Chapter 10:

On-policy Control with Approximation

(Semi-)gradient methods carry over to control in the usual on-policy GPI way (episodic cases)

- Always learn the action-value function of the current policy
- Always act near-greedily wrt the current action-value estimates
- The learning rule is the same as in Chapter 9:

$$\boldsymbol{w}_{t+1} \doteq \boldsymbol{w}_t + \alpha \left[U_t - \hat{q}\left(S_t, A_t, \boldsymbol{w}_t\right) \right] \nabla \hat{q}\left(S_t, A_t, \boldsymbol{w}_t\right)$$

update target e.g. $U_t = G_t$ (MC) $\quad U_t = R_{t+1} + \gamma \hat{q}\left(S_{t+1}, A_{t+1}, \boldsymbol{w}_t\right)$ (Sarsa)

$$U_t = R_{t+1} + \gamma \sum_a \pi\left(a|S_{t+1}\right) \hat{q}\left(S_{t+1}, A_{t+1}, \boldsymbol{w}_t\right) \text{ (Expected Sarsa)}$$

$$U_t = \sum_{s',r} p\left(s', r|S_t, A_t\right) \left[r + \gamma \sum_{a'} \pi\left(a'|s'\right) \hat{q}\left(s', a', \boldsymbol{w}_t\right) \right] \text{ (DP)}$$

(Semi-)gradient methods carry over to control

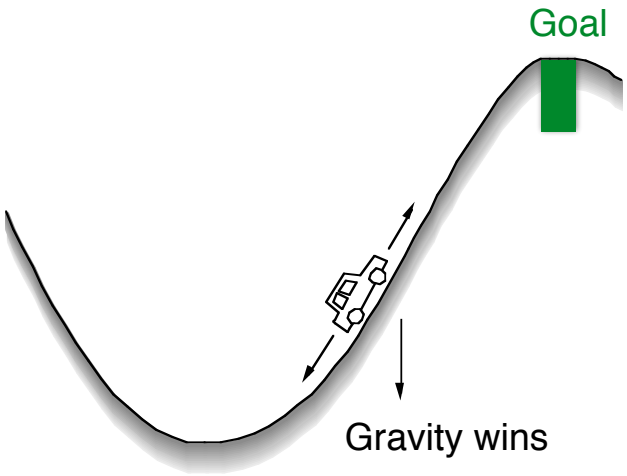
$$\boldsymbol{w}_{t+1} \doteq \boldsymbol{w}_t + \alpha \left[U_t - \hat{q}\left(S_t, A_t, \boldsymbol{w}_t\right)\right] \nabla \hat{q}\left(S_t, A_t, \boldsymbol{w}_t\right)$$

Episodic Semi-gradient Sarsa for Estimating $\hat{q} \approx q_*$

Input: a differentiable function $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$

Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Repeat (for each episode):
    $S, A \leftarrow$ initial state and action of episode (e.g., ε-greedy)
    Repeat (for each step of episode):
        Take action A, observe R, S'
        If S' is terminal:
            $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R - \hat{q}(S, A, \mathbf{w})\big]\nabla\hat{q}(S, A, \mathbf{w})$
            Go to next episode
        Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g., ε-greedy)
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R + \gamma\hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})\big]\nabla\hat{q}(S, A, \mathbf{w})$
        $S \leftarrow S'$
        $A \leftarrow A'$

