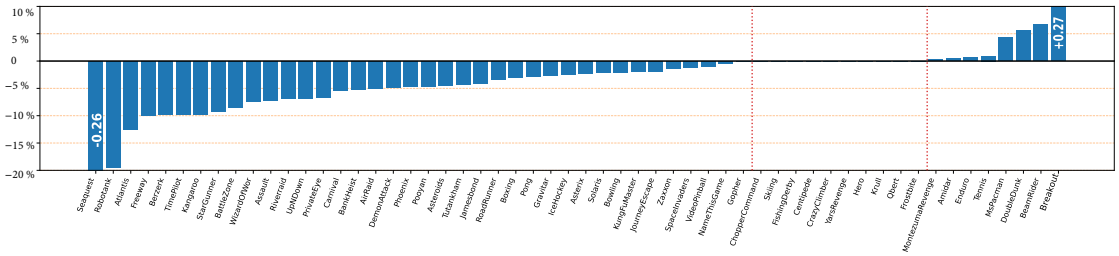

Figure 5: Lifetime MSE comparison between adaptive SSC-TD and TD on the Atari Prediction benchmark, with $\alpha = 10^{-5}$ for both. The y-axis shows percentage differences relative to TD: negative values favor adaptive SSC-TD; positive values favor TD.

of (Javed et al. 2024), with linear function approximation over 201,619 features, $\gamma = 0.98$, and a lifetime of $T = 210{,}000$ steps, trained in a single stream with no replay. We evaluated each prediction against the realized return $G_t = \sum_{\tau=t}^{T_t-1} \gamma^{\tau-t} r_\tau$, where T_t was the earlier of episode termination and the end of the lifetime. Performance was measured by $\text{Lifetime MSE}(T) \overset{\text{def}}{=} \frac{1}{T}\sum_{t=0}^{T-1} \big(V_{\boldsymbol{w}_t}(s_t) - G_t\big)^2$.

We compared TD, fixed and adaptive SSC-TD, and Constrained TD (Durugkar & Stone 2017). Constrained TD used $\Delta \boldsymbol{w}_s^{\text{CTD}} = \alpha\delta_s\big(\boldsymbol{x}_s - c_s \boldsymbol{x}_{s^+}/\|\boldsymbol{x}_{s^+}\|^2\big)$, with the projection omitted at terminal transitions. Unlike SSC-TD, it did not rescale the projected update to TD update's norm. Implementation details are specified in Appendix D.3. At $\alpha = 10^{-5}$, adaptive SSC-TD reduced lifetime MSE relative to TD on 38 of 56 games, tied on 10, and increased it on 8. (Figure 5). Further details and more results are reported in Appendix D.4.

6.4 Stability on Baird's Counterexample

We tested whether controlling successor shift prevented divergence on Baird's counterexample (Baird 1995). We compared TD, SSC-TD over a range of fixed η, and Constrained TD. To separate update dynamics from sampling noise, we applied each method's exact expected update. The results show that TD diverges, whereas SSC-TD remained bounded for every tested $\eta \geqslant 0.26$. Constrained TD also remained bounded (Durugkar & Stone 2017) here. Details are given in Appendix D.5.

7 Related Work

That a TD update moves the target it was formed from has been recognized for a long time, and a common response is to slow the target rather than to change the update. Delayed target networks hold the bootstrap parameters fixed for a number of steps (Mnih et al. 2015), and target-based TD keeps a separate set of target parameters and proves convergence for several ways of updating the two parameter vectors ((Lee & He 2019). Fixed-horizon methods avoid bootstrapping from the same horizon's value function, since each horizon bootstraps from a different value function (De Asis et al. 2020). These methods change wither the state or weight-recency that the bootstrap target is computed from. SSC-TD instead keeps the TD target, and changes the resulting update in a direction to suppress the successor shift.

Two methods act on the update itself and are the closest to SSC-TD. Constrained TD projects the TD update onto the subspace orthogonal to the successor's features, removing the successor-aligned component entirely, and shows that this prevents divergence on its Baird example (Durugkar & Stone 2017). SSC-TD, in contrast, permits partial suppression and restores TD update's norm. The adaptive rule then estimates the effect of TD's successor shift on squared value error, and activates suppression when the estimate indicates harm. HR-TD (Durugkar et al. 2019) instead adds a weight-dependent regularizer based on the current and successor features. This regularizer does not directly constrain successor shift.

A different response is to make the target's dependence on the parameters explicit rather than to suppress it. The residual-gradient method differentiates through the target and descends the mean squared Bellman error, and the same work interpolates between that update and TD (Baird 1995). On a deterministic transition, its update is $\alpha\delta_s(\boldsymbol{x}_s - \gamma\boldsymbol{x}_{s^+})$. These methods however suffer from froward flow of information, and are generally slower than TD.

Generalization and bootstrapping together are also known to be involved in divergence. Achiam et al. (2019) characterize divergence in deep Q-learning through the combined effects of generalization and bootstrapping on the update dynamics. Theirs is an account of the aggregate dynamics. Successor-error amplification is one specific path within such dynamics. Our local signed error-change measure is distinct from a condition for divergence.

