Reinforcement Learning: Basic models and algorithms

Optimal decisions, Part VII

Christos Dimitrakakis

Chalmers

November 20, 2013

The reinforcement learning problem and MDPs

Markov decision processes

- Description of environments
- Solutions to bandit problems

Algorithms for unknown MDPs.

- Stochastic exact algorithms
- Stochastic estimation algorithms
- Online algorithms

The stochastic n-armed bandit problem

$$\mathscr{P} = \{P_i \mid i = 1, \dots, n\}.$$

$$r_t \mid a_t = i \sim P_i.$$

$$\mathbb{E}_\pi U_t = \mathbb{E}_\pi \sum_{k=t}^{T} r_k, \qquad a_t^* \triangleq \max \{\mathbb{E}(r_t \mid a_t = i) \mid i = 1, \dots, n\}.$$

$$\mathscr{P} = \{P_i(\cdot \mid \omega) \mid \omega \in \Omega\}, \qquad r_t \mid a_t = i, \omega^* = \omega \sim P_i(r \mid \omega^*). \tag{1.1}$$

$$\mathbb{E}_\xi^\pi U_t = \mathbb{E}_\xi^\pi \sum_{k=t}^{T} r_k. \tag{1.2}$$

Algorithm 1 Robbins-Monro bandit algorithm

1: **input** Step-sizes $(\alpha_t)_t$, initial estimates $(\mu_{i,0})_i$, policy π.
2: **for** $t = 1, \ldots, T$ **do**
3: Take action $a_t = i$ with probability $\pi(i \mid a_1, \ldots, a_{t-1}, r_1, \ldots, r_{t-1})$.
4: Observe reward r_t.
5: $\mu_{t,i} = \alpha_{i,t} r_t + (1 - \alpha_{i,t})\mu_{i,t-1}$ `// estimation step`
6: $\mu_{t,i} = \mu_{j,t-1}$ for $j \neq i$.
7: **end for**
8: **return** μ_T

Definition 1

ϵ-greedy action selection (w.p. $1 - \epsilon$, select an apparently best action, otherwise a random action)

$$\hat{\pi}^*_\epsilon \triangleq (1 - \epsilon_t)\hat{\pi}^*_t + \epsilon_t \mathcal{U}nif(\mathcal{A}), \tag{1.3}$$

$$\hat{\pi}^*_t(i) = \mathbb{I}\left\{i \in \hat{\mathcal{A}}^*_t\right\} / |\hat{\mathcal{A}}^*_t|, \qquad \hat{\mathcal{A}}^*_t = \arg\max_{i \in \mathcal{A}} \mu_{t,i} \tag{1.4}$$

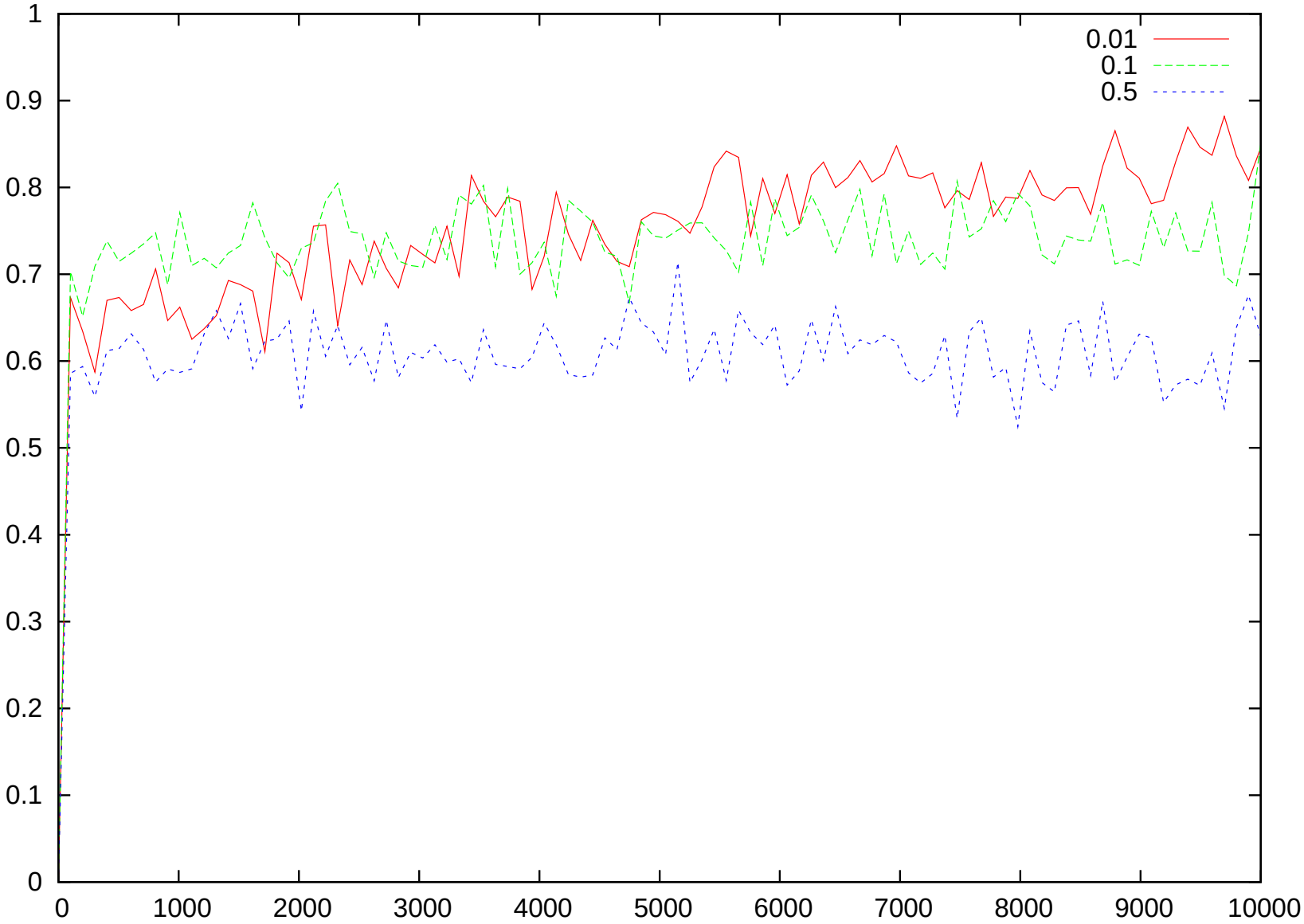

Figure: $\epsilon_t = 0.1$, $\alpha \in \{0.01, 0.1, 0.5\}$.

