8 Multi-step Bootstrapping

Melih Kandemir

Özyeğin University
Computer Science Department
melih.kandemir@ozyegin.edu.tr

14 Nov 2017

Bridging the gap between TD(0) and MC

In one-step TD, the error term determines

- how often the action can be changed,
- as well as time interval over which bootstrapping is done.

Multi-step TD suggests

- updating the action first to incorporate the experience into the model immediately,
- but doing bootstrapping less often so that a recognizeable state change can take place.

One-step TD prediction

- Using $n-$step backups is TD because it changes an earlier estimate based on its difference from a later estimate.
- The Monte Carlo (MC) backup estimates v_π with the complete return

$$G_t \triangleq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T$$

- One-step TD uses $V_t(S_{t+1}) \approx v_\pi$ as a proxy for the rewards after time step $t+1$ in return

$$G_t^{(1)} \triangleq R_{t+1} + \gamma \underbrace{R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T}_{V_t(S_{t+1})}$$

leading to the TD(0) target introduced earlier

$$G_t^{(1)} \triangleq R_{t+1} + \gamma V_t(S_{t+1}).$$

Two-step TD prediction

Similarly, for two steps

$$G_t^{(2)} \triangleq R_{t+1} + \gamma R_{t+2} + \gamma^2 V_{t+1}(S_{t+2}),$$

where $V_{t+1}(S_{t+2})$ replaces

$$R_{t+3} + \gamma R_{t+4} + \cdots + \gamma^{T-t-3} R_T.$$

$n-$step TD prediction

- Generalizing to n steps,

$$G_t^{(n)} \triangleq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V_{t+n-1}(S_{t+n}),$$
$$n \geq 1, 0 \leq t < T - n.$$

- Approximate the full return by truncating after n steps and correcting the missing terms by $V_{t+n-1}(S_{t+n})$.
- Update by

$$V_{t+n}(S_t) \leftarrow V_{t+n-1}(S_t) + \alpha \left[G_t^{(n)} - V_{t+n-1}(S_t) \right], \;\; 0 \leq t < T.$$

while other states remain unchanged

$$V_{t+n}(s) \leftarrow V_{t+n-1}(s), \; \forall s \neq S_t.$$

The error reduction property

$$\max_s \left| \mathbb{E}_\pi \left[G_t^{(n)} \middle| S_t = s \right] - v_\pi(s) \right| \leq \gamma^n \max_s \left| V_{t+n-1}(s) - v_\pi(s) \right|$$

for all $n \geq 1$. Hence, the $n-$step target reduces worst-case estimation error.

$n-$step TD for estimating $V \approx v_\pi$

Initialize $V(s)$

All store/access ops for S_t and R_t can take their index mod n

repeat (for each episode)