Example: Mountain-Car task



STATES:
car's position & velocity

ACTIONS (+1,-1,0):
forward, reverse, none

REWARDS:
–1 until goal

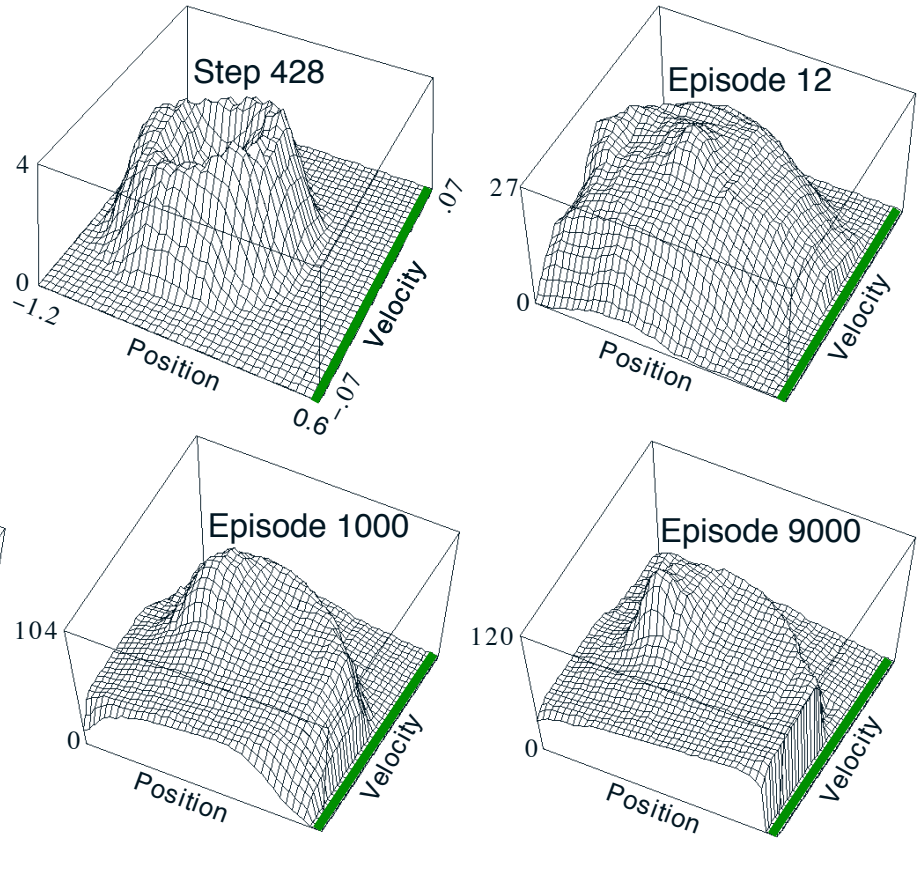
Minimum-Time-to-Goal Problem

Episodic, No Discounting

$$x_{t+1} \doteq bound\left[x_t + \dot{x}_{t+1}\right]$$

$$\dot{x}_{t+1} \doteq bound\left[\dot{x}_t + 0.001 A_t - 0.0025\cos(3x_t)\right]$$

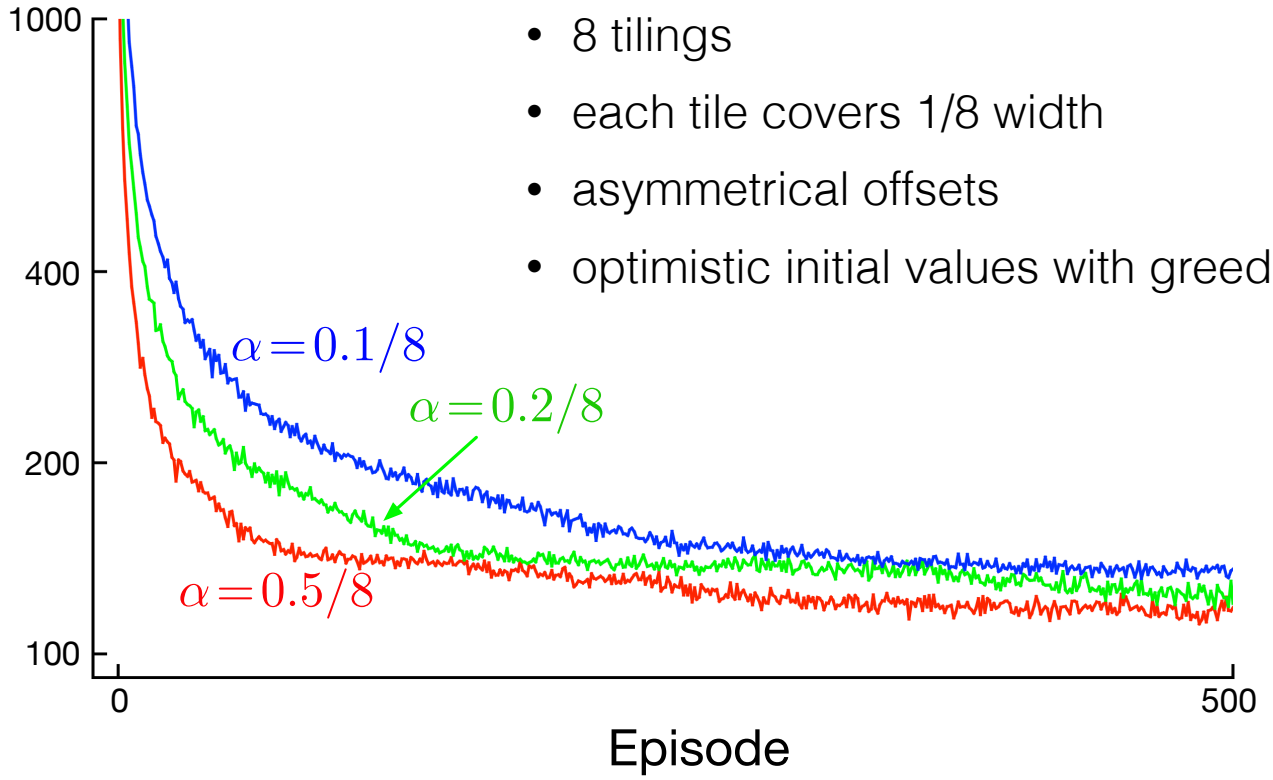
$-\max_a \hat{q}\left(s, a, \boldsymbol{w}\right)$