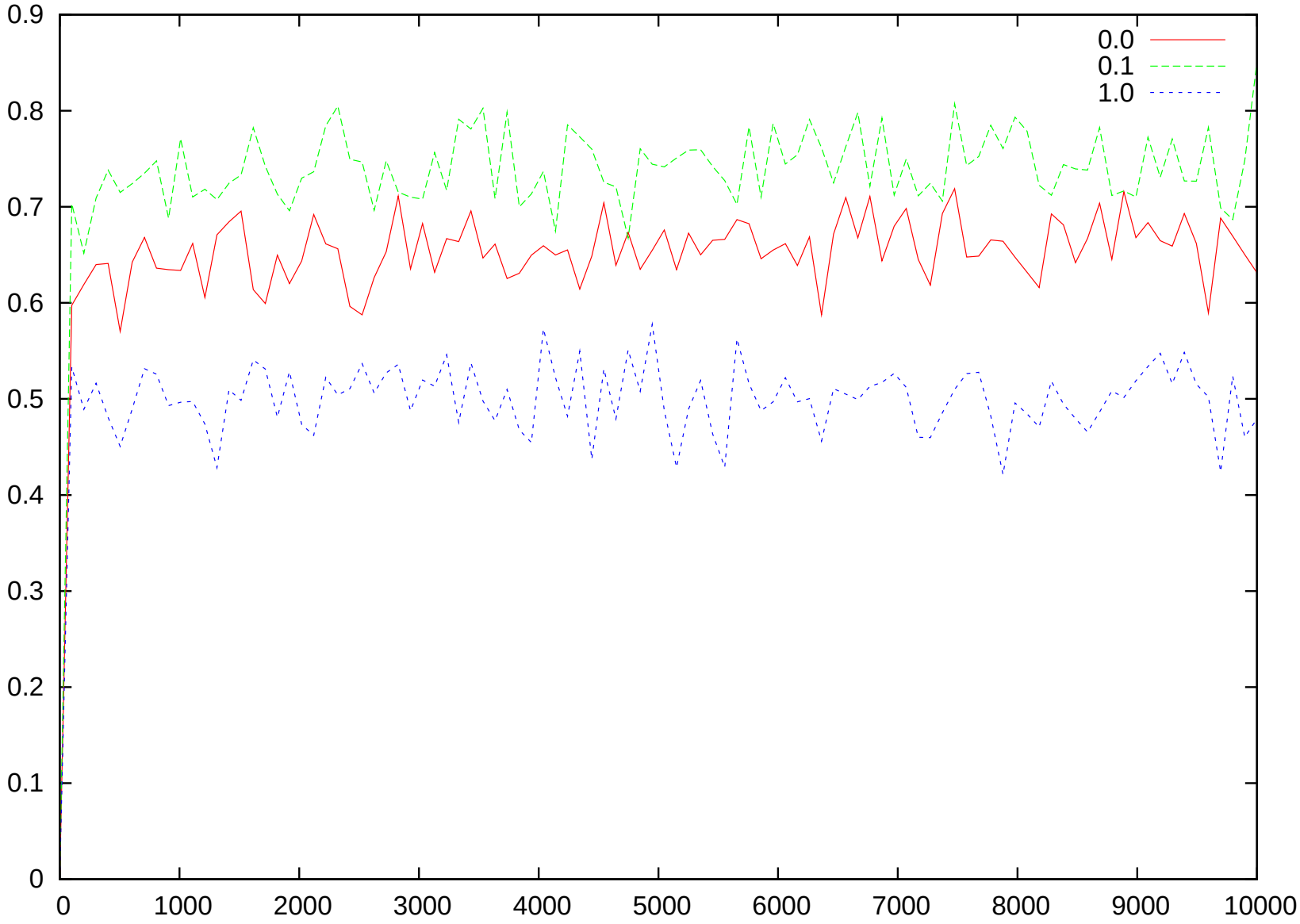

Figure: $\epsilon_t = \epsilon$, $\alpha = 0.1$.

Main idea of the algorithm

- Estimate parameters
- Act according to the estimates

Requirements

- Good estimation procedure.
- Balance estimation with getting rewards!

Consider the algorithm

$$\mu_{t+1} = \mu_t + \alpha_t z_{t+1}. \tag{1.5}$$

Let $h_t = \{\mu_t, z_t, \alpha_t, \ldots\}$ be the history of the algorithm.

Assumption 1

Assume a function $f : \mathbb{R}^n \to \mathbb{R}$ such that:

(i) $f(x) \geq 0$ *for all* $x \in \mathbb{R}^n$.

(ii) *(Lipschitz derivative) f is continuously differentiable and $\exists L > 0$ such that:*

$$\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|, \qquad \forall\, x, y \in \mathbb{R}^n$$

(iii) *(Pseudo-gradient) $\exists c > 0$ such that:*

$$c \|\nabla f(\mu_t)\|^2 \leq -\nabla f(\mu_t)^\top \mathbb{E}(z_{t+1} \mid h_t), \qquad \forall\, t.$$

(iv) $\exists K_1, K_2 > 0$ *such that*

$$\mathbb{E}(\|z_{t+1}\|^2 \mid h_t) \leq K_1 + K_2 \|\nabla f(\mu_t)\|^2$$

Theorem 2

For the algorithm

$$\mu_{t+1} = \mu_t + \alpha_t z_{t+1},$$

where $\alpha_t \geq 0$ *satisfy*

$$\sum_{t=0}^{\infty} \alpha_t = \infty, \qquad \sum_{t=0}^{\infty} \alpha_t^2 < \infty, \tag{1.6}$$

and under Assumption 1, with probability 1:

1. *The sequence* $\{f(\mu_t)\}$ *converges.*
2. $\lim_{t\to\infty} \nabla f(\mu_t) = 0$.
3. *Every limit point* μ^* *of* μ_t *satisfies* $\nabla f(\mu^*) = 0$.

A demonstration

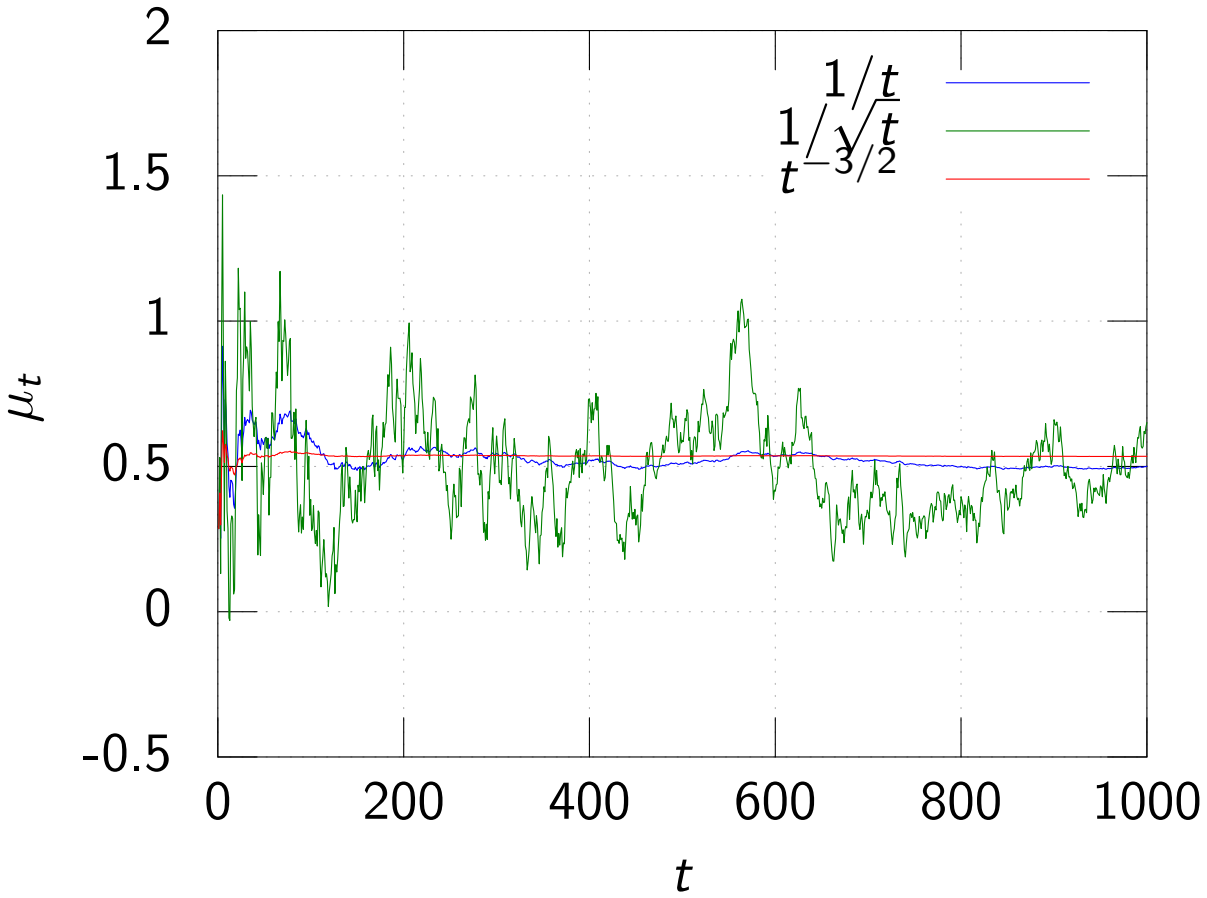

Figure: Estimation of the expectation of $x_t \sim \mathcal{N}(0.5, 1)$ using use three step-size schedules.

