Exercise 10.1 We have not explicitly considered or given pseudocode for any Monte Carlo methods in this chapter. What would they be like? Why is it reasonable not to give pseudocode for them? How would they perform on the Mountain Car task? □

```
Monte Carlo method is just an extreme variation but still fits within the general
bounds of TD methods. One can achieve a Monte Carlo method by setting the number of
steps n equal to episode length T in the pseudocode for n-step SARSA in section 10.2

On top of that computational cost of Monte Carlo methods is quite high in practical
settings. Monte Carlo methods won't start to learn before completing the first
episode. Ending an episode can be tricky because the initial policy has random weights
which gives agent the natural tendency to repeat the same randomly initialized mistake
over nad over again. SARSA and other TD methods can learn during progress, i.e.
online, and achieve convergence faster.
```

Exercise 10.2 Give pseudocode for semi-gradient one-step *Expected* Sarsa for control. □

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

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$
Algorithm parameters: step size $\alpha > 0$, small $\varepsilon > 0$
Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

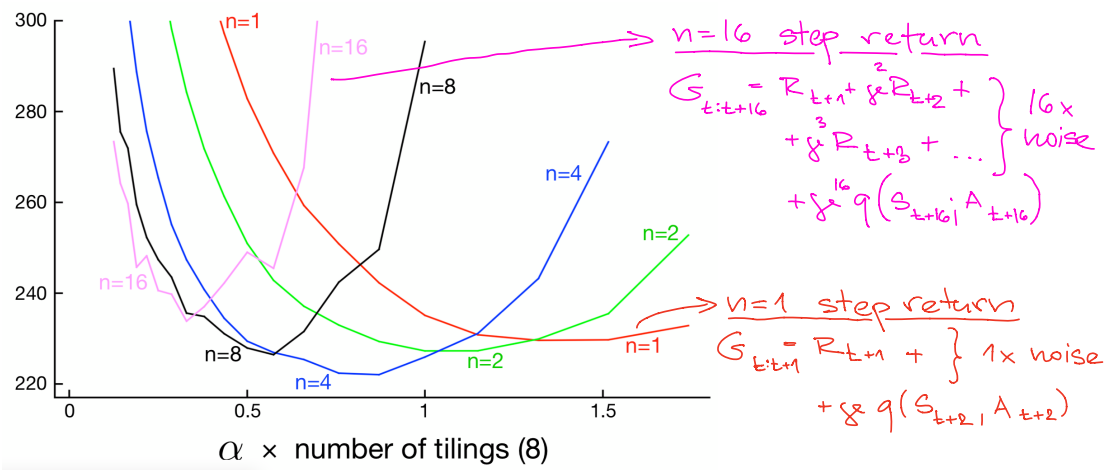
Loop for each episode:
  $S, A \leftarrow$ initial state and action of episode (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    If S' is terminal:
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R - \hat{q}(S, A, \mathbf{w})\big]\nabla\hat{q}(S, A, \mathbf{w})$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

$\sum_a \pi(a|S')\,\hat{q}(S', a, \mathbf{w})$ ... Expected value function

Exercise 10.3 Why do the results shown in Figure 10.4 have higher standard errors at large n than at small n? □

```
Higher values of n tend to value longer histories of rewards and therefore amplify
the noise present in the reward. Especially during earlier episodes. Strange action
selections during early episode have a larger effect on many more states.

Smaller values of n look only to the next state and even with a single unfortunate
action selection it has a relatively limited effect on the variation of the learned
action-value function.
```



Exercise 10.4 Give pseudocode for a differential version of semi-gradient Q-learning. □

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

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$
Algorithm 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 $\bar{R} \in \mathbb{R}$ arbitrarily (e.g., $\bar{R} = 0$)