Learning curves
for semi-gradient Sarsa with tile coding

Mountain Car

Steps per episode
log scale
averaged over 100 runs

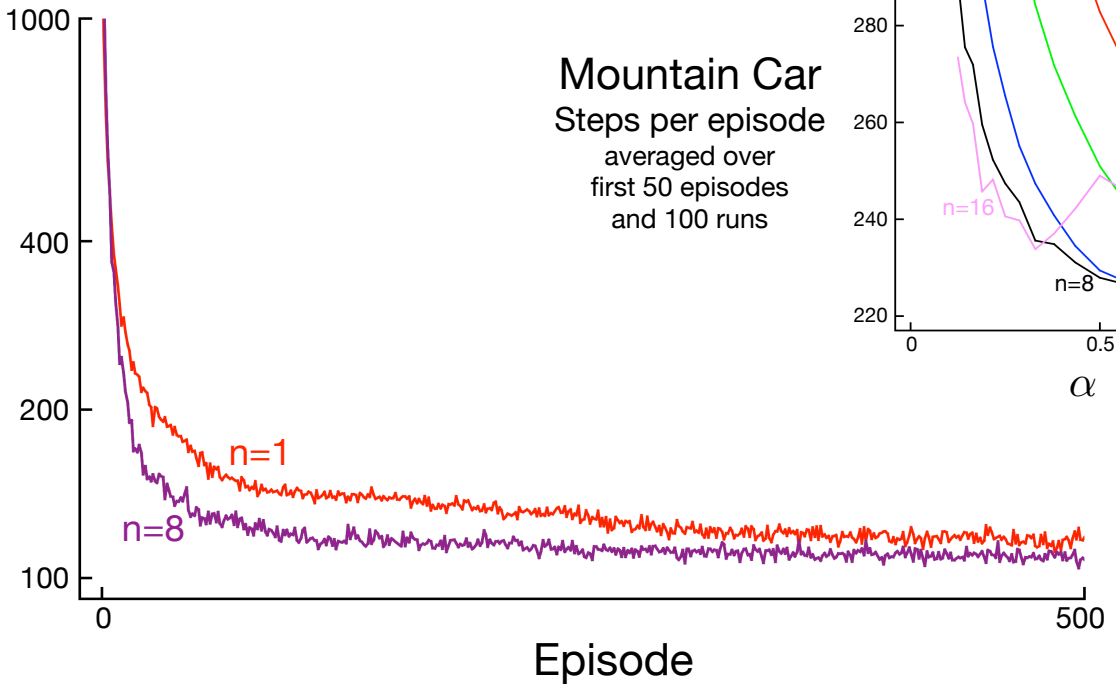


- 8 tilings
- each tile covers 1/8 width
- asymmetrical offsets
- optimistic initial values with greedy action