8 Discussion and Limitations

Generalization is central to the success of learning in large state spaces because it allows experience at one state to improve predictions at others. Its benefits, however, need not be uniform across states. While generalization to previous states helps backward propagation of information, our analysis identifies a particular difficulty with generalization to the successor state in TD learning, where generalization can amplify the same successor value error that entered the bootstrap target. We isolated that component, showed that under simple conditions it makes the complete successor shift harmful on average, and is less favorable than predecessor generalization. We then gave a return-based criterion for adapting suppression and an update that suppresses successor shift while preserving TD's parameter-step norm. Successor-error amplification is a result of bootstrapping, and does not arise in Monte Carlo learning, whose targets do not bootstrap. We hope this distinction will guide future theory and methods for better understanding and shaping update-generalization.

An important limitation of SSC-TD, also relevant to Constrained TD, is that the true values may not be a fixed point under stochastic transitions, even when representable. Correlation between the sampled TD error and the successor-dependent update direction can produce a nonzero expected update at the true values where expected TD errors are zero. A possible correction could use two independent transitions from the same state: one for the TD error, the other for projection and norm restoration in (13), following the double-sampling principle (Baird 1995). Our analysis has focused on deterministic problems, and we leave the development of such correction to future work.

The adaptive rule is a discrete decision, ans is based on effect of TD's complete successor shift, rather than the effect of switching from TD to SSC-TD. A global coefficient can hide differences between states. These limitations motivate state-dependent suppression and criteria that account for the actual update and uncertainty in its estimated utility.

The other limitations are narrower. We have no general convergence result for SSC-TD. With nonlinear approximation, replacing features by value gradients preserves the norm of the TD semi-gradient update, but the prediction-change formulas in (14) hold only to first order in the step size. Our experiments are limited to linear function approximation. Finally, SSC-TD requires additional feature inner products, and a nonlinear implementation also requires the successor value gradient.

Several directions follow. The analysis should be extended to eligibility traces, where the returning path spans more than one transition. Successor shift could also be controlled by choosing when to bootstrap rather than by projecting the update, holding λ high while the eligibility trace and the next feature vector overlap strongly and lowering it once they separate. Another useful next step is to characterize the decay of the overall value-error loss for TD and SSC-TD on deterministic chains.

AI USE STATEMENT

In this work, we used generative AI tools for coding assistance, writing improvement, including improving formulation of mathematical claims and proofs, and literature review. We have not used generative AI tools to propose or refine hypotheses, provide critical ingredients for proving mathematical claims, design or provide feedback on research methodology or experiments, help develop theoretical models or conceptual frameworks, or interpret results. Other potential uses of generative AI tools are not applicable to this work; examples include clean and reformat dataset, support qualitative and thematic data analysis, generate synthetic data sets, and assist with translation. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Achiam, J., Knight, E., & Abbeel, P. (2019). Towards characterizing divergence in deep Q-learning. *arXiv preprint arXiv:1903.08894*.

Baird, L. C., III. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning*, 30–37. Morgan Kaufmann.

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-13*(5), 834–846.

Bengio, E., Pineau, J., & Precup, D. (2020). Interference and generalization in temporal difference learning. In *Proceedings of the 37th International Conference on Machine Learning*, PMLR 119, 767–777.

De Asis, K., Chan, A., Pitis, S., Sutton, R. S., & Graves, D. (2020). Fixed-horizon temporal difference methods for stable reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(4), 3741–3748.

Durugkar, I., & Stone, P. (2017). TD learning with constrained gradients. In *Proceedings of the Deep Reinforcement Learning Symposium, NIPS 2017*.

Durugkar, I., Liu, B., & Stone, P. (2019). HR-TD: A regularized TD method to avoid overgeneralization. *OpenReview manuscript, ICLR 2019 submission*.

Fujita, Y., Nagarajan, P., Kataoka, T., & Ishikawa, T. (2021). ChainerRL: A deep reinforcement learning library. *Journal of Machine Learning Research*, *22*(77), 1–14.

Ghiassian, S., Rafiee, B., & Sutton, R. S. (2017). A first empirical study of emphatic temporal difference learning. *arXiv preprint arXiv:1705.04185*.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M. G., & Silver, D. (2018). Rainbow: Combining improvements in deep reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1), 3215–3222.

Javed, K., Shah, H., Sutton, R. S., & White, M. (2023). Scalable real-time recurrent learning using columnar-constructive networks. *Journal of Machine Learning Research*, *24*(256), 1–34.

Javed, K., Sharifnassab, A., & Sutton, R. S. (2024). SwiftTD: A fast and robust algorithm for temporal difference learning. *Reinforcement Learning Journal*, *2*, 840–863.

Kumar, A., Agarwal, R., Ma, T., Courville, A., Tucker, G., & Levine, S. (2022). DR3: Value-based deep reinforcement learning requires explicit regularization. In *International Conference on Learning Representations*.

Lee, D., & He, N. (2019). Target-based temporal-difference learning. In *Proceedings of the 36th International Conference on Machine Learning*, PMLR 97, 3713–3722.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou,

I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

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

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). The MIT Press.

A Predecessor–Successor Comparison

We prove (10) for the exact squared-error changes. Consider the deterministic chain of Section 3, with constant consecutive feature overlap $c > 0$. Hold $\boldsymbol{w}$ fixed and write $e(u) = V_{\boldsymbol{w}}(u) - V^{\pi}(u)$. All expectations are under the on-policy stationary distribution d^{π}, with $\mathbb{E}_{\pi}[e(s)^2] < \infty$.