Algorithm 2 Generic reinforcement learning algorithm

1: **input** Update-rule $f : \Theta \times \mathcal{S}^2 \times \mathcal{A} \times \mathscr{R} \to \Theta$, initial parameters $\theta_0 \in \Theta$, policy $\pi : \mathcal{S} \times \Theta \to \mathfrak{D}(\mathcal{A})$.
2: **for** $t = 1, \ldots, T$ **do**
3: $\quad a_t \sim \pi(\cdot \mid \theta_t, s_t)$ `// take action`
4: $\quad$ Observe reward r_{t+1}, state s_{t+1}.
5: $\quad \theta_{t+1} = f(\theta_t, s_t, a_t, r_{t+1}, s_{t+1})$ `// update estimate`
6: **end for**

Questions

- What should we estimate?
- What policy should we use?

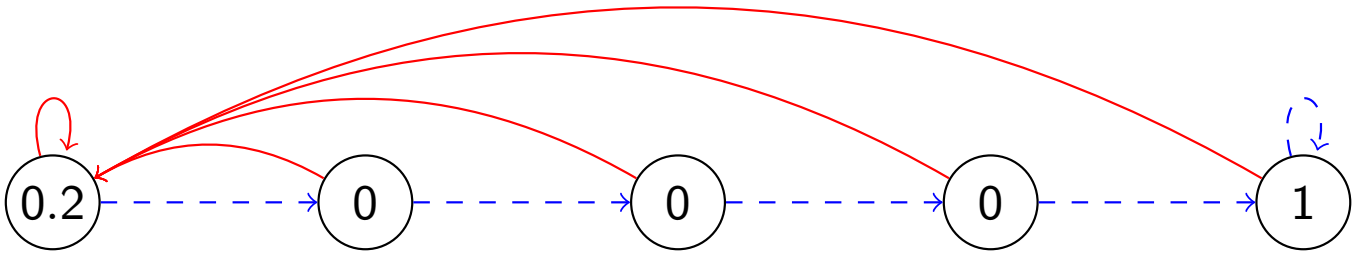

Figure: The chain task

s	s_1	s_2	s_3	s_4	s_5
$V^*(s)$	6.672	7.111	7.689	8.449	9.449
$Q^*(s,1)$	6.622	6.532	6.676	6.866	7.866
$Q^*(s,2)$	6.672	7.111	7.689	8.449	9.449

Table: The chain task's value function for $\gamma = 0.95$

Algorithm 3 Stochastic policy evaluation

input Initial parameters $\boldsymbol{v}_0$, Markov policy π.
for $s \in \mathcal{S}$ **do**
  $s_1 = s$.
  for $k = 1, \ldots, K$ **do**
    Run policy π for T steps.
    Observe utility $U_k = \sum_t r_t$.
    Update estimate $\boldsymbol{v}_{k+1}(s) = \boldsymbol{v}_k(s) + \alpha_k(U_k - \boldsymbol{v}_k(s))$
  end for
end for
return $\boldsymbol{v}_K$

For $\alpha_k = 1/k$ and iterating over all $\mathcal{S}$, this is the same as Monte-Carlo policy evaluation.

Algorithm 4 Approximate policy iteration

input Initial parameters $\boldsymbol{v}_0$, inital Markov policy π_0, stochastic estimator f.
for $i = 1, \ldots, N$ **do**
  Get estimate $\boldsymbol{v}_i = f(\boldsymbol{v}_{i-1}, \pi_{i-1})$.
  Calculate new policy $\pi_i = \arg\max_\pi \mathscr{L}\boldsymbol{v}_i$.
end for

Monte Carlo update

Note that $s_1, \dots, s_T$ contains $s_k, \dots, s_T$.

Algorithm 5 Every-visit Monte-Carlo update

input Initial parameters $\boldsymbol{v}_k$, trajectory $s_1, \dots, s_T$, rewards $r_1, \dots, r_T$ visit counts n.
for $t = 1, \dots, T$ **do**
  $U_t = \sum_{t=1}^{T} r_t$.
  $n_t(s_t) = n_{t-1}(s_t) + 1$
  $\boldsymbol{v}_{t+1}(s_t) = \boldsymbol{v}_t(s) + \alpha_{n_t(s_t)}(s_t)(U_t - \boldsymbol{v}_t(s_t))$
  $n_t(s) = n_{t-1}(s)$, $\boldsymbol{v}_t(s) = \boldsymbol{v}_{t-1}(s)\ \forall s \neq s_t$.
end for
return $\boldsymbol{v}_K$

Example 3

Consider a two-state chain with $\mathbb{P}(s_{t+1} = 1 \mid s_t = 0) = \delta$ and $\mathbb{P}(s_{t+1} = 1 \mid s_t = 1) = 1$, and reward $r(1) = 1$, $r(0) = 0$. Then the every-visit estimate is biased.

Unbiased Monte-Carlo update

Algorithm 6 First-visit Monte-Carlo update

input Initial parameters $\boldsymbol{v}_k$, trajectory $s_1, \ldots, s_T$, rewards $r_1, \ldots, r_T$, visit counts n.
Let $m \in \mathbb{N}^{|\mathcal{S}|}$ be trajectory visit counts.
for $t = 1, \ldots, T$ **do**
    $U_t = \sum_{t=1}^{T} r_t$.
    $n_t(s_t) = n_{t-1}(s_t) + 1$
    $m_t(s_t) = m_{t-1}(s_t) + 1$
    $\boldsymbol{v}_{t+1}(s_t) = \boldsymbol{v}_t(s) + \alpha_{n_t(s_t)}(s_t)(U_t - \boldsymbol{v}_t(s_t))$ if $m_t(s_t) = 1$.
    $n_t(s) = n_{t-1}(s)$, $\boldsymbol{v}_t(s) = \boldsymbol{v}_{t-1}(s)$ otherwise
end for
return $\boldsymbol{v}_K$