  Init and store $S_0 \neq$ terminal

  $T \leftarrow \infty$

  for $t = 0, 1, 2, \cdots$

    if $t < T$, **then**

      Take action according to $\pi(\cdot|S_t)$

      Observe and store R_{t+1} and S_{t+1}

      If S_{t+1} is terminal, then $T \leftarrow t + 1$

    $\tau \leftarrow t - n + 1$

    if $\tau \geq 0$, **then**

      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$

      if $\tau + n < T$, **then** $G \rightarrow G + \gamma^n V(S_{\tau+n})$

      $V(S_\tau) \leftarrow V(S_\tau) + \alpha[G - V(S_\tau)]$

  until $\tau = T - 1$

Random Walk with 19 states

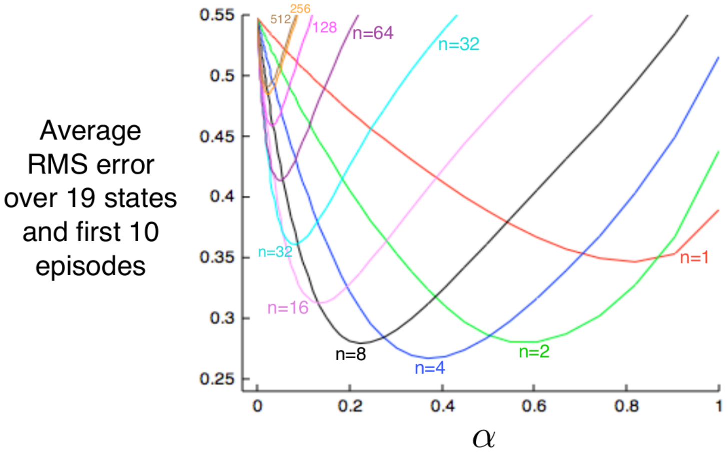

Figure: R. Sutton, A. Barto, MIT Press, 2017

The spectrum between TD(0) and MC

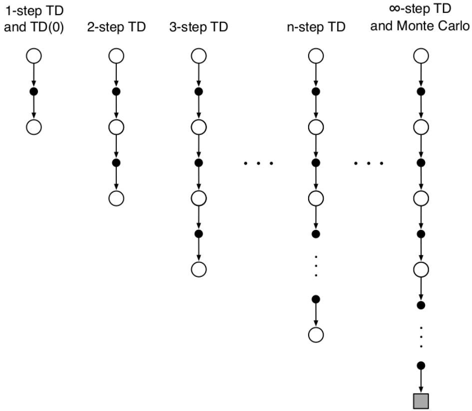

Figure: R. Sutton, A. Barto, MIT Press, 2017

White Circle: State, **Black Dot:** Action

$n-$step Sarsa

Redefine the $n-$step return in terms of estimated action values

$$G_t^{(n)} \triangleq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n Q_{t+n-1}(S_{t+n}, A_{t+n}),$$
$$n \geq 1, 0 \leq t < T - n$$

with $G_t^{(n)} = G_t$ if $t + n \geq T$.

The update reads

$$Q_{t+n}(S_t, A_t) \leftarrow Q_{t+n-1}(S_t, A_t) + \alpha \Big[G_t^{(n)} - Q_{t+n-1}(S_t, A_t) \Big],$$
$$0 \leq t < T$$

while the values of all other states remain unchanged

$$Q_{t+n}(s, a) = Q_{t+n-1}(s, a),$$

$\forall s, a$ s.t. $s \neq S_t$ or $a \neq A_t$.

$n-$step Expected Sarsa

$$G_t^{(n)} \triangleq \underbrace{R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n}}_{observe}$$
$$+ \underbrace{\gamma^n \sum_a \pi(a|S_{t+n}) Q_{t+n-1}(S_{t+n}, A_{t+n})}_{estimate},$$
$$n \geq 1, 0 \leq t < T - n.$$

The spectrum between Sarsa(0) and MC

Differently from above, Sarsa starts and ends with an action, not a state.