The deterministic Bellman equation gives

$$\delta_s = \gamma e(s^+) - e(s). \tag{20}$$

For average-reward TD, this identity holds with $\gamma = 1$ after subtracting the true average reward and choosing a fixed differential value function. Since $\boldsymbol{x}_{s^-}^{\top}\boldsymbol{x}_s = \boldsymbol{x}_{s^+}^{\top}\boldsymbol{x}_s = c$, the TD update gives

$$\Delta_{\Delta \boldsymbol{w}_s} e(s^-) = \Delta_{\Delta \boldsymbol{w}_s} e(s^+) = \alpha c \delta_s. \tag{21}$$

Stationarity at the fixed weights gives

$$\mathbb{E}_{\pi}[e(s)e(s^+)] = \mathbb{E}_{\pi}[e(s^-)e(s)], \tag{22}$$

$$\mathbb{E}_{\pi}[e(s^+)^2] = \mathbb{E}_{\pi}[e(s^-)^2]. \tag{23}$$

Then,

$$\begin{aligned}
\mathcal{G}_{\mathrm{succ}} - \mathcal{G}_{\mathrm{pre}} &= \frac{1}{2}\mathbb{E}_{\pi}\Big[\big(e(s^+) + \Delta_{\Delta \boldsymbol{w}_s} e(s^+)\big)^2 - e(s^+)^2\Big] \\
&\quad - \frac{1}{2}\mathbb{E}_{\pi}\Big[\big(e(s^-) + \Delta_{\Delta \boldsymbol{w}_s} e(s^-)\big)^2 - e(s^-)^2\Big] \\
&= E_{s\sim d^{\pi}}\left[e(s^+)\Delta_{\Delta \boldsymbol{w}_s} e(s^+) + \frac{1}{2}\Delta_{\Delta \boldsymbol{w}_s} e(s^+)^2\right] \\
&\quad - E_{s\sim d^{\pi}}\left[e(s^-)\Delta_{\Delta \boldsymbol{w}_s} e(s^-) + \frac{1}{2}\Delta_{\Delta \boldsymbol{w}_s} e(s^-)^2\right] \\
&= \mathbb{E}_{\pi}\big[e(s^+)\Delta_{\Delta \boldsymbol{w}_s} e(s^+) - e(s^-)\Delta_{\Delta \boldsymbol{w}_s} e(s^-)\big] \\
&= \alpha c\, \mathbb{E}_{\pi}\big[\big(e(s^+) - e(s^-)\big)\big(\gamma e(s^+) - e(s)\big)\big] \\
&= \alpha\gamma c\, \mathbb{E}_{\pi}\big[e(s^+)^2 - e(s^-)e(s^+)\big] \\
&= \frac{\alpha\gamma c}{2}\mathbb{E}_{\pi}\Big[\big(e(s^+) - e(s^-)\big)^2\Big] \geqslant 0,
\end{aligned}$$

where the first equality is by definition (5), the third and equality is due to (21), the fourth equality follows from (20) and (21), the fifth equality is from (22), and the last equality from (23)

This proves (10). The ordering thus holds for every $\alpha > 0$. For $\gamma > 0$, equality holds only if $e(s^+) = e(s^-)$ almost surely.

B Derivation of SSC-TD

Fix a transition (s, r, s^+) with nonzero $\|\boldsymbol{x}_s\|$ and $\|\boldsymbol{x}_{s^+}\|$. For $\alpha > 0$ and $\eta \in [0, 1)$, the objective in (11) is a strictly convex quadratic function. Its unique minimizer satisfies

$$\Delta \boldsymbol{w} + \frac{\eta}{1-\eta}\frac{\boldsymbol{x}_{s^+}^{\top}\Delta \boldsymbol{w}}{\|\boldsymbol{x}_{s^+}\|^2}\boldsymbol{x}_{s^+} = \alpha\delta_s \boldsymbol{x}_s. \tag{24}$$

Taking the inner product of both sides with $\boldsymbol{x}_{s^+}$ gives $\boldsymbol{x}_{s^+}^\top \Delta \boldsymbol{w} = \alpha \delta_s (1-\eta) c_s$. Substituting into (24) gives (12). Its squared norm is

$$\|\Delta \boldsymbol{w}\|^2 = \alpha^2 \delta_s^2 \left(\|\boldsymbol{x}_s\|^2 - \eta(2-\eta) \frac{c_s^2}{\|\boldsymbol{x}_{s^+}\|^2} \right) = \alpha^2 \delta_s^2 \|\boldsymbol{x}_s\|^2 n_s^2 = n_s^2 \|\Delta \boldsymbol{w}_s^{\mathrm{TD}}\|^2.$$

Since $n_s^2 = 1 - \rho_s^2 + (1-\eta)^2 \rho_s^2 \geqslant (1-\eta)^2 > 0$, division by n_s restores TD's norm and gives (13). When $\delta_s = 0$, both updates are zero.

Taking inner products with the two features gives (14). Both fractions are positive, equal one at $\eta = 0$, and have derivatives

$$\frac{\partial}{\partial \eta} \left(\frac{1 - \eta \rho_s^2}{n_s} \right) = -\frac{\eta \rho_s^2 (1 - \rho_s^2)}{n_s^3},$$
$$\frac{\partial}{\partial \eta} \left(\frac{1 - \eta}{n_s} \right) = -\frac{1 - \rho_s^2}{n_s^3}.$$

They are therefore nonincreasing and lie in $(0, 1]$. For fixed $|\rho_s| < 1$, n_s approaches $\sqrt{1 - \rho_s^2} > 0$ as η approaches one, so the successor fraction tends to zero.