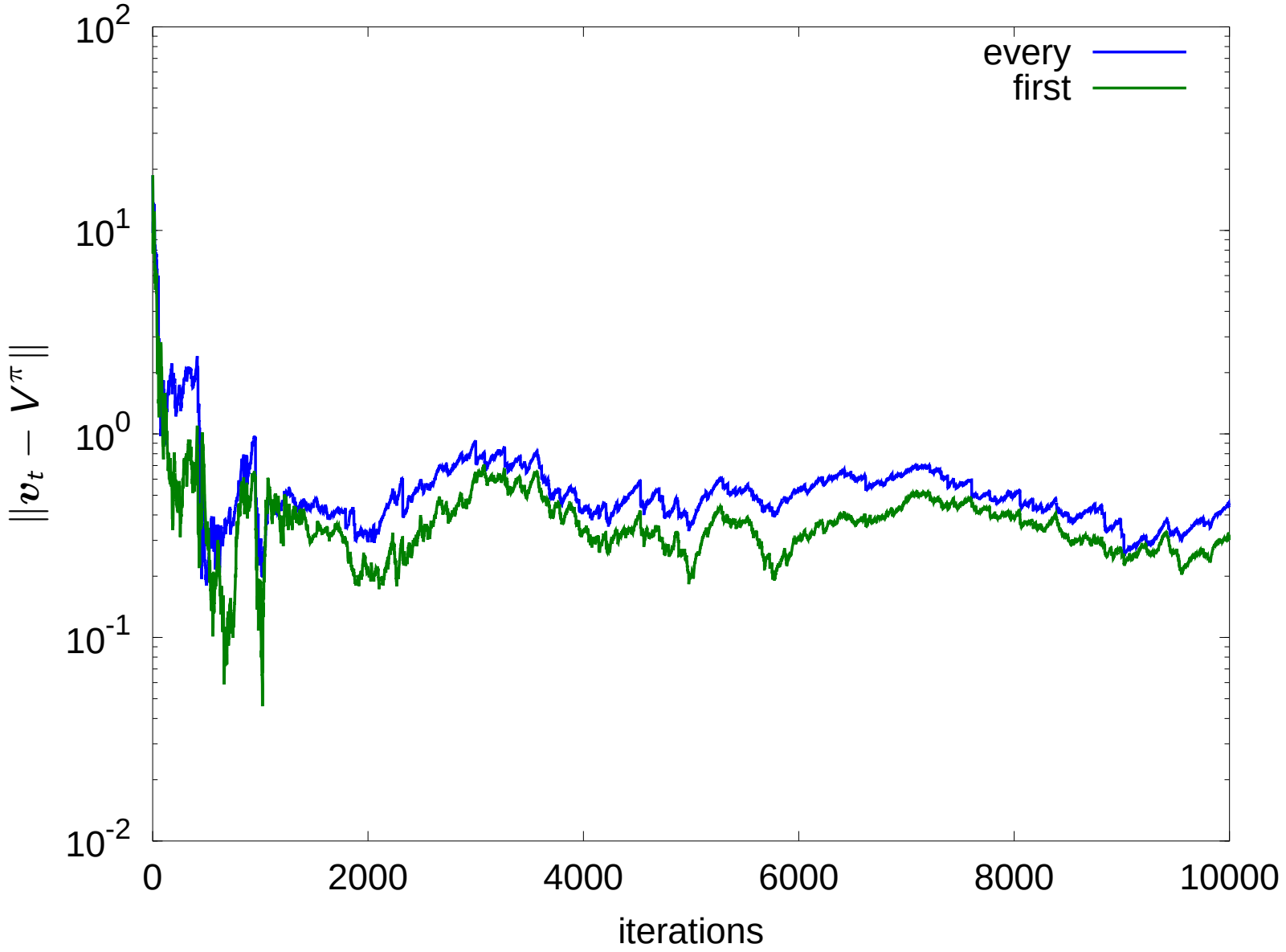

Figure: Error as the number of iterations n increases, for first and every visit Monte Carlo estimation.

Temporal differences

The full stochastic update is of the form:

$$v_{k+1}(s) = v_k(s) + \alpha(U_k - v_k(s)),$$

Using the temporal difference error $d(s_t, s_{t+1}) = v(s_t) - [r(s_t) + \gamma v(s_{t+1})]$,

$$v_{k+1}(s) = v_k(s) + \alpha \sum_t \gamma^t d_t, \qquad d_t \triangleq d(s_t, s_{t+1}) \tag{2.1}$$

Stochastic, incremental, update:

$$v_{t+1}(s) = v_t(s) + \alpha \gamma^t d_t. \tag{2.2}$$

TD(λ)

Temporal-difference operator

$$v_{n+1}(i) = v_n i + \tau_n i, \qquad \tau_n(i) \triangleq \sum_{t=0}^{\infty} \mathbb{E}_{\pi_n, \mu} \left[(\gamma\lambda)^m d_n(s_t, s_{t+1}) \mid s_0 = i \right].$$

Stochastic update:

$$v_{n+1}(s_t) = v_n(s_t) + \alpha \sum_{k=t}^{\infty} (\gamma\lambda)^{k-t} d_k. \tag{2.3}$$

Algorithm 7 Online TD(λ)

input Initial parameters $\boldsymbol{v}_k$, trajectories (s_t, a_t, r_t)
$\boldsymbol{e}_0 = \mathbf{0}$.
for $t = 1, \ldots, T$ **do**
  $d_t \triangleq d(s_t, s_{t+1})$ `temporal difference`
  $\boldsymbol{e}_t(s_t) = \boldsymbol{e}_{t-1}(s_t) + 1$ `eligibility increase`
  for $s \in \mathcal{S}$ **do**
    $\boldsymbol{v}_{t+1}(s_t) = \boldsymbol{v}_t(s) + \alpha_t \boldsymbol{e}_t(s) d_t$. `update all eligible states`
  end for
  $\boldsymbol{e}_{t+1} = \lambda \boldsymbol{e}_t$
end for
return $\boldsymbol{v}_T$

Algorithm 8 Simulation-based value iteration

1: Input μ, $\mathcal{S}$.
2: Initialise $s_t \in \mathcal{S}, \boldsymbol{v}_0 \in \mathcal{V}$.
3: **for** $t = 1, 2, \ldots$ **do**
4: $\quad s = s_t$.
5: $\quad \pi_t(s) = \arg\max_a \mathbb{P}_\mu(s'|s, a)\boldsymbol{v}_{t-1}(s')$
6: $\quad \boldsymbol{v}_t(s) = r(s) + \sum_{s'\mathcal{S}} \mathbb{P}_\mu(s'|s, \pi_t(s))\boldsymbol{v}_{t-1}(s')$
7: $\quad s_{t+1} \sim (1-\epsilon) \cdot \mathbb{P}(s_{t+1} \mid s_t = a, \pi_t, \mu) + \epsilon \cdot \mathcal{U}nif(\mathcal{S})$.
8: **end for**
9: Return π_n, V_n.