Initialize state S, ~~and action A~~
Loop for each step:
~~Take action A, observe R, S'~~ Select action A as argmax $\hat{q}(S, \cdot, w)$
~~Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g., ε-greedy)~~ Use A and observe R, S
$\delta \leftarrow R - \bar{R} + ~~\hat{q}(S', A', \mathbf{w})~~ - \hat{q}(S, A, \mathbf{w})$
$\bar{R} \leftarrow \bar{R} + \beta\delta$ ← max $\hat{q}(S, \cdot, w)$
$\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$
$S \leftarrow S'$
$A \leftarrow A'$

Exercise 10.5 What equations are needed (beyond 10.10) to specify the differential version of TD(0)? □

I'm not really sure whether I understand the question and if this is the right answer…

$$w_{t+1} = w_t + \alpha \delta_t \nabla \hat{v}(s_t, w_t)$$

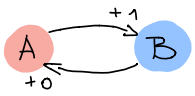
Exercise 10.6 Suppose there is an MDP that under any policy produces the deterministic sequence of rewards $+1, 0, +1, 0, +1, 0, \ldots$ going on forever. Technically, this is not allowed because it violates ergodicity; there is no stationary limiting distribution μ_π and the limit (10.7) does not exist. Nevertheless, the average reward (10.6) is well defined; What is it? Now consider two states in this MDP. From A, the reward sequence is exactly as described above, starting with a $+1$, whereas, from B, the reward sequence starts with a 0 and then continues with $+1, 0, +1, 0, \ldots$. The differential return (10.9) is not well defined for this case as the limit does not exist. To repair this, one could alternately define the value of a state as

$$v_\pi(s) \doteq \lim_{\gamma \to 1} \lim_{h \to \infty} \sum_{t=0}^{h} \gamma^t \Big(\mathbb{E}_\pi[R_{t+1} | S_0 = s] - r(\pi) \Big). \tag{10.13}$$

Under this definition, what are the values of states A and B? □

$$v_\pi(A) = \lim_{\gamma \to 1} \lim_{h \to \infty} \sum_{t=0}^{h} \gamma^t \frac{-1^t}{2} = \frac{1}{2} \lim_{\gamma \to 1} \underbrace{\lim_{h \to \infty} \sum_{t=0}^{h} -\gamma^t}_{\text{sum of infinite geometric series}} = \frac{1}{2} \lim_{\gamma \to 1} \underbrace{\lim_{h \to +\infty} \frac{1 - (-\gamma)^h}{1 + \gamma}}_{1/(1+\gamma)} = \frac{1}{4}$$

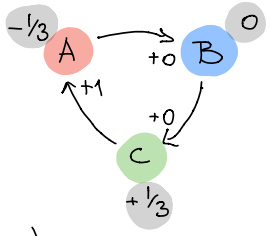
$$v_\pi(B) = -v_\pi(A) = -\frac{1}{4}$$



Exercise 10.7 Consider a Markov reward process consisting of a ring of three states A, B, and C, with state transitions going deterministically around the ring. A reward of $+1$ is received upon arrival in A and otherwise the reward is 0. What are the differential values of the three states using (10.13)? □

Average reward of the policy

$$r(\pi) = \lim_{t \to \infty} \mathbb{E}\left[R_t \,\middle|\, S_0, A_{0:t-1} \sim \pi \right] = \frac{1}{3}$$



$$v_\pi(A) = \lim_{\gamma \to 1} \lim_{h \to \infty} \sum_{t=0}^{h} \gamma^t \left(\mathbb{E}_\pi\left[R_{t+1} | S_0 = A \right] - r(\pi) \right) =$$

$$= \lim_{\gamma \to 1} \lim_{h \to \infty} \left(\sum_{t=0}^{h} (-1/3) \gamma^{3t} + \gamma \sum_{t=0}^{h} (-1/3) \gamma^{3t} + \gamma^2 \sum_{t=0}^{h} 2/3 \, \gamma^{3t} \right) =$$

$$= \lim_{\gamma \to 1} \left(-1/3 - 1/3 \, \gamma + 2/3 \gamma^2 \right) \frac{1}{1 - \gamma^3} = \lim_{\gamma \to 1} -\frac{2\gamma + 1}{3(\gamma^2 + \gamma + 1)} = -\frac{1}{3}$$