At $\eta = 0$, the successor fraction has derivative $-(1 - \rho_s^2)$, while the current-state fraction has zero first derivative and second derivative $-\rho_s^2(1 - \rho_s^2)$. For $0 < |\rho_s| < 1$, these give the first-order reduction in the successor fraction and second-order reduction in the current-state fraction.

C Implementation of Adaptive SSC-TD

In this appendix, we extend Lemma 1 to changing weights, prove the extended lemma, and give the complete algorithm.

During learning, let $\boldsymbol{w}_t$ denote the weights before the update at time t, and let c_t and δ_t be feature correlation and TD error computed at $\boldsymbol{w}_t$. Consecutive TD errors are then evaluated at different weights. For $t \geqslant 1$, define the mixed TD error

$$\tilde{\delta}_t \stackrel{\text{def}}{=} r_t + \gamma V_{\boldsymbol{w}_t}(s_{t+1}) - V_{\boldsymbol{w}_{t-1}}(s_t), \tag{25}$$

which evaluates s_t at the same weights as the bootstrap term of δ_{t-1}. At fixed weights, $\tilde{\delta}_t = \delta_t$.

Lemma 2 (Generalization of Lemma 1 to Changing Weights)**.** *Consider an episode $s_0, r_0, \ldots, s_T$, with $G_T = 0$ and $V_{\boldsymbol{w}}(s_T) = 0$ for every $\boldsymbol{w}$. Let $h_0 = 0$ and $h_{k+1} = \gamma h_k + c_k \delta_k$. Then,*

$$\sum_{t=0}^{T-1} c_t \delta_t \big(V_{\boldsymbol{w}_t}(s_{t+1}) - G_{t+1} \big) = -\sum_{k=1}^{T-1} \tilde{\delta}_k h_k. \tag{26}$$

In particular, Lemma 1 is the special case of fixed weights.

Proof. The recursive definition, $h_{k+1} = \gamma h_k + c_k \delta_k$, implies that for any $k \geqslant 0$,

$$h_k = \sum_{t=0}^{k-1} \gamma^{k-t-1} c_t \delta_t \tag{27}$$

Expanding (25), consecutive prediction terms cancel telescopically, giving

$$\sum_{k=t+1}^{T-1} \gamma^{k-t-1} \tilde{\delta}_k = \sum_{k=t+1}^{T-1} \gamma^{k-t-1} r_k - V_{\boldsymbol{w}_t}(s_{t+1}) = G_{t+1} - V_{\boldsymbol{w}_t}(s_{t+1}). \tag{28}$$

Multiplying by $-c_t \delta_t$, summing over t, and exchanging the finite sums gives

$$\begin{aligned}
\sum_{t=0}^{T-1} c_t \delta_t \big(V_{\boldsymbol{w}_t}(s_{t+1}) - G_{t+1} \big) &= -\sum_{t=0}^{T-1} c_t \delta_t \sum_{k=t+1}^{T-1} \gamma^{k-t-1} \tilde{\delta}_k \\
&= -\sum_{k=1}^{T-1} \tilde{\delta}_k \sum_{t=0}^{k-1} \gamma^{k-t-1} c_t \delta_t \\
&= -\sum_{k=1}^{T-1} \tilde{\delta}_k h_k.
\end{aligned}$$

Algorithm 1 Adaptive SSC-TD, linear case, streaming setting

Parameters step size α, discount γ, maximum suppression $\eta_{\max} \in [0, 1)$, averaging rate $\tau \ll 1 - \gamma$
Initialize $\boldsymbol{w}$ arbitrarily, and $\hat{\mathcal{G}} \leftarrow 0$, $h \leftarrow 0$, $v \leftarrow 0$
for $t = 0, 1, 2, \ldots$ **do**
  observe R_t and S_{t+1} ▷ $\boldsymbol{x}_{t+1} = \mathbf{0}$ if S_{t+1} is terminal
  $\delta \leftarrow R_t + \gamma\, \boldsymbol{x}_{t+1}^\top \boldsymbol{w} - \boldsymbol{x}_t^\top \boldsymbol{w}, \quad c \leftarrow \boldsymbol{x}_t^\top \boldsymbol{x}_{t+1}$
  $\tilde{\delta} \leftarrow R_t + \gamma\, \boldsymbol{x}_{t+1}^\top \boldsymbol{w} - v$ ▷ $v = \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$
  $\hat{\mathcal{G}} \leftarrow (1 - \tau)\hat{\mathcal{G}} - \tau\, \tilde{\delta}\, h$
  $h \leftarrow \gamma h + c\, \delta, \quad v \leftarrow \boldsymbol{x}_{t+1}^\top \boldsymbol{w}$
  $\eta \leftarrow \eta_{\max}$ if $\hat{\mathcal{G}} > 0$, else $\eta \leftarrow 0$ ▷ (16)
  if $c \neq 0$ **then**
    $n \leftarrow \sqrt{1 - \eta(2 - \eta)c^2 / \big(\|\boldsymbol{x}_t\|^2 \|\boldsymbol{x}_{t+1}\|^2\big)}$
    $\boldsymbol{w} \leftarrow \boldsymbol{w} + \dfrac{\alpha\delta}{n}\Big(\boldsymbol{x}_t - \eta \dfrac{c}{\|\boldsymbol{x}_{t+1}\|^2} \boldsymbol{x}_{t+1}\Big)$ ▷ (13)
  else
    $\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha\delta\, \boldsymbol{x}_t$
  end if
  if S_{t+1} is terminal **then**
    $h \leftarrow 0$ ▷ the next step starts a new episode
  end if
end for

where the first equality is due to (28), the second equality is by swapping the finite sums, and the last equality is from (27). This establishes (26). At fixed weights, $\tilde{\delta}_k = \delta_k$, giving Lemma 1. □