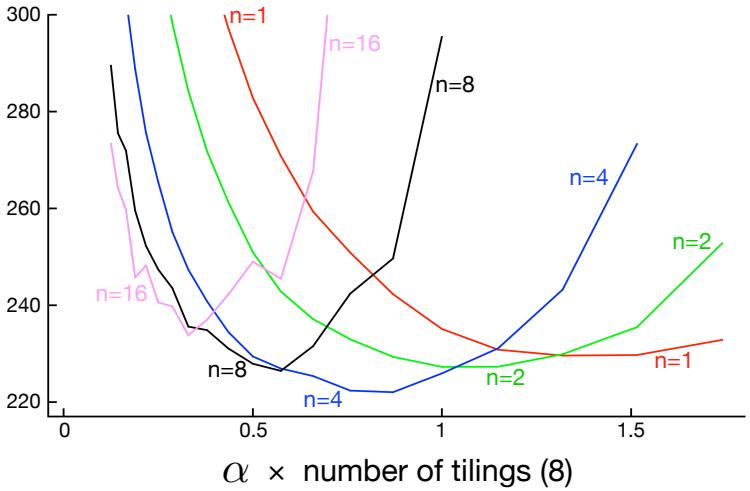
n-step semi-gradient Sarsa is better for *n* >1

$$\boldsymbol{w}_{t+n} \doteq \boldsymbol{w}_{t+n-1} + \alpha \left[G_{t:t+n} - \hat{q}\left(S_t, A_t, \boldsymbol{w}_{t+n-1}\right) \right] \nabla \hat{q}\left(S_t, A_t, \boldsymbol{w}_{t+n-1}\right), \quad 0 \leq t < T$$

Mountain Car
Steps per episode
log scale
averaged over 100 runs



Mountain Car
Steps per episode
averaged over
first 50 episodes
and 100 runs



A new goal for continuing tasks: Maximizing average reward per time step

Continuing problems: (discounting is problematic with fn approximation: no states clearly associated to the discounted return due to approximation)

The interaction between agent and environment goes on and on forever without termination or start states, no discounting.

Average reward: new maximization objective

$$
\begin{aligned}
r(\pi) &\doteq \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[R_t \mid A_{0:t-1} \sim \pi] \\
&= \lim_{t\to\infty} \mathbb{E}[R_t \mid A_{0:t-1} \sim \pi], \\
&= \sum_s \mu_\pi(s) \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) r
\end{aligned}
$$

assuming that these limits exist
is known as the *ergodicity* property

on-policy distribution

Redefine everything in the average reward setting

Differential return: $G_t \doteq R_{t+1} - r(\pi) + R_{t+2} - r(\pi) + R_{t+3} - r(\pi) + \cdots$

Differential value functions (stablizing things)

Bellman Eqs:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{r,s'} p(s', r|s, a) \left[r - r(\pi) + v_\pi(s') \right]$$

$$q_\pi(s, a) = \sum_{r,s'} p(s', r|s, a) \left[r - r(\pi) + \sum_{a'} \pi(a'|s') \, q_\pi(s', a') \right]$$

Differential TD errors:

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \hat{v}(S_{t+1}, \boldsymbol{w}_t) - \hat{v}(S_t, \boldsymbol{w}_t) \text{ or } \delta_t \doteq R_{t+1} - \bar{R}_t + \hat{q}(S_{t+1}, A_{t+1}, \boldsymbol{w}_t) - \hat{q}(S_t, A_t, \boldsymbol{w}_t)$$

estimate of $r(\pi)$

Differential semi-gradient Sarsa for estimating $\hat{q} \approx q_*$

Input: a differentiable function $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$
Parameters: step sizes $\alpha, \beta > 0$

Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Initialize average reward estimate $\bar{R}$ arbitrarily (e.g., $\bar{R} = 0$)
Initialize state S, and action A

Repeat (for each step):

Take action A, observe R, S'
Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g., ε-greedy)
$\delta \leftarrow R - \bar{R} + \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})$
$\bar{R} \leftarrow \bar{R} + \beta\delta$
$\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$
$S \leftarrow S'$
$A \leftarrow A'$

Discounting is futile in continuing control settings

- If we average the discounted return over a sufficiently large time interval, it turns out that the average of the discounted returns is always $r(\pi) / (1 - \gamma)$, proportional to the average reward.
- In particular, the *ordering* of all policies in the average discounted return setting would be exactly the same as in the average-reward setting.

The discount rate γ thus has no effect on the problem formulation!

Conclusions

- Control is straightforward in the on-policy, episodic, linear case
- For the continuing case, we need the average-reward setting
  - which is a lot like just replacing R_t with $R_t - r(\pi)$ everywhere
  - where $r(\pi)$ is the average reward per step, or its estimate
  - discounting has no effect on the ranking of policies