Reinforcement Learning & Intelligent Agents

Lecture 6: Temporal-Difference & n-step returns

S. M. Ahsan Kazmi

Recap

Last lecture:

- Model-free prediction to estimate values in an unknown MDP
  - Monte-Carlo Learning
  - Temporal-Difference Learning (TD 0)

This lecture:

- Model-free prediction to estimate values in an unknown MDP
  - Temporal-Difference Learning

Recap: MC Summary

MC has several **advantages** over DP:

- Can learn directly from interaction with the environment
- No need for full models
- No need to learn about ALL states (no bootstrapping)
- Less harmed by violating Markov property
- MC methods provide an alternate policy evaluation process

Recap: Advantages and Disadvantages of MC vs. TD

TD can learn before knowing the final outcome

- TD can learn online after every step
- MC must wait until the end of the episode before a return is known

TD can learn without the final outcome

- TD can learn from incomplete sequences
- MC can only learn from complete sequences
- TD works in continuing (non-terminating) environments
- MC only works for episodic (terminating) environments

Both MC and TD converge (under certain assumptions)

Recap: Batch MC and TD

- Batch Updating: train completely on a finite amount of data
    - e.g., train repeatedly on 10 episodes until convergence.
- Compute updates according to TD or MC, but only update estimates after each complete pass through the data.
- For any finite Markov prediction task, under batch updating, TD converges for sufficiently small α.
- Constant-α MC also converges under these conditions, but to a different answer!

TD methods bootstrap and sample

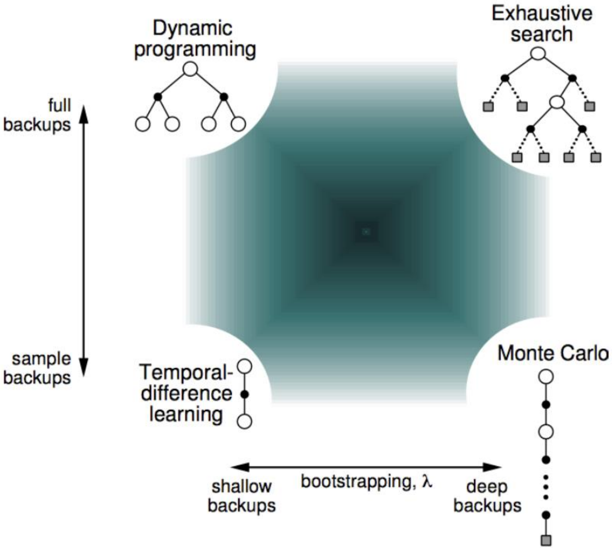
Bootstrapping: update involves an estimate of the value function

- TD and DP methods bootstrap
- MC methods do not bootstrap

Sampling: update does not involve an expected value

- TD and MC method sample
- Classical DP does not sample

Unified View of Reinforcement Learning



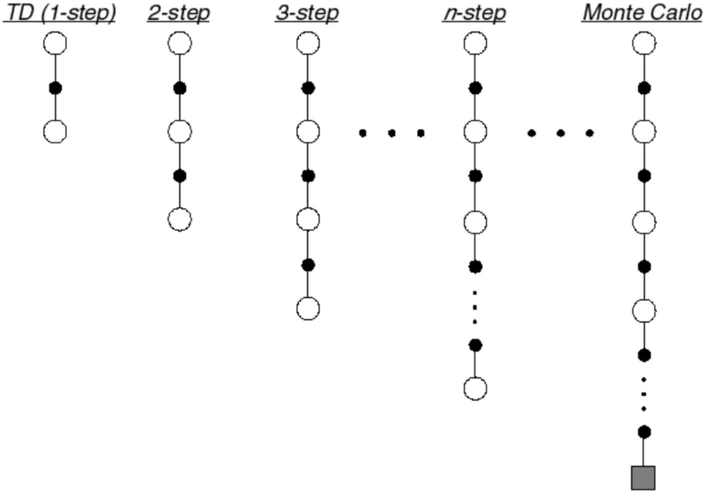
Multi-Step Updates

Multi-Step Updates

- TD uses value estimates which might be inaccurate
- In addition, information can propagate back quite slowly
- In MC information propagates faster, but the updates are noisier
- We can go in between TD and MC