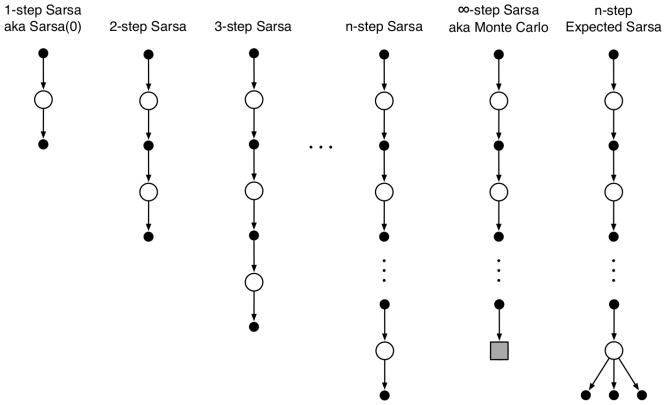

Figure: R. Sutton, A. Barto, MIT Press, 2017

GridWorld Example

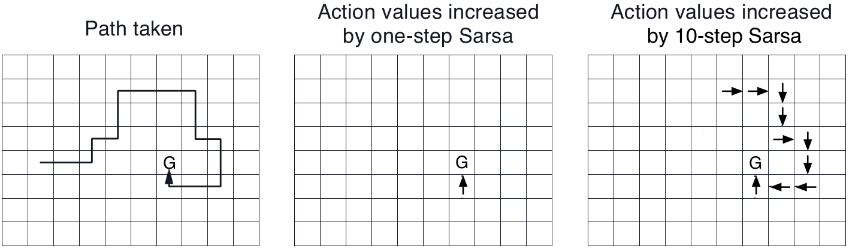

Figure: R. Sutton, A. Barto, MIT Press, 2017

$n-$step Sarsa for estimating $Q \approx q_*$

Initialize $Q(s,a)$, π to $\epsilon-$greedy wrt Q
All store/access ops for S_t, A_t, and R_t can take their index mod n
repeat (for each episode)
  Init and store $S_0 \neq$ terminal
  Select and store $A_0 \sim \pi(\cdot|S_0)$
  $T \leftarrow \infty$
  for $t = 0, 1, 2, \cdots$
    if $t < T$, **then**
      Take action A_t
      Observe and store R_{t+1} and S_{t+1}
      if S_{t+1} is terminal, **then** $T \leftarrow t+1$
      else Select and store $A_{t+1} \sim \pi(\cdot|S_{t+1})$
    $\tau \leftarrow t - n + 1$
    if $\tau \geq 0$, **then**
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
      if $\tau + n < T$, **then** $G \rightarrow G + \gamma^n Q(S_{\tau+n}, A_{\tau+n}))$
      $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha[G - Q(S_\tau, A_\tau)]$
      If π is being learned, ensure that $\pi(\cdot|S_\tau)$ is $\epsilon-$greedy wrt Q
  until $\tau = T - 1$

$n-$step off-policy learning by importance sampling

- The update for time step t is

$$V_{t+n}(S_t) \leftarrow V_{t+n-1}(S_t) + \alpha \rho_t^{t+n} \left[G_t^{(n)} - V_{t+n-1}(S_t) \right], \; 0 \leq t < T$$

  where

$$\rho_t^{t+n} \triangleq \prod_{k=t}^{\min(t+n-1, T-1)} \frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}$$

  is the **importance sampling ratio.**

- **Only the sampled timesteps are reweighted!**
- The update for the action-value function estimate is

$$Q_{t+n}(S_t, A_t) \leftarrow Q_{t+n-1}(S_t, A_t) + \alpha \rho_{t+1}^{t+n} \left[G_t^{(n)} - Q_{t+n-1}(S_t, A_t) \right],$$

$$0 \leq t < T.$$

  Note that ρ is one step ahead here, as the first action is pre-set.

Notes on $n-$step off-policy Sarsa

- Importance sampling **does** enable off-policy learning.
- This comes at the cost of increased update variance, necessitating a small learning rate.
- Small learning rate means slow learning.
- Off-policy methods are observed to train slower than on-policy methods overall.
- Seeking for solutions to this fundamental issue is an active research topic.

$n-$step Sarsa for estimating $Q \approx q_*$

input: a behavior policy μ s.t. $\mu(a|s) > 0,\ \forall s, a$ Initialize $Q(s,a)$, π to $\epsilon-$greedy wrt Q
All store/access ops for S_t, A_t, and R_t can take their index mod n
repeat (for each episode)
  Init and store $S_0 \neq$ terminal
  Select and store $A_0 \sim \pi(\cdot|S_0)$
  $T \leftarrow \infty$
  for $t = 0, 1, 2, \cdots$
    if $t < T$, **then**
      Take action A_t
      Observe and store R_{t+1} and S_{t+1}
      if S_{t+1} is terminal, **then** $T \leftarrow t + 1$
      else Select and store $A_{t+1} \sim \pi(\cdot|S_{t+1})$
    $\tau \leftarrow t - n + 1$
    if $\tau \geq 0$, **then**
      $\rho \leftarrow \prod_{i=\tau+1}^{\min(\tau+n-1, T-1)} \frac{\pi(A_i|S_i)}{\mu(A_i|S_i)}$
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n, T)} \gamma^{i-\tau-1} R_i$
      if $\tau + n < T$, **then** $G \rightarrow G + \gamma^n Q(S_{\tau+n}, A_{\tau+n}))$
      $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\rho[G - Q(S_\tau, A_\tau)]$
      If π is being learned, ensure that $\pi(\cdot|S_\tau)$ is $\epsilon-$greedy wrt Q
  until $\tau = T - 1$

The $n-$step Tree Backup Algorithm

- Off-policy learning without importance sampling is indeed possible!
- For state-to-action transitions, take a full backup (perform Expected Sarsa)!
- For action-to-state transitions, choose an arbitrary action and observe the next state.
- Weight the state-to-action transitions by π, also the entire tree below it.

