Putting it all together

RL Superman: uses Monte-Carlo to estimate state-action values using ϵ-greedy exploration with decaying ϵ

- Why super?
  - Learns Q-values from observed returns (doesn't require a model)
  - Estimates become better over time with more experience
  - Can choose the best action as $\text{argmax}_a\, Q^\pi(s,a)$
  - Starts out with exploration ($\epsilon = 1$), but slowly becomes greedy ($\epsilon \approx 0$)
- But...
  - requires a lot of experience to get good estimates and policy
  - works for small finite MDPs only
  - applicable to episodic problems only
  - wastes time evaluating bad policies
- Solution: Temporal Difference Learning

Temporal Differencing (SuBa6)

"If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning." (SuBa6)

- Monte Carlo
  - update values after an episode is done (based on returns $R = \sum_t r_t$
- Temporal-difference learning
  - update values after each step (based on immediate reward r_t)

Temporal Differencing (SuBa6)

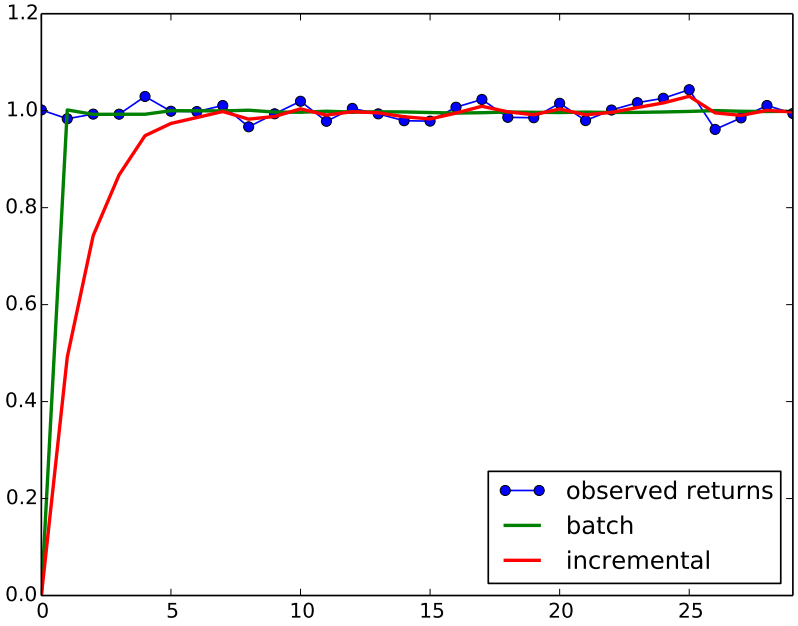
"If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning." (SuBa6)

- Monte Carlo
  - update values after an episode is done (based on returns $R = \sum_t r_t$
- Temporal-difference learning
  - update values after each step (based on immediate reward r_t)

- Explained in two steps
  1. Show incremental version of Monte Carlo (uses R)
  2. Show TD learning (uses r_t)