For online estimation, we replace $-\delta_t h_t$ in the moving average of Section 5 by $-\tilde{\delta}_t h_t$:

$$\hat{\mathcal{G}} \leftarrow (1 - \tau)\hat{\mathcal{G}} - \tau \tilde{\delta}_t h_t, \qquad \tau \in (0, 1]. \tag{29}$$

We initialize $\hat{\mathcal{G}} = 0$ and take $\tilde{\delta}_0 = 0$. Equation (26) is exact for complete episode sums; exponential averaging gives a delayed estimate of the evolving criterion. We choose $\tau \ll 1 - \gamma$ so the averaging window exceeds the discounted return horizon. The trace h is reset at the start of each episode, while $\hat{\mathcal{G}}$ is retained across episodes. In a continuing task, $T = \infty$ and the trace is never reset.

Algorithm 1 gives the complete algorithm. It computes (25) using one saved prediction, $v = V_{\boldsymbol{w}_{t-1}}(s_t)$, rather than storing the previous weights. Since $\tilde{\delta}_t = \delta_t + \boldsymbol{x}_t^\top \boldsymbol{w}_t - \boldsymbol{x}_t^\top \boldsymbol{w}_{t-1}$, the estimate requires no inner products beyond those of SSC-TD. The estimate is updated before the parameter update, and its sign determines the suppression used in that update.

D Experimental Details

This appendix provides the experimental details for Section 6. Adaptive SSC-TD follows Algorithm 1.

D.1 Mountain Car and CartPole

Mountain Car and CartPole use the dynamics of (Sutton & Barto 2018) and (Barto et al. 1983), with a reward of -1 per step. Both problems are deterministic, and we fixed the start state at $[-0.5, 0.0]$ for Mountain Car and $[0.0, 0.0, 0.05, 0.0]$ for CartPole, so every episode traverses the same sequence of states. The Mountain Car policy accelerates in the direction of the current velocity, taking the positive action when the velocity is zero Ghiassian et al. (2017). The CartPole policy wis a fixed hand-coded linear threshold controller on the pole angle and angular velocity: it applied action 1 when $\theta + 0.5\dot{\theta} \geqslant 0$ and action 0 otherwise.

The feature representation was sparse tile coding over the raw observation variables in both domains. For Mountain Car, we used 16 tilings with an 8×8 grid per tiling, yielding $16 \cdot 8^2 = 1024$ features. For CartPole, we used 64 tilings with an $16 \times 16 \times 16 \times 16$ grid per tiling, yielding $64 \cdot 16^4 = 4194304$ features. In both cases, active features were unnormalized and the TD weights were initialized to zero.

Because both chains are deterministic and terminate, $V^\pi(s_t) = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k}$ is the discounted sum of the rewards actually observed from S_t onward. We computed it once per problem by a backward pass over the trajectory. The value errors $e = V_{\boldsymbol{w}} - V^\pi$ are therefore exact rather than estimated, and the first term of (7) is zero at every transition rather than only in expectation.

We trained with TD(0) at $\alpha = 0.01$ for 400 episodes. Every 100 episodes we froze the weights and swept the frozen weights over the whole trajectory, accumulating $\gamma c_s e(s^+)^2$, $-c_s e(s) e(s^+)$ and $-\|\boldsymbol{x}_{s^+}\|^2 e(s^+)\delta_{s^+}$ over all transitions with a nonterminal successor. The three coefficients of (18) are the ratios of the first two accumulators, and of their sum, to the third. No learning occurs during a measurement sweep, so the coefficients describe the weights at that checkpoint. The reported feature overlap is the mean of ρ_s over the same transitions.

D.2 Ring

We used the 20-state cycle and feature map of Section 6.2. The feature matrix is invertible, and we obtained the initial weights by solving (19) exactly. Only the initial-error pattern differed across the three conditions. States were visited in cycle order, with one update per transition. All methods used $\gamma = 0.9$ and $\alpha = 0.05$. Fixed SSC-TD used $\eta = 0.9$. Adaptive SSC-TD used $\eta_{\max} = 0.9$. After every sweep, we evaluated RMSVE uniformly over all 20 states at fixed weights. The ring and classic-control runs were deterministic, so no uncertainty bands are shown.

D.3 Atari Prediction

We used the Atari Prediction Benchmark of (Javed et al. 2023), in the configuration of Section 4 of (Javed et al. 2024). Actions are chosen by fixed pretrained Rainbow-DQN policies (Hessel et al. 2018) taken from the ChainerRL model zoo (Fujita et al. 2021), so the prediction problem is on-policy with respect to a fixed policy and the agent never acts. Each run is a single stream of $T = 210{,}000$ steps with no replay buffer and no target network, with $\gamma = 0.98$.