The $n-$step Tree Backup Algorithm

Define the expected action value under π as

$$V_t \triangleq \sum_a \pi(a|S_t)Q_{t-1}(S_t, a).$$

Then redefine the TD error as

$$\delta_t \triangleq R_{t+1} + \gamma V_{t+1} - Q_{t-1}(S_t, A_t).$$

The trick here is that the target also contains the TD error!

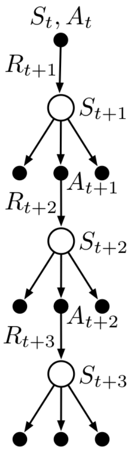

Figure: R. Sutton, A. Barto, MIT Press, 2017

The $n-$step Tree Backup Algorithm

Then the $n - step$ returns follow

$$
\begin{aligned}
G_t^{(1)} &\triangleq R_{t+1} + \gamma V_{t+1} \\
&= Q_{t-1}(S_t, A_t) + \delta_t,
\end{aligned}
$$

$$
\begin{aligned}
G_t^{(2)} &\triangleq R_{t+1} + \gamma V_{t+1} - \gamma\pi(A_{t+1}|S_{t+1})Q_t(S_{t+1}, A_{t+1}) \\
&\qquad\qquad + \gamma\pi(A_{t+1}|S_{t+1})[R_{t+2} + \gamma V_{t+2}] \\
&= R_{t+1} + \gamma V_{t+1} \\
&\qquad + \gamma\pi(A_{t+1}|S_{t+1})[R_{t+2} + \gamma V_{t+2} - Q_t(S_{t+1}, A_{t+1})] \\
&= R_{t+1} + \gamma V_{t+1} + \gamma\pi(A_{t+1}|S_{t+1})\delta_{t+1} \\
&= Q_{t-1}(S_t, A_t) + \delta_t + \gamma\pi(A_{t+1}|S_{t+1})\delta_{t+1},
\end{aligned}
$$

The $n-$step Tree Backup Algorithm

$$G_t^{(3)} \triangleq Q_{t-1}(S_t, A_t) + \delta_t + \gamma\pi(A_{t+1}|S_{t+1})\delta_{t+1} \\ + \gamma^2\pi(A_{t+1}|S_{t+1})\pi(A_{t+2}|S_{t+2})\delta_{t+2},$$

$$G_t^{(n)} \triangleq Q_{t-1}(S_t, A_t) + \sum_{k=t}^{\min(t+n-1, T-1)} \delta_k \pi_{i=t+1} \gamma\pi(A_i|S_i).$$

The $n-$step Tree Backup Algorithm

$$Q_{t+n}(S_t, A_t) \triangleq Q_{t+n-1}(S_t, A_t) + \alpha\Big[G_t^{(n)} - Q_{t+n-1}(S_t, A_t)\Big],$$
$$0 \le t < T$$

while the values of all other states remain unchanged

$$Q_{t+n}(s, a) = Q_{t+n-1}(s, a),$$

$\forall s, a$ s.t. $s \neq S_t, \;\; a \neq A_t$.

$n-$step Tree Backup Algorithm Pseudocode

Initialize $Q(s,a)$, π to $\epsilon-$greedy wrt Q
All store/access ops can take their index mod n
repeat (for each episode)
  Init and store $S_0 \neq$ terminal
  Select and store $A_0 \sim \pi(\cdot|S_0)$
  Store $Q(S_0, A_0)$ as Q_0
  $T \leftarrow \infty$
  for $t = 0, 1, 2, \cdots$
    if $t < T$, **then**
      Take action A_t
      Observe R, observe and store S_{t+1}
      if S_{t+1} is terminal, **then**
        $T \leftarrow t+1$
        Store $R - Q_t$ as δ_t
      else
        Store $R + \gamma \sum_a \pi(a|S_{t+1}) Q(S_{t+1}, a) - Q_t$ as δ_t
        Select and store an arbitrary action as A_{t+1}
        Store $Q(S_{t+1}, A_{t+1})$ as Q_{t+1} and $\pi(A_{t+1}|S_{t+1})$ as π_{t+1}
    $\tau \leftarrow t - n + 1$
    if $\tau \geq 0$, **then**
      $E \leftarrow 1$ and $G \leftarrow Q_\tau$
      for $k = \tau, \cdots, \min(\tau + n - 1, T - 1)$
        $G \leftarrow G + E\delta_k$ and $E \leftarrow \gamma E \pi_{k+1}$
      $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha[G - Q(S_\tau, A_\tau)]$
      If π is being learned, ensure that $\pi(a|S_\tau)$ is $\epsilon-$greedy wrt $Q(S_\tau, \cdot)$
  until $\tau = T - 1$