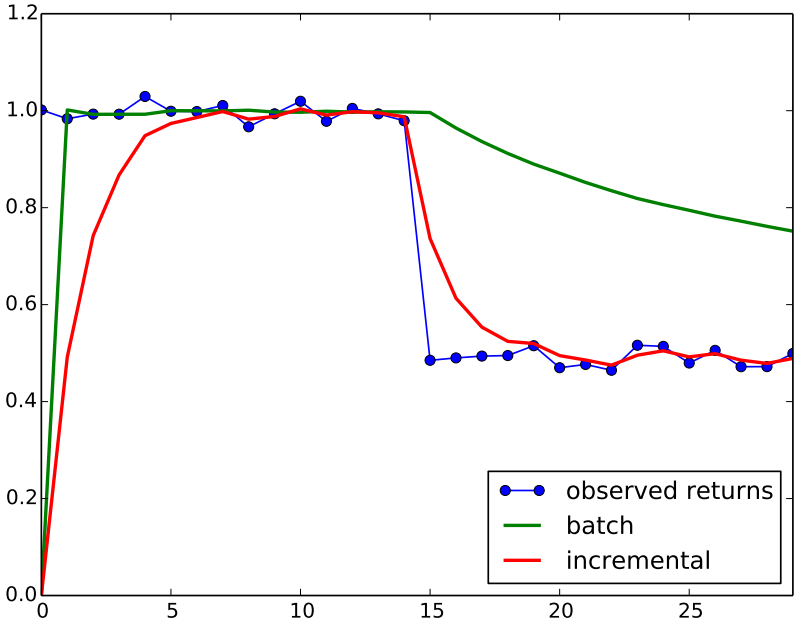
Monte Carlo: Batch vs. Incremental

- Batch: $V^{\pi}(s) = average(Returns(s)) = \frac{1}{N} \sum_{e=1}^{N} R(s)^e$
  - N is the number of episodes
- Incremental updates: $V^{\pi}(s) = V^{\pi}(s) + \alpha \left[R - V^{\pi}(s)\right]$
  - $0 < \alpha < 1$ is 'learning rate' ($\alpha = 1$ would be batch, $\alpha = 0$ would mean no learning)
  - α also known as 'step-size'



Monte Carlo: Batch vs. Incremental

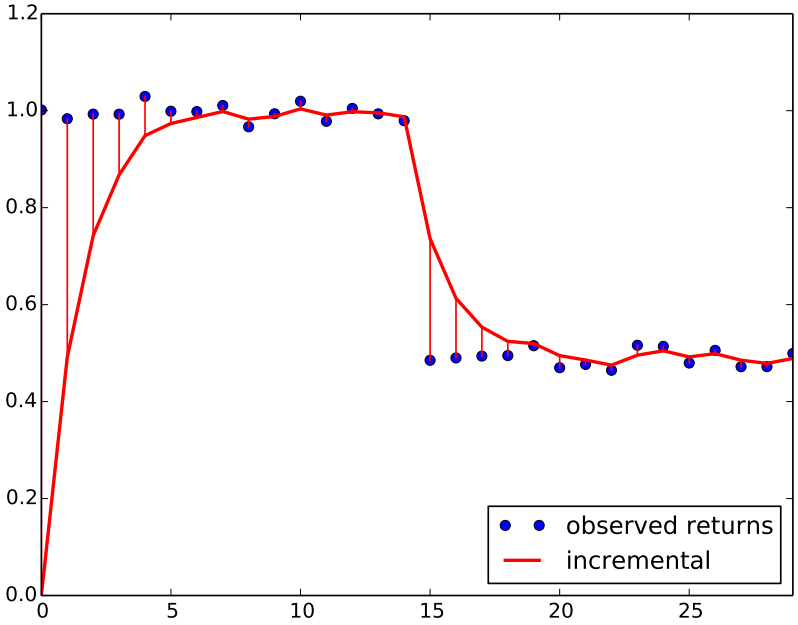
- Batch: $V^\pi(s) = average(Returns(s)) = \frac{1}{N} \sum_{e=1}^{N} R(s)^e$
  - N is the number of episodes
- Incremental updates: $V^\pi(s) = V^\pi(s) + \alpha\,[R - V^\pi(s)]$
  - $0 < \alpha < 1$ is 'learning rate' ($\alpha = 1$ would be batch, $\alpha = 0$ would mean no learning)
  - α also known as 'step-size'



- Incremental better if environment is non-stationary (changes over time)

Monte Carlo: Batch vs. Incremental

- Batch: $V^\pi(s) = average(Returns(s)) = \frac{1}{N} \sum_{e=1}^{N} R(s)^e$
  - N is the number of episodes
- Incremental updates: $V^\pi(s) = V^\pi(s) + \alpha\,[R - V^\pi(s)]$
  - $0 < \alpha < 1$ is 'learning rate' ($\alpha = 1$ would be batch, $\alpha = 0$ would mean no learning)
  - α also known as 'step-size'



From incremental MC to TD (SuBa6.1)

Definition

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\}$$

Incremental estimation from experience

$$V^{\pi}(s_t) = V^{\pi}(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

From incremental MC to TD (SuBa6.1)

Definition

$$V^\pi(s) = E_\pi\{R_t | s_t = s\} = E_\pi\{\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\}$$

Incremental estimation from experience

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

From incremental MC to TD (SuBa6.1)

Definition

$$
\begin{aligned}
V^{\pi}(s) &= E_{\pi}\{R_t | s_t = s\} \\
&= E_{\pi}\{\textstyle\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\} \\
&= E_{\pi}\{r_{t+1} + \gamma \textstyle\sum_{k=1}^{T} \gamma^{k-1} r_{t+k+1} | s_t = s\}
\end{aligned}
$$