We compared TD(0), SSC-TD at fixed $\eta \in \{0.5, 0.66, 0.8, 0.9, 0.99\}$, adaptive SSC-TD $\eta_{\max} \in \{0.8, 0.9, 0.99\}$, and Constrained TD (Durugkar & Stone 2017). We further performed hyperparameters tuning, for finding the best set of parameters which were used both in SSC-TD and adaptive variation. All methods used the same step sizes $\alpha \in \{3 \times 10^{-6}, 10^{-5}\}$ and the same policy, seeds and feature construction, and differ only in the update rule. Adaptive SSC-TD used $\eta_{\max} = 0.9$, and $\tau = 0.0001$, chosen in preliminary experiments and then held fixed across all games. For further fine-tuning $\eta_{\max}$, we also obtained results for $\eta_{\max} = 0.99$, which is considered maximum amplification error suppression factor. Constrained TD used $\Delta \boldsymbol{w}_s^{\text{constrained}-\text{TD}} = \alpha\delta_s\big(\boldsymbol{x}_s - c_s \boldsymbol{x}_{s^+}/\|\boldsymbol{x}_{s^+}\|^2\big)$, with the projection omitted at terminal transitions.

Lifetime MSE is computed against the realized return $G_t = \sum_{\tau=t}^{T_t-1} \gamma^{\tau-t} r_\tau$, where T_t is the earlier of the next episode termination and the end of the lifetime. Each prediction $V_{\boldsymbol{w}}(s)$ is recorded before the update at s, so no target uses information from its own update. We evaluated multiple hyperparameter configurations and report results for the selected configuration; each game is evaluated on a single fixed trajectory. We use one run per game, evaluated on a fixed 210,000-step trajectory generated for that game. Appendix D.4 gives the spread.

D.4 Additional Atari Results

Figure 5 in the main text reports $\alpha = 10^{-5}$. Table 1 collects the per-game lifetime MSE of every method. Tables 1–3 report lifetime MSE across all 56 Atari environments for TD, fixed SSC-TD, Adaptive SSC-TD, and Constrained TD at $\alpha = 10^{-5}$. Constrained TD improves over TD on 20 of the 56 environments and achieves the lowest lifetime MSE among the four methods on 14 environments. Its performance, however, is highly game dependent and can degrade substantially on some environments. Bold game names mark environments where Adaptive SSC-TD improves over both

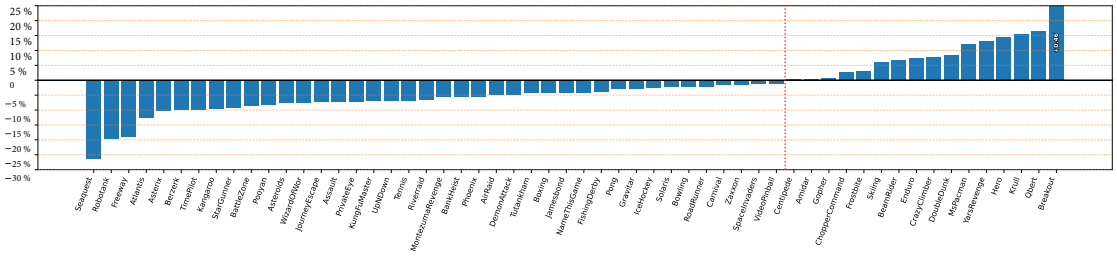

Figure 6: Lifetime MSE comparison between SSC-TD and TD on the Atari Prediction benchmark, with $\alpha = 10^{-5}$ for both. The y-axis shows percentage differences relative to TD.

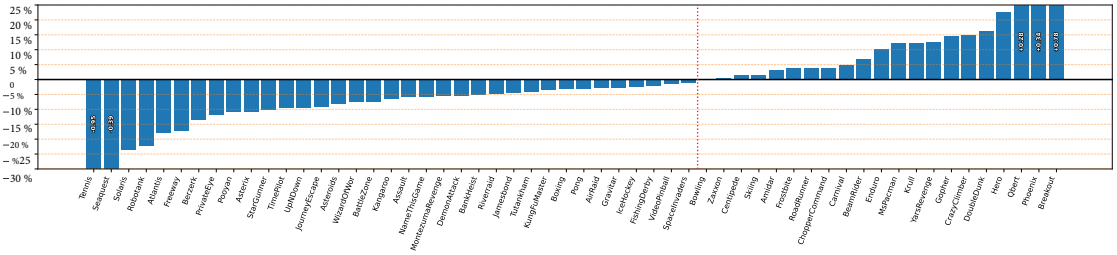

Figure 7: Average MSE over final 20,000 steps of SSC-TD compared against that of TD on the Atari Prediction benchmark, with $\alpha = 10^{-5}$ for both. The y-axis shows percentage differences relative to TD.

TD and fixed SSC-TD, while bold numerical entries denote the minimum lifetime MSE among all four methods in each row.

Adaptive SSC-TD improved on TD on 27 of 56 games at $\alpha = 3 \times 10^{-6}$ and on 38 of 56 at $\alpha = 10^{-5}$. Fixed SSC-TD at the best single η improved on 40 of 56 with no ties, which is the comparison that isolates the value of adapting. The SSC-TD lifetime performance is demonstrated in Figure 6. For fixed SSC-TD, lifetime MSE was lower than TD on 40 of 56 Atari games. The largest improvement occurred on Seaquest, where SSC-TD reduced lifetime MSE by 26.3%, while the largest degradation occurred on Breakout, where lifetime MSE increased by 45.9%.