A unifying algorithm: $n-$step $Q(\sigma)$

- $n-$step Sarsa always samples.
- The tree-backup algo always takes full state-to-action transition backups.
- $n-$step Expected Sarsa samples until the final action of the episode, then terminates with a full backup.

Let us unify these approaches:

- Decide at every step whether to sample or take full backup!
- Sample the binary decision from $z \sim \mathrm{Bernoulli}(\sigma)$ for some $\sigma \in [0, 1]$.
- $\sigma = 1$: Sarsa, $\quad \sigma = 0$: Tree-backup!

A unifying algorithm: $n-$step $Q(\sigma)$

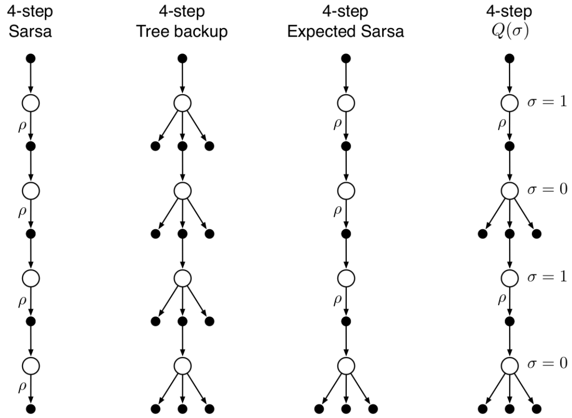

Figure: R. Sutton, A. Barto, MIT Press, 2017

A unifying algorithm: $n-$step $Q(\sigma)$

The $n-$step return of Sarsa should be extended as

$$G_t^{(n)} = Q_{t-1}(S_t, A_t) + \sum_{k=t}^{\min(t+n-1,T-1)} \gamma^{k-t}\Big[R_{k+1} + \gamma Q_k(S_{k+1}, A_{k+1}) - Q_{k-1}(S_k, A_k)\Big],$$

and the TD error should be generalized as

$$\delta_k = R_{t+1} + \gamma[\sigma_{t+1} Q_t(S_{t+1}, A_{t+1}) + (1-\sigma_{t+1})V_{t+1}] - Q_{t-1}(S_t, A_t).$$

Then the one-step return reads

$$\begin{aligned} G_t^{(1)} &\triangleq R_{t+1} + \gamma[\sigma_{t+1} Q_t(S_{t+1}, A_{t+1}) + (1-\sigma_{t+1})V_{t+1}] \\ &= \delta_t + Q_{t-1}(S_t, A_t). \end{aligned}$$

A unifying algorithm: $n-$step $Q(\sigma)$

The two-step return similarly is

$$
\begin{aligned}
G_t^{(2)} &\triangleq R_{t+1} + \gamma[\sigma_{t+1} Q_t(S_{t+1}, A_{t+1}) + (1-\sigma_{t+1})V_{t+1}] \\
&- \gamma(1-\sigma_{t+1})\pi(A_{t+1}|S_{t+1})Q_t(S_{t+1}, A_{t+1}) \\
&+ \gamma(1-\sigma_{t+1})\pi(A_{t+1}|S_{t+1})\Big[R_{t+2} + \gamma[\sigma_{t+2} Q_t(S_{t+2}, A_{t+2}) \\
&\qquad + (1-\sigma_{t+2})V_{t+2}]\Big] - \gamma\sigma_{t+1} Q_t(S_{t+1}, A_{t+1}) \\
&+ \gamma\sigma_{t+1}\Big[[R_{t+2} + \gamma[\sigma_{t+2} Q_t(S_{t+2}, A_{t+2}) + (1-\sigma_{t+2})V_{t+2}]\Big] \\
&= Q_{t-1}(S_t, A_t) + \delta_t + \gamma(1-\sigma_{t+1})\pi(A_{t+1}|S_{t+1})\delta_{t+1} + \gamma\sigma_{t+1}\delta_{t+1} \\
&= Q_{t-1}(S_t, A_t) + \delta_t + \gamma[(1-\sigma_{t+1})\pi(A_{t+1}|S_{t+1}) + \sigma_{t+1}]\delta_{t+1}.
\end{aligned}
$$