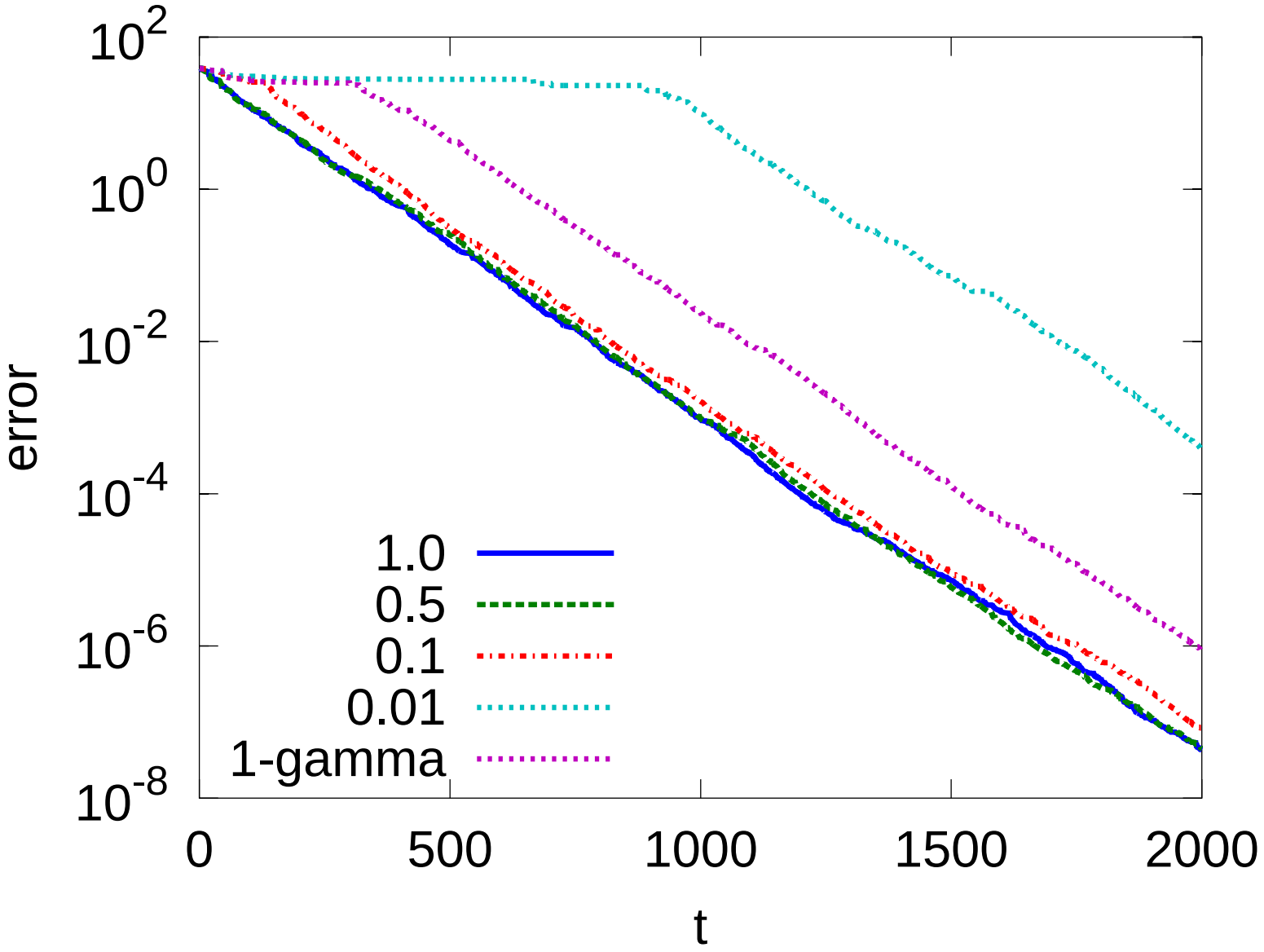

Figure: Simulation-based value iteration with $v_0 = 0$, varying $\epsilon_t = 0.1$.

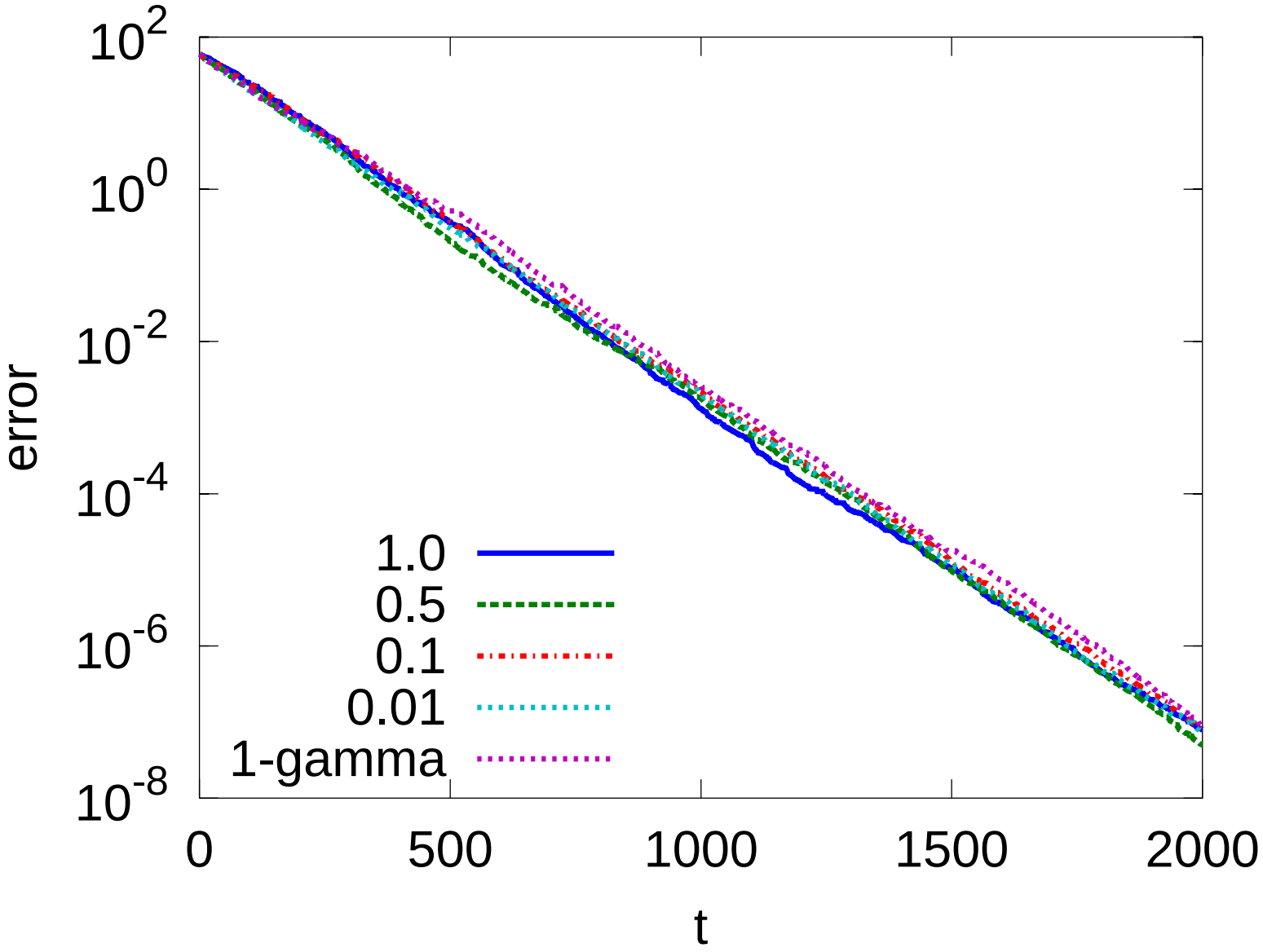

Figure: Simulation-based value iteration with $v_0 = 20 = 1/(1-\gamma)$, varying ϵ.

Algorithm 9 Q-learning

Input μ, $\mathcal{S}, \epsilon_t, \alpha_t$.
Initialise $s_t \in \mathcal{S}, \boldsymbol{q}_0 \in \mathcal{V}$.
for $t = 1, 2, \ldots$ **do**
    $s = s_t$.
    $a_t \sim \hat{\pi}^*_{\epsilon_t}(a \mid s_t, \boldsymbol{q}_t)$
    $s_{t+1} \sim \mathbb{P}(s_{t+1} \mid s_t = a, \pi_t, \mu)$.
    $\boldsymbol{q}_{t+1}(s_t, a_t) = \boldsymbol{q}_t(s_t, a_t) - \alpha_t[r(s) + \boldsymbol{v}_t(s_{t+1})]$, where $\boldsymbol{v}_t(s) = \max_{a \in \mathcal{A}} \boldsymbol{q}_t(s, a)$.
end for
Return π_n, V_n.