In addition to lifetime MSE over the full 210,000-step trajectory, we evaluate MSE over the final 20,000 steps as a measure of late-stage performance, allowing us to assess whether improvements persist after the initial learning transient(Figures 7 and 8). Over the final 20k steps, SSC-TD improved over TD on $35/56$ games, compared with $40/56$ over the full lifetime. Despite this reduction in win count, the mean per-game MSE ratio remained essentially unchanged at 0.980, indicating that the average improvement persists late in training, although with greater variability across games. On several games the late-stage gains were substantial; for example, SSC-TD reduced MSE by about 39.1% on Seaquest, 22.2% on Robotank, 17.9% on Atlantis, and 17.3% on Freeway relative to TD.

Over the final 20,000 steps, adaptive SSC-TD improved over TD on $37/56$ games and 10 games ties, compared with $38/56$ over the full lifetime. The mean per-game MSE ratio decreased from 0.969 over the full trajectory to 0.956 over the final window, corresponding to an increase in the average MSE reduction from 3.15% to 4.36%. Late-stage gains were substantial on several games, including reductions of 95% on Tennis, 39% on Seaquest, over 20% on Solaris and Robotank, indicating that the benefit of adaptive SSC-TD persists and on average becomes somewhat stronger, late in training.

D.5 Baird's Counterexample

To separate the update dynamics from sampling noise, each method was applied to its exact expected update under the behaviour distribution, taken uniform over the seven states.

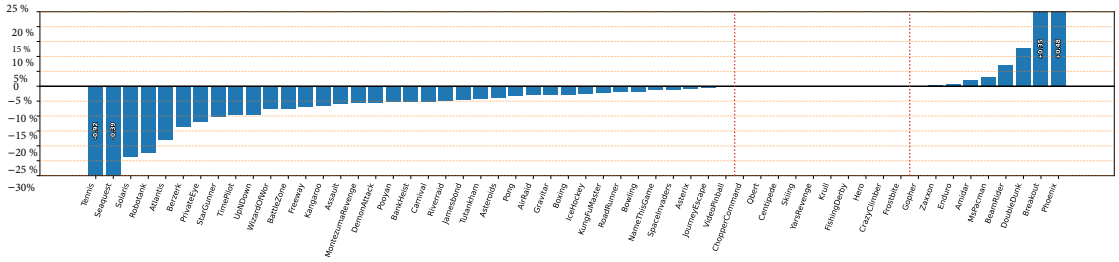

Figure 8: Lifetime MSE of adaptive SSC-TD for final 20,000 steps divided by that of TD on the Atari Prediction benchmark, for $\alpha = 10^{-5}$ for both algorithms.

Table 1: Lifetime MSE by game and method at $\alpha = 10^{-5}$. Lower is better. Bold game names indicate environments where Adaptive SSC-TD has lower lifetime MSE than both TD and fixed SSC-TD ($\eta = 0.9$). The lowest MSE among all four methods in each row is bolded.

Game	TD	SSC-TD ($\eta = 0.9$)	Adaptive SSC-TD	Constrained TD
Seaquest	0.2581371	0.1903164	**0.1903012**	0.3111439
Atlantis	0.2020334	0.1767531	**0.1767242**	0.4933203
StarGunner	0.2374392	0.2156040	**0.2155608**	0.2298734
Riverraid	0.4312033	0.4024526	**0.4012184**	0.6096340
UpNDown	0.2990371	0.2787409	**0.2782706**	0.2936495
Carnival	0.1436888	0.1412810	**0.1359213**	0.1645669
AirRaid	0.0579225	0.0550553	**0.0550410**	0.0585674
DemonAttack	0.3111350	0.2963483	**0.2963305**	0.3735742
RoadRunner	0.0504977	0.0494633	0.0487521	**0.0411576**
Solaris	0.0136201	0.0133250	**0.0133212**	0.0143765
Zaxxon	0.0960027	0.0946937	**0.0946909**	0.1096369
SpaceInvaders	0.3204988	0.3165391	**0.3165129**	0.7196061
Gopher	0.2165532	0.2183451	**0.2164270**	0.2315182
ChopperCommand	0.7621428	0.7822990	**0.7619833**	0.8349909
Robotank	0.0542440	**0.0436732**	0.0436734	0.0607064
Freeway	0.0693852	0.0562752	0.0624796	**0.0555871**
Berzerk	0.1842049	**0.1661782**	0.1661797	0.2250546
TimePilot	0.0932797	**0.0841974**	0.0842044	0.1539440
Kangaroo	0.0908732	0.0820499	0.0820522	**0.0744150**

The importance-sampling ratio is 7 for the solid action and 0 for the dashed action, so only solid transitions contribute and they contribute with total weight $\frac{1}{7} \cdot \frac{1}{7} \cdot 7 = \frac{1}{7}$. Every update therefore has successor state 7, and the expected update is the linear iteration