Incremental estimation from experience

$$V^{\pi}(s_t) = V^{\pi}(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

From incremental MC to TD (SuBa6.1)

$$
\begin{aligned}
V^\pi(s) & \quad \text{Definition} \\
& = E_\pi\{R_t | s_t = s\} \\
& = E_\pi\{\textstyle\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\} \\
& = E_\pi\{r_{t+1} + \gamma \textstyle\sum_{k=1}^{T} \gamma^{k-1} r_{t+k+1} | s_t = s\} \\
& = E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\} \\
& \quad \text{Bellman equation}
\end{aligned}
$$

Incremental estimation from experience

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

From incremental MC to TD (SuBa6.1)

Definition

$$
\begin{aligned}
V^\pi(s) &= E_\pi\{R_t | s_t = s\} \\
&= E_\pi\{\textstyle\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\} \\
&= E_\pi\{r_{t+1} + \gamma \textstyle\sum_{k=1}^{T} \gamma^{k-1} r_{t+k+1} | s_t = s\} \\
&= E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\}
\end{aligned}
$$

Bellman equation

Incremental estimation from experience

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

From incremental MC to TD (SuBa6.1)

$$
\begin{aligned}
V^\pi(s) &= E_\pi\{R_t | s_t = s\} && \text{Definition}\\
&= E_\pi\{\textstyle\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\}\\
&= E_\pi\{r_{t+1} + \gamma \textstyle\sum_{k=1}^{T} \gamma^{k-1} r_{t+k+1} | s_t = s\}\\
&= E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\} && \text{Bellman equation}
\end{aligned}
$$

Incremental estimation from experience

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad \text{(TD)}$$

From incremental MC to TD (SuBa6.1)

Definition

$$
\begin{aligned}
V^\pi(s) &= E_\pi\{R_t | s_t = s\} \\
&= E_\pi\{\textstyle\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\} \\
&= E_\pi\{r_{t+1} + \gamma \textstyle\sum_{k=1}^{T} \gamma^{k-1} r_{t+k+1} | s_t = s\} \\
&= E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\}
\end{aligned}
$$

Bellman equation

Incremental estimation from experience

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad \text{(TD)}$$

Initialize $V(s)$ arbitrarily, π to the policy to be evaluated
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
    $a \leftarrow$ action given by π for s
    Take action a; observe reward, r, and next state, s'
    $V(s) \leftarrow V(s) + \alpha\big[r + \gamma V(s') - V(s)\big]$
    $s \leftarrow s'$
  until s is terminal

Figure : Policy evaluation with $TD(0)$.

From incremental MC to TD (SuBa6.1)

Definition

$$
\begin{aligned}
V^\pi(s) &= E_\pi\{R_t | s_t = s\} \\
&= E_\pi\{\textstyle\sum_{k=0}^{T} \gamma^k r_{t+k+1} | s_t = s\} \\
&= E_\pi\{r_{t+1} + \gamma \textstyle\sum_{k=1}^{T} \gamma^{k-1} r_{t+k+1} | s_t = s\} \\
&= E_\pi\{r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\}
\end{aligned}
$$

Bellman equation

Incremental estimation from experience

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[R_t - V(s_t)] \quad \text{(MC)}$$