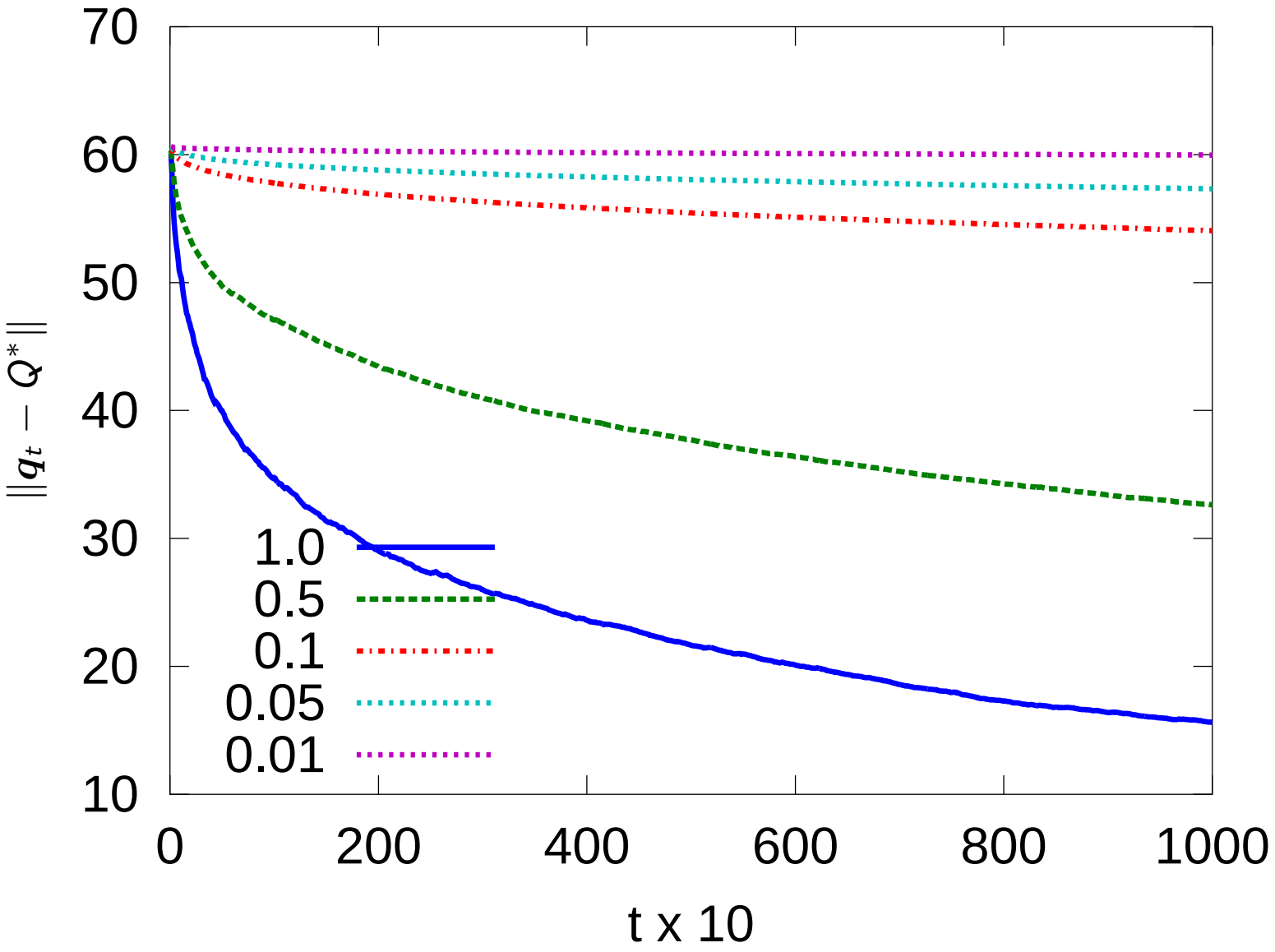

Figure: Q-learning estimation error with $v_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.

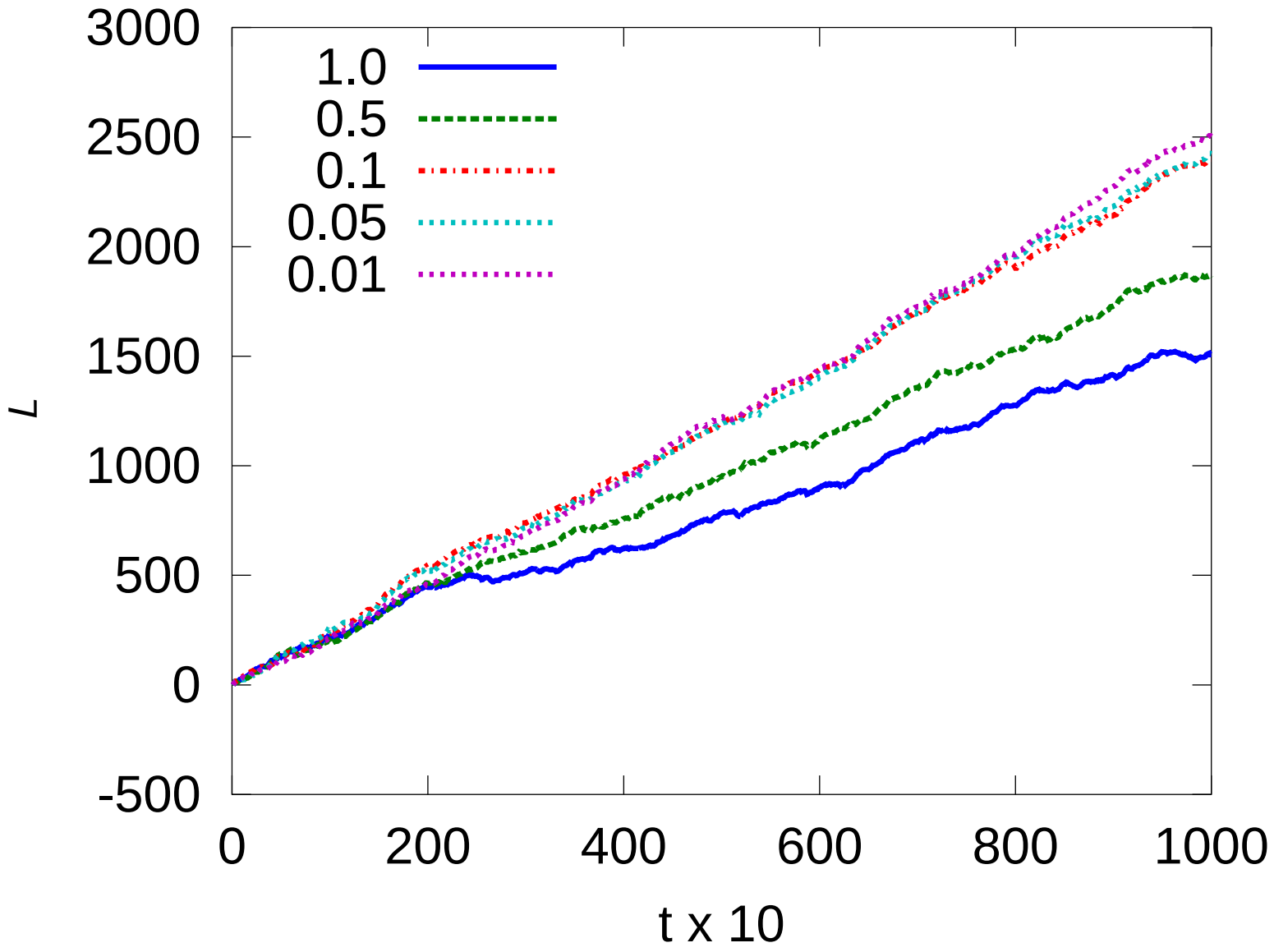

Figure: Q-learning estimation error with $v_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.

Algorithm 10 Generalised stochastic value iteration

Input $\hat{\mu}_0$, $\mathcal{S}, \epsilon_t, \alpha_t$.
Initialise $s_1 \in \mathcal{S}, \boldsymbol{q}_1 \in \mathcal{V}$.
for $t = 1, 2, \ldots$ **do**
  $a_t \sim \hat{\pi}^*_{\epsilon_t}(a \mid s_t, \boldsymbol{q}_t)$
  Observe s_{t+1}, r_{t+1}.
  $\hat{\mu}_t = \hat{\mu}_{t-1} \mid s_t, a_t, s_{t+1}, r_{t+1}$. `// update MDP estimate.`
  for $s \in \mathcal{S}$, $a \in \mathcal{A}$ **do**
    With probability $\sigma_t(s, a)$ do:

$$\boldsymbol{q}_{t+1}(s, a) = \boldsymbol{q}_t(s, a) - \alpha_t \left[r(s) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s_{t+1} = s' \mid s_t = s, a_t = a, \hat{\mu}_t) \boldsymbol{v}_t(s') \right]$$

    otherwise $\boldsymbol{q}_{t+1}(s, a) = \boldsymbol{q}_t(s, a)$.
  end for
end for
Return π_n, V_n.