$$v_\pi(B) = \lim_{\gamma \to 1} \lim_{h \to \infty} \left(\sum_{t=0}^{h} (-1/3) \gamma^{3t} + \gamma \sum_{t=0}^{h} 2/3 \, \gamma^{3t} + \gamma^2 \sum_{t=0}^{h} (-1/3) \gamma^{3t} \right) = \lim_{\gamma \to 1} \frac{1 - \gamma}{3(\gamma^2 + \gamma + 1)} = 0$$

$$v_\pi(C) = \lim_{\gamma \to 1} \lim_{h \to \infty} \left(\sum_{t=0}^{h} 2/3 \, \gamma^{3t} + \gamma \sum_{t=0}^{h} (-1/3) \gamma^{3t} + \gamma^2 \sum_{t=0}^{h} (-1/3) \gamma^{3t} \right) = \lim_{\gamma \to 1} -\frac{\gamma + 2}{3(\gamma^2 + \gamma + 1)} = +\frac{1}{3}$$

Exercise 10.8 The pseudocode in the box on page 251 updates $\bar{R}_t$ using δ_t as an error rather than simply $R_{t+1} - \bar{R}_t$. Both errors work, but using δ_t is better. To see why, consider the ring MRP of three states from Exercise 10.7. The estimate of the average reward should tend towards its true value of $\frac{1}{3}$. Suppose it was already there and was held stuck there. What would the sequence of $R_{t+1} - \bar{R}_t$ errors be? What would the sequence of δ_t errors be (using (10.10))? Which error sequence would produce a more stable estimate of the average reward if the estimate were allowed to change in response to the errors? Why? □

The sequence of differential rewards would be (starting from A):

$$R_t - \bar{R}_t = \{ -\tfrac{1}{3}, \tfrac{2}{3}, -\tfrac{1}{3}, \dots \}$$

The sequence of δ_t is (also starting from A):

$$\delta_t = \{ -\tfrac{1}{3}+\tfrac{1}{3}-0, -\tfrac{1}{3}+\tfrac{2}{3}-\tfrac{1}{3}, \tfrac{2}{3}+0-\tfrac{2}{3}, \dots \} = \{ 0, 0, 0, \dots \}$$

This sequence would result into no updates at all. The learned value function has already converged to accurate estimates.

The δ_t update gives more stable estimates because it will automatically decrease the speed of oscillation when learning is close to convergence. On the other hand, during the early stages of learning when the estimated values are not close to the target, learning happens faster. The target is larger by the temporal difference - $\hat{v}(S_{t+1}) - \hat{v}(S_t)$.

Exercise 10.9 In the differential semi-gradient n-step Sarsa algorithm, the step-size parameter on the average reward, β, needs to be quite small so that $\bar{R}$ becomes a good long-term estimate of the average reward. Unfortunately, $\bar{R}$ will then be biased by its initial value for many steps, which may make learning inefficient. Alternatively, one could use a sample average of the observed rewards for $\bar{R}$. That would initially adapt rapidly but in the long run would also adapt slowly. As the policy slowly changed, $\bar{R}$ would also change; the potential for such long-term nonstationarity makes sample-average methods ill-suited. In fact, the step-size parameter on the average reward is a perfect place to use the unbiased constant-step-size trick from Exercise 2.7. Describe the specific changes needed to the boxed algorithm for differential semi-gradient n-step Sarsa to use this trick. □

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

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$
Algorithm 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 $\bar{R} \in \mathbb{R}$ arbitrarily (e.g., $\bar{R} = 0$)
Initialize state S, and action A
Loop 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)
  $\delta \leftarrow R - \bar{R} + \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})$
  $\bar{R} \leftarrow \bar{R} + \beta\delta$
  $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$
  $S \leftarrow S'$
  $A \leftarrow A'$