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad \text{(TD)}$$

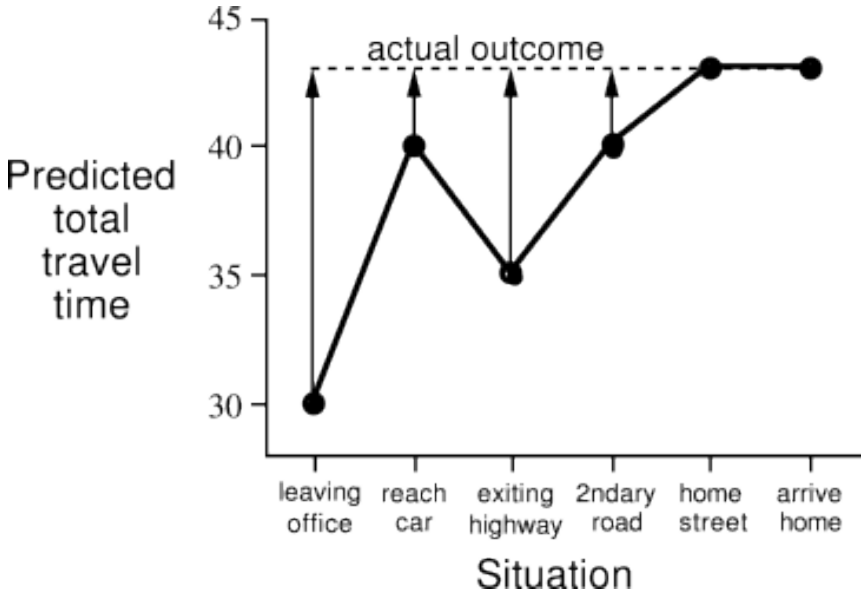


Figure : Driving home (MC)

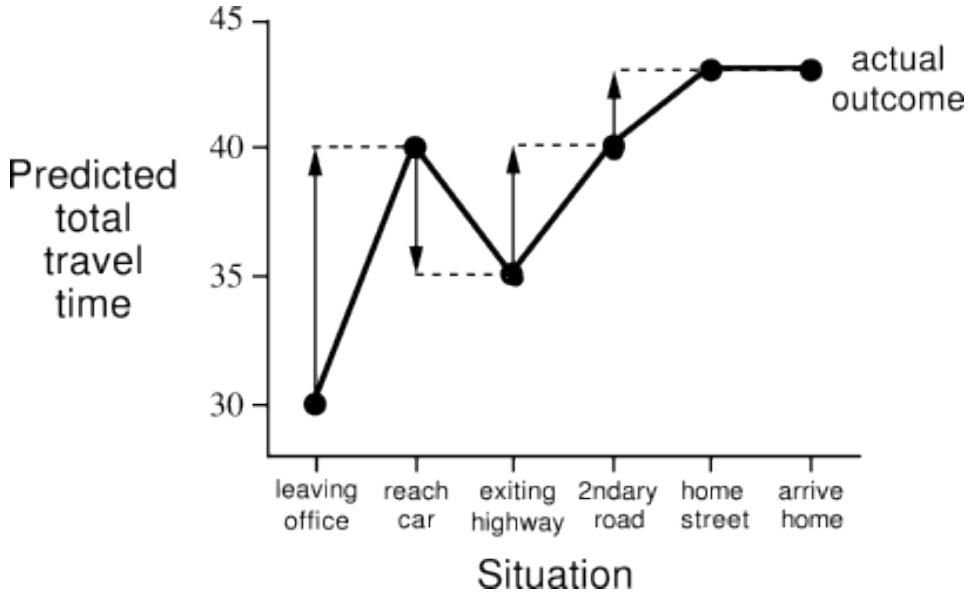


Figure : Driving home (TD)

TD learning for Q-Values (SARSA) (SuBA6.4)

$$V^{\pi}(s_t) = V^{\pi}(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \tag{1}$$

$$Q^{\pi}(s_t, a_t) = Q^{\pi}(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{2}$$

TD learning for Q-Values (SARSA) (SuBA6.4)

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad (1)$$

$$Q^\pi(s_t, a_t) = Q^\pi(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (2)$$

- Incremental update requires $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$
  - Known as "SARSA"
  - If s_{t+1} is terminal, then $Q(s_{t+1}, a_{t+1})$ is zero

TD learning for Q-Values (SARSA) (SuBa6.4)

$$V^{\pi}(s_t) = V^{\pi}(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad (1)$$

$$Q^{\pi}(s_t, a_t) = Q^{\pi}(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (2)$$

- Incremental update requires $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$
  - Known as "SARSA"
  - If s_{t+1} is terminal, then $Q(s_{t+1}, a_{t+1})$ is zero

Initialize $Q(s, a)$ arbitrarily
Repeat (for each episode):
  Initialize s
  Choose a from s using policy derived from Q (e.g., ε-greedy)
  Repeat (for each step of episode):
    Take action a, observe r, s'
    Choose a' from s' using policy derived from Q (e.g., ε-greedy)
    $Q(s, a) \leftarrow Q(s, a) + \alpha\left[r + \gamma Q(s', a') - Q(s, a)\right]$
    $s \leftarrow s'$; $a \leftarrow a'$;
  until s is terminal

Figure : SARSA: an on-policy TD control algorithm