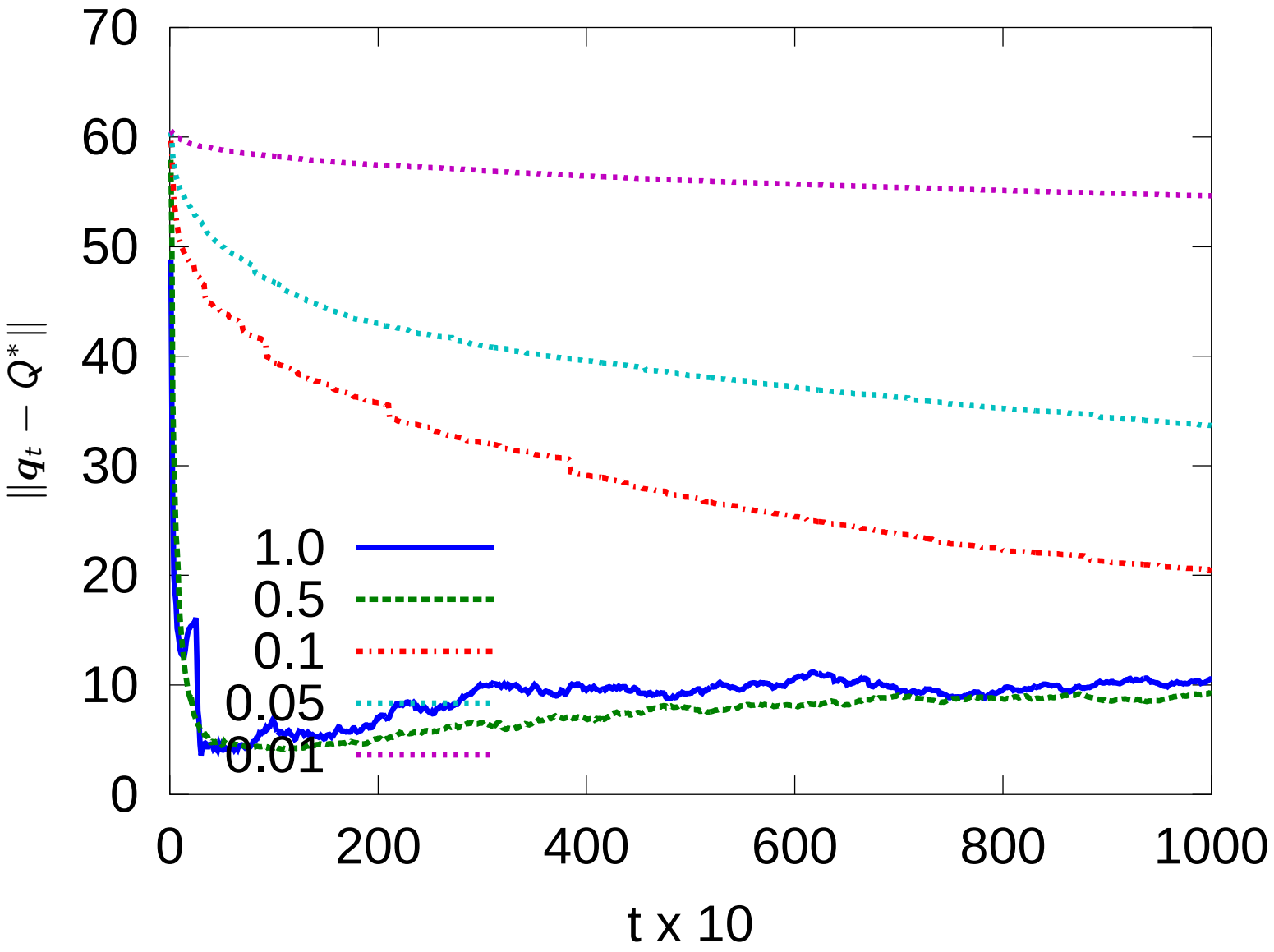

Figure: GSVI with Dirichlet model and single state-action update, with $v_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.

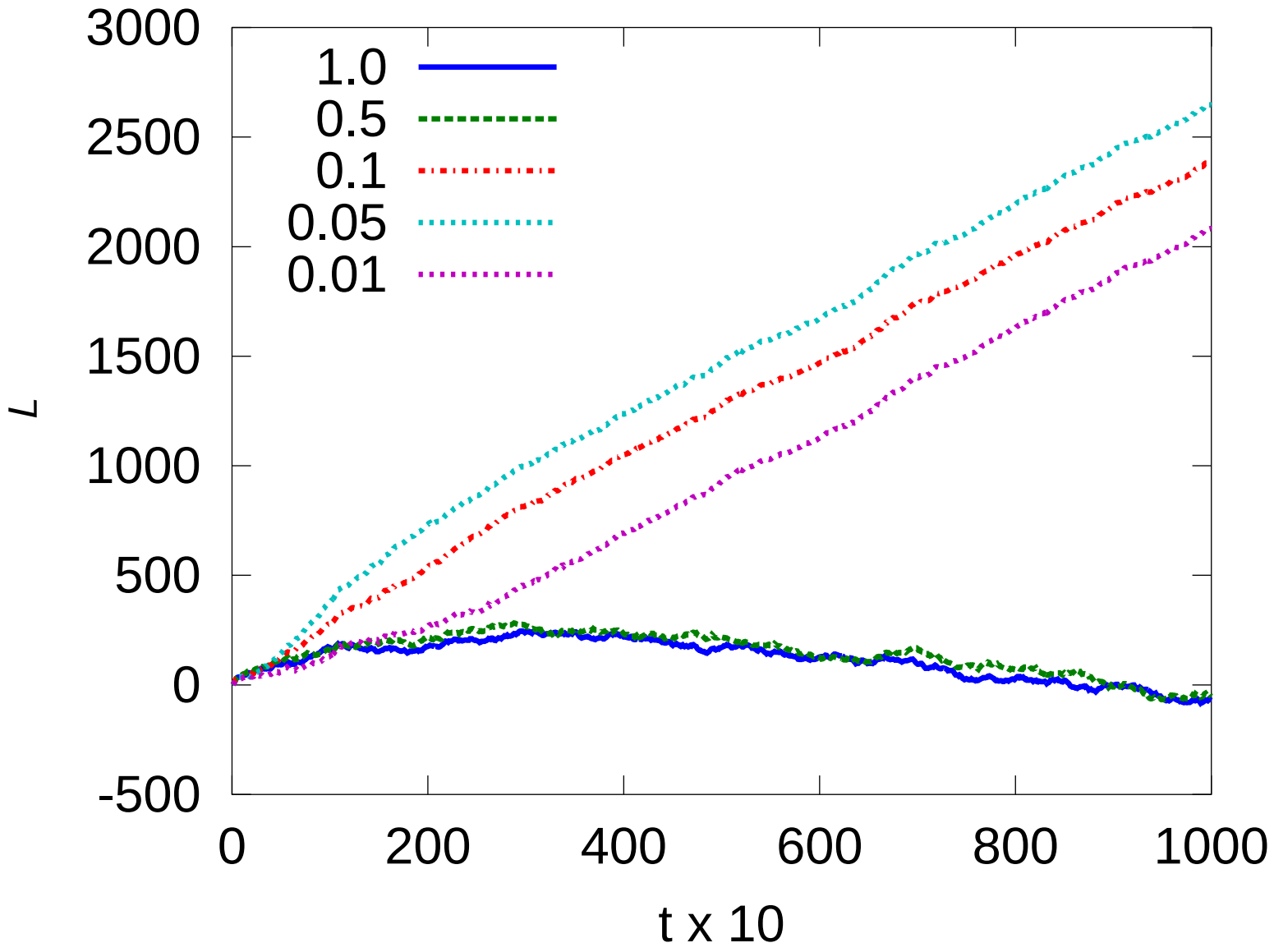

Figure: Q-learning with Dirichlet model and single state-action update with $v_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.