Multi-Step Updates

Let TD target look n steps into the future



Multi-Step Updates

Consider the following n-step returns for $n = 1, 2, \infty$:

$$
\begin{array}{lll}
n = 1 & (TD) & G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1}) \\
n = 2 & & G_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2}) \\
\vdots & & \vdots \\
n = \infty & (MC) & G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T
\end{array}
$$

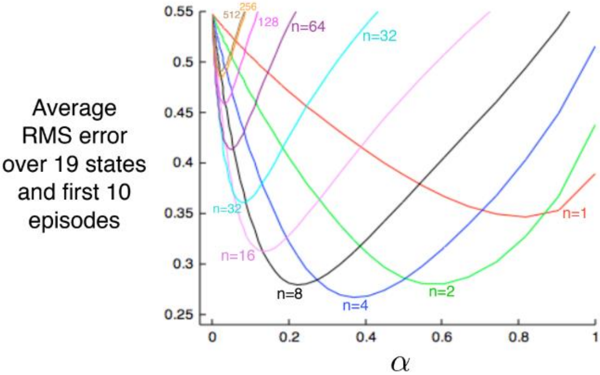
Multi-step temporal-difference learning

$$
V(S_t) \leftarrow V(S_t) + \alpha \left(G_t^{(n)} - V(S_t) \right)
$$

Large Random Walk

- Assume a 19-state random walk example.
  - How does 2-step TD work here?
  - How about 3-step TD?

Large Random Walk



n-step TD results

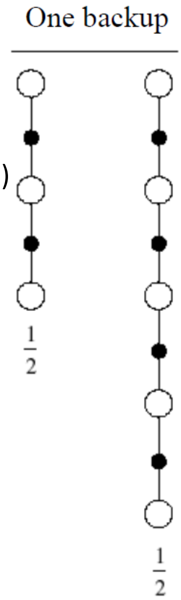
- An intermediate α is best
- An intermediate n is best
- Do you think there is an optimal n? for every task?

Averaging n-Step Returns

- We can average n-step returns over different n
  - e.g. average the 2-step and 4-step returns (Called a compound backup)

$$\frac{1}{2}G^{(2)} + \frac{1}{2}G^{(4)}$$

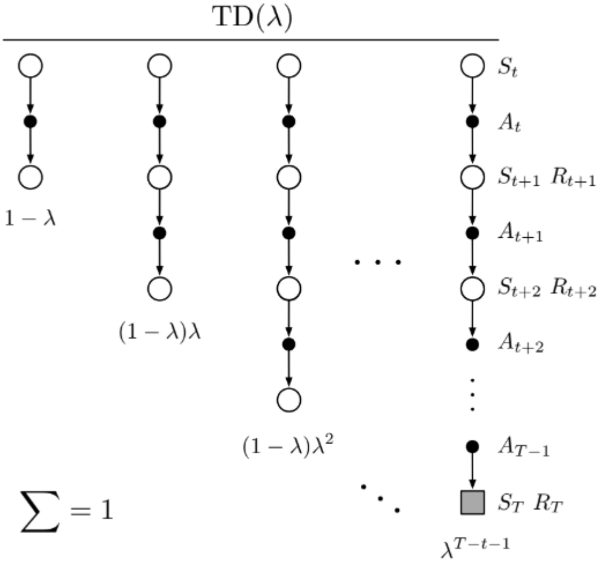
- Combines information from two different time-steps
- Can we efficiently combine information from all time steps?



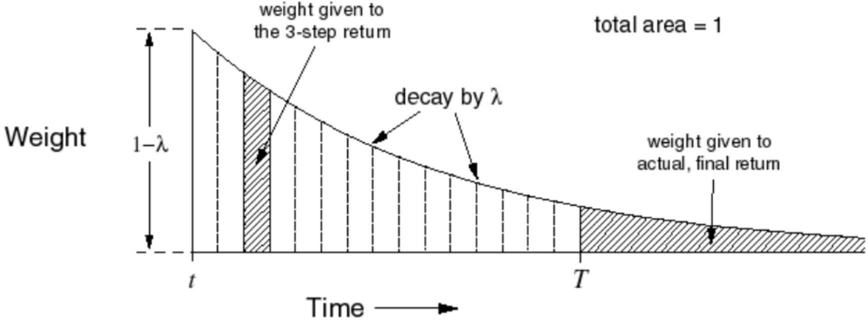
The λ-return is a compound update target

- The λ-return a target that averages all n-step targets
  - each weighted by λn-1

$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$



TD(λ) Weighting Function