A unifying algorithm: $n-$step $Q(\sigma)$

More generally, the $n-$step return is

$$G_t^{(n)} \triangleq Q_{t-1}(S_t, A_t) + \sum_{k=t}^{\min(t+n-1,T-1)} \delta_k \prod_{i=t+1}^{k} \gamma[(1-\sigma_i)\pi(A_i|S_i) + \sigma_i].$$

To perform off-policy learning, we need to extend the sampling probability term by the importance sampling ratio applied to all steps except the last

$$\rho_t^{t+n} \triangleq \prod_{k=t}^{\min(t+n-1,T-1)} \left(1 - \sigma_k + \sigma_k \frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}\right).$$

$n-$step $Q(\sigma)$ for estimating $Q \approx q_*$

input: a behavior policy μ s.t. $\mu(a|s) > 0$ for all s, a.
Initialize $Q(s,a)$, π to $\epsilon-$greedy wrt Q, all store/access ops can take their index mod n
repeat (for each episode)
  Init/store $S_0 \neq$ terminal, select/store $A_0 \sim \mu(\cdot|S_0)$, store $Q(S_0, A_0)$ as Q_0, and $T \leftarrow \infty$
  for $t = 0, 1, 2, \cdots$
    if $t < T$, **then**
      Take action A_t
      Observe R, observe and store S_{t+1}
      if S_{t+1} is terminal, **then**
        $T \leftarrow t + 1$ and store $\delta_t \leftarrow R - Q_t$
      else
        Select and store $A_{t+1} \sim \mu(\cdot|S_{t+1})$ and σ_{t+1}
        $Q_{t+1} \leftarrow Q(S_{t+1}, A_{t+1})$
        $\delta_t \leftarrow R + \gamma\sigma_{t+1}Q_{t+1} + \gamma(1 - \sigma_{t+1}) \sum_a \pi(a|S_{t+1})Q(S_{t+1}, a) - Q_t$
        $\pi_{t+1} \leftarrow \pi(A_{t+1}|S_{t+1})$ and $\rho_{t+1} \leftarrow \frac{\pi(A_{t+1}|S_{t+1})}{\mu(A_{t+1}|S_{t+1})}$
    $\tau \leftarrow t - n + 1$
    if $\tau \geq 0$, **then**
      $\rho \leftarrow 1$ and $E \leftarrow 1$ and $G \leftarrow Q_\tau$
      for $k = \tau, \cdots, \min(\tau + n - 1, T - 1)$
        $G \leftarrow G + E\delta_k$ and $E \leftarrow \gamma E[(1 - \sigma_{k+1})\pi_{k+1} + \sigma_{k+1}]$
        $\rho \leftarrow \rho(1 - \sigma_k + \sigma_k\rho_k)$
      $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\rho[G - Q(S_\tau, A_\tau)]$
      If π is being learned, ensure that $\pi(a|S_\tau)$ is $\epsilon-$greedy wrt $Q(S_\tau, \cdot)$
  until $\tau = T - 1$