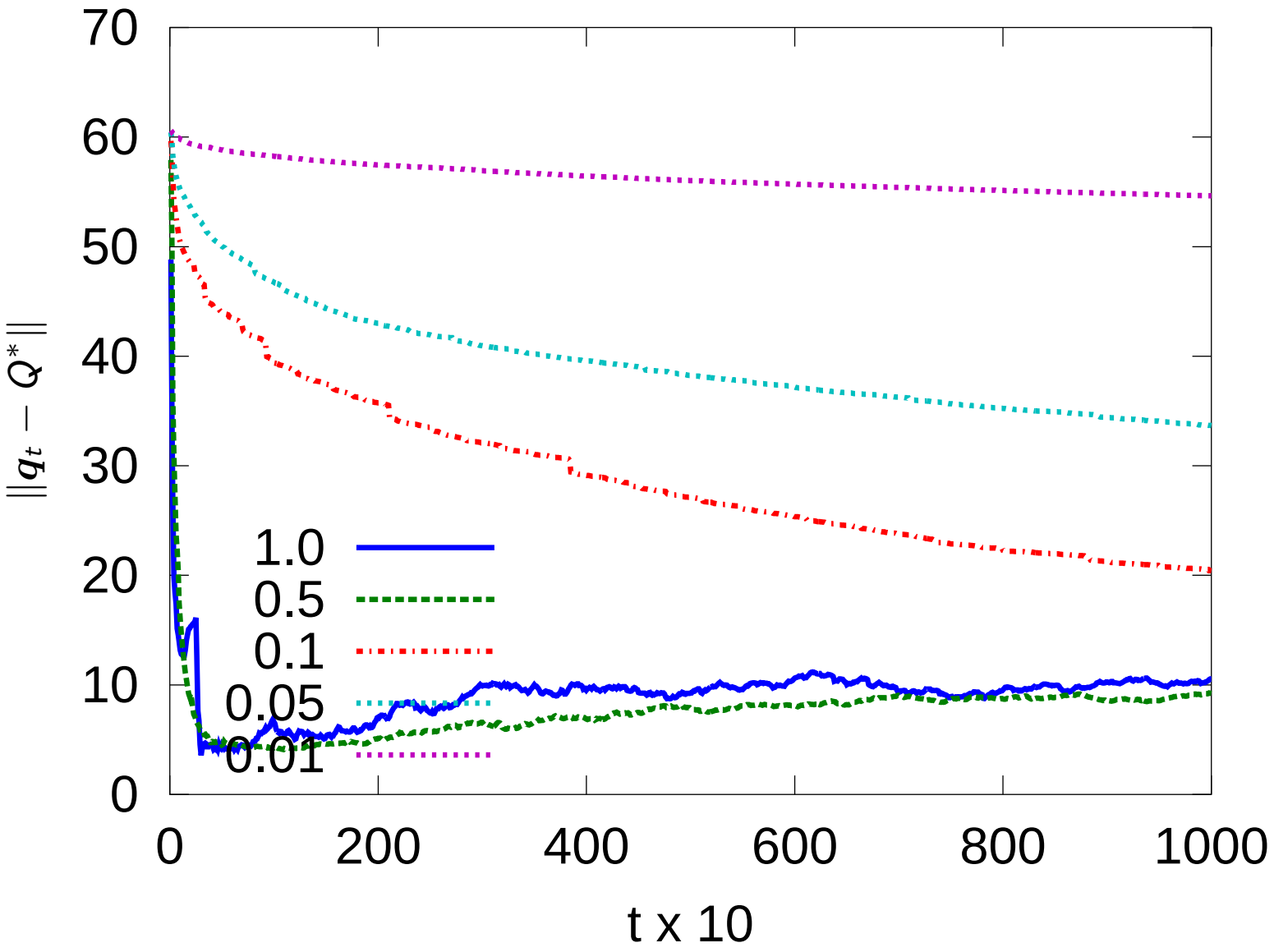

Figure: GSVI with Dirichlet model estimation and a uniform sweep over the state-space with $v_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.

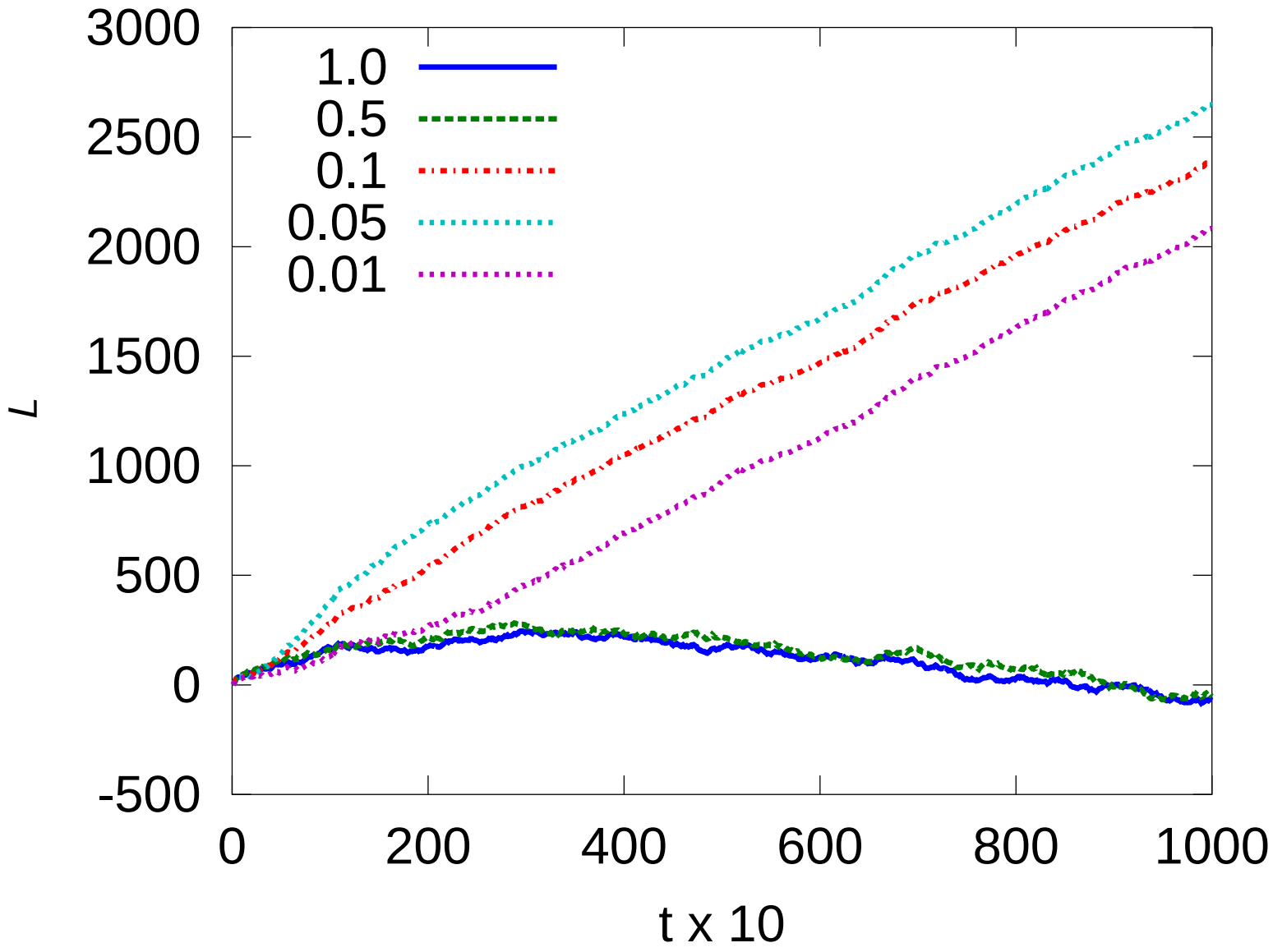

Figure: GSVI with Dirichlet estimation, and a uniform sweep over the state space, with $v_0 = 1/(1-\gamma)$, $\epsilon_t = 1/n_{s_t}$, $\alpha_t \in \alpha n_{s_t}^{-2/3}$.