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha A_\eta \boldsymbol{w}, \qquad A_\eta \stackrel{\text{def}}{=} \frac{1}{7} \sum_{s=1}^{7} \tilde{\boldsymbol{x}}_\eta(s) \big(\gamma \boldsymbol{x}(7) - \boldsymbol{x}(s)\big)^\top, \tag{30}$$

where $\tilde{\boldsymbol{x}}_\eta(s)$ is the factor multiplying $\alpha\delta_s$ in (13), with successor state 7:

$$\tilde{\boldsymbol{x}}_\eta(s) = \frac{1}{n_s} \left(\boldsymbol{x}_s - \eta \frac{c_s}{\|\boldsymbol{x}_7\|^2} \boldsymbol{x}_7 \right), \qquad n_s = \sqrt{1 - \eta(2-\eta)\rho_s^2}.$$

At $\eta = 0$ it is $\boldsymbol{x}(s)$ and the iteration is expected TD. For Constrained TD it is $\boldsymbol{x}(s) - c_s \boldsymbol{x}(7)/\|\boldsymbol{x}(7)\|^2$.

For $s = 1, \ldots, 6$ we have $c_s = \boldsymbol{x}(s)^\top \boldsymbol{x}(7) = 2$ and $\|\boldsymbol{x}(s)\|^2 = \|\boldsymbol{x}(7)\|^2 = 5$, so $\rho_s^2 = 4/25 = 0.16$ at every off-diagonal transition. At $s = 7$ the two feature vectors coincide, so $\rho_7 = 1$ and SSC-TD equals TD at that transition for every η. Suppression therefore acts on six of the seven transitions.

Stability threshold. The iteration (30) diverges when A_η has an eigenvalue with positive real part in a direction that changes the predicted values. The matrix A_η always has one eigenvalue

Table 2: Lifetime MSE by game and method, continued. Lower is better; the lowest MSE among all four methods in each row is bolded.

Game	TD	SSC-TD ($\eta = 0.9$)	Adaptive SSC-TD	Constrained TD
BattleZone	0.0782104	0.0716242	0.0716246	**0.0643826**
WizardOfWor	0.0279710	0.0258551	0.0258808	**0.0222889**
Assault	0.0299199	0.0277288	0.0277488	**0.0249216**
PrivateEye	0.0025431	**0.0023590**	0.0023714	0.0023625
BankHeist	0.0133969	**0.0126629**	0.0126869	0.0136299
Phoenix	0.0695170	**0.0657089**	0.0662833	0.0844795
Pooyan	0.1411291	**0.1294847**	0.1346618	0.1576636
Asteroids	0.2109087	**0.1947842**	0.2016241	0.2032211
Tutankham	0.1159962	**0.1109567**	0.1109573	0.1157024
Jamesbond	0.0458139	0.0438948	0.0439013	**0.0426108**
Boxing	0.5139909	**0.4921134**	0.4985418	0.5226829
Pong	0.0695450	0.0675954	0.0676006	**0.0597398**
Gravitar	0.0262224	0.0254978	0.0255036	**0.0227522**
IceHockey	0.0345171	0.0336365	0.0336579	**0.0287392**
Asterix	0.5895767	**0.5291981**	0.5765742	0.5575998
Bowling	0.0317773	0.0311118	0.0311310	**0.0305000**
KungFuMaster	0.2548596	**0.2370981**	0.2497314	0.3456738
JourneyEscape	0.1004643	0.0930879	0.0985166	**0.0918292**
VideoPinball	0.3473693	**0.3432446**	0.3441080	0.5304386

Table 3: Lifetime MSE by game and method, continued. Lower is better; the lowest MSE among all four methods in each row is bolded.

Game	TD	SSC-TD ($\eta = 0.9$)	Adaptive SSC-TD	Constrained TD
NameThisGame	0.8750000	**0.8388298**	0.8712521	0.9419751
Skiing	**5.5182621**	5.8597455	**5.5182621**	1292.9905709
FishingDerby	1.9380093	**1.8661587**	1.9380679	2.0598137
Centipede	**1.8515350**	1.8557835	1.8516369	2.7878072
CrazyClimber	**7.0708319**	7.6155350	7.0712636	23.7366459
YarsRevenge	**1.9625745**	2.2220060	1.9627382	2.4221125
Hero	**14.4373370**	16.4985028	14.4408646	28.0006951
Krull	**7.3398220**	8.4787641	7.3424220	15.7079678
Qbert	**3.3205688**	3.8669863	3.3217482	6.5826900
Frostbite	**17.3265125**	17.8636563	17.3337847	24.1448280
MontezumaRevenge	0.0004465	0.0004213	0.0004476	**0.0003606**
Amidar	**0.1367659**	0.1373793	0.1373058	0.1624018
Enduro	**1.0679950**	1.1466099	1.0751371	1.6335144
Tennis	0.0001493	0.0001393	0.0001507	**0.0001199**
MsPacman	**0.9558096**	1.0704486	0.9970861	2.4036549
DoubleDunk	**0.0660538**	0.0716725	0.0697522	0.0694593
BeamRider	**0.1199676**	0.1280831	0.1279157	0.1340804
Breakout	**0.1936969**	0.2825305	0.2456857	0.3367655

exactly at zero, whose eigenvector lies in the null space of the feature matrix and therefore leaves all predictions unchanged. Excluding that direction, the largest real part is positive for $\eta < \eta^\star$ and negative for $\eta > \eta^\star$, with

$$\eta^\star \approx 0.259. \tag{31}$$

This is why suppression at $\eta \geqslant 0.26$ is stable, as reported in Section 6.4.