$n-$step $Q(\sigma)$ on 19-state random walk

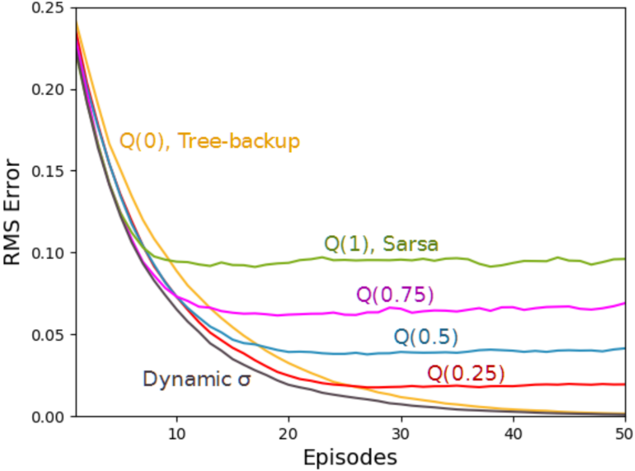

Figure: de Asis et al., ArXiv:1703.01327v1, 2017

$n-$step $Q(\sigma)$ on Mountain Car

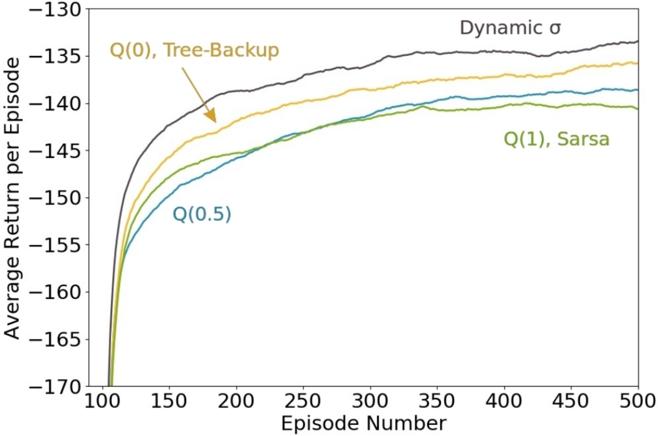

Figure: de Asis et al., ArXiv:1703.01327v1, 2017

$n-$step semi-gradient TD

The return is defined as

$$G_{t:t+n} \triangleq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{v}(S_{t+n}, \mathbf{w}_{t+n-1}),$$
$$0 \leq t \leq T - n,$$

and the parameter update as

$$\mathbf{w}_{t+n} \leftarrow \mathbf{w}_{t+n-1} + \alpha [G_{t:t+n} - \hat{v}(S_t, \mathbf{w}_{t+n-1})] \nabla \hat{v}(S_t, \mathbf{w}_{t+n-1}),$$
$$0 \leq t < T.$$

$n-$step semi-gradient TD for estimating $V \approx v_\pi$

Initialize $V(s)$
All store/access ops for S_t and R_t can take their index mod n
repeat (for each episode)
  Init and store $S_0 \neq$ terminal
  $T \leftarrow \infty$
  for $t = 0, 1, 2, \cdots$
    if $t < T$, **then**
      Take action according to $\pi(\cdot|S_t)$
      Observe and store R_{t+1} and S_{t+1}
      If S_{t+1} is terminal, then $T \leftarrow t + 1$
    $\tau \leftarrow t - n + 1$
    if $\tau \geq 0$, **then**
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
      if $\tau + n < T$, **then** $G \rightarrow G + \gamma^n \hat{v}(S_{\tau+n})$
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha[G - \hat{v}(S_t, \mathbf{w})]\nabla\hat{v}(S_t, \mathbf{w})$
  until $\tau = T - 1$

$n-$step semi-gradient Sarsa

The return is

$$G_{t:t+n} \triangleq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}),$$
$$n \geq 1, 0 \leq t < T - n,$$

and the update rule is

$$\mathbf{w}_{t+n} \triangleq \mathbf{w}_{t+n-1} + \alpha [G_{t:t+n} - \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1})] \nabla \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1}),$$
$$0 \leq t < T.$$

$n-$step semi-gradient Sarsa algorithm

input: A differentiable $\hat{q}$
Initialize $\mathbf{w}$
All store/access ops for S_t, A_t, and R_t can take their index mod n
repeat (for each episode)
  Init and store $S_0 \neq$ terminal
  Select and store $A_0 \sim \pi(\cdot|S_0)$, $\epsilon-$greedy wrt $\hat{q}(S_0, \cdot, \mathbf{w})$
  $T \leftarrow \infty$
  for $t = 0, 1, 2, \cdots$
    if $t < T$, **then**
      Take action A_t
      Observe and store R_{t+1} and S_{t+1}
      if S_{t+1} is terminal, **then** $T \leftarrow t + 1$
      else Select/store $A_{t+1} \sim \pi(\cdot|S_{t+1})$, $\epsilon-$greedy wrt $\hat{q}(S_0, \cdot, \mathbf{w})$
    $\tau \leftarrow t - n + 1$
    if $\tau \geq 0$, **then**
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
      if $\tau + n < T$, **then** $G \rightarrow G + \gamma^n \hat{q}(S_{\tau+n}, A_{\tau+n}, \mathbf{w})$
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha[G - \hat{q}(S_\tau, A_\tau, \mathbf{w})]\nabla\hat{q}(S_\tau, A_\tau, \mathbf{w})$
  until $\tau = T - 1$

$n-$step semi-gradient Sarsa on Mountain Car

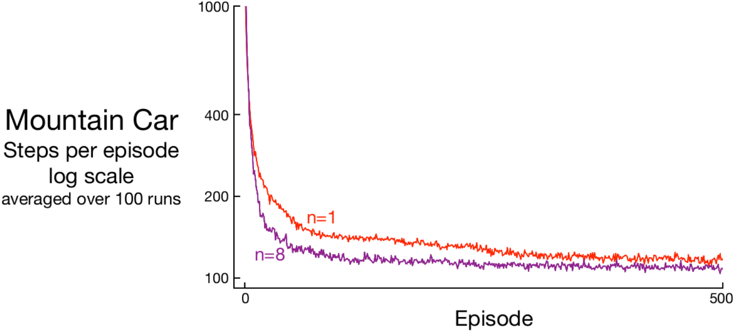

Figure: R. Sutton, A. Barto, MIT Press, 2017

$n-$step semi-gradient Sarsa on Mountain Car

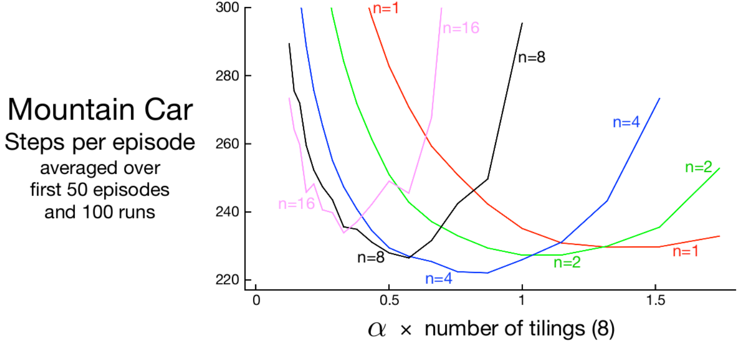

Figure: R. Sutton, A. Barto, MIT Press, 2017

$n-$step differential semi-gradient Sarsa

The return is defined as

$$G_{t:t+n} \triangleq R_{t+1} - \bar{R}_{t+1} + R_{t+2} - \bar{R}_{t+2} + \cdots \\ + R_{t+n} - \bar{R}_{t+n} + \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1})$$

where $\hat{R}$ is an estimate of $r(\pi)$, and $n \geq 1$.

The $n-$step TD error reads

$$\delta_t \triangleq G_{t:t+n} - \hat{q}(S_t, A_t, \mathbf{w}).$$

The $n-$step differential semi-gradient Sarsa algorithm

Input: a differentiable $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$ and a positive integer n
All store/access operations S_t, A_t, R_t can take their index mod n
Initialize $\mathbf{w} \in \mathbb{R}^d$, $\bar{R}$, S, A and $\bar{R} \in \mathbb{R}$
Initialize and store S_0 and A_0
for $t = 0, 1, 2, ...$
  Take action A_t, observe R_{t+1} and S_{t+1}
  $A_{t+1} \sim \pi(\cdot|S_{t+1})$ s.t. π is $\epsilon-$greedy wrt $\hat{q}(S_0, \cdot, \mathbf{w})$
  $\tau \leftarrow t - n + 1$
  if$\tau \geq 0$
    $\delta \leftarrow \sum_{i=\tau+1}^{\tau+n}(R_i - \hat{R}) + \hat{q}(S_{\tau+n}, A_{\tau+n}, \mathbf{w}) - \hat{q}(S_\tau, A_\tau, \mathbf{w})$
    $\hat{R} \leftarrow \hat{R} + \beta\delta$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S_\tau, A_\tau, \mathbf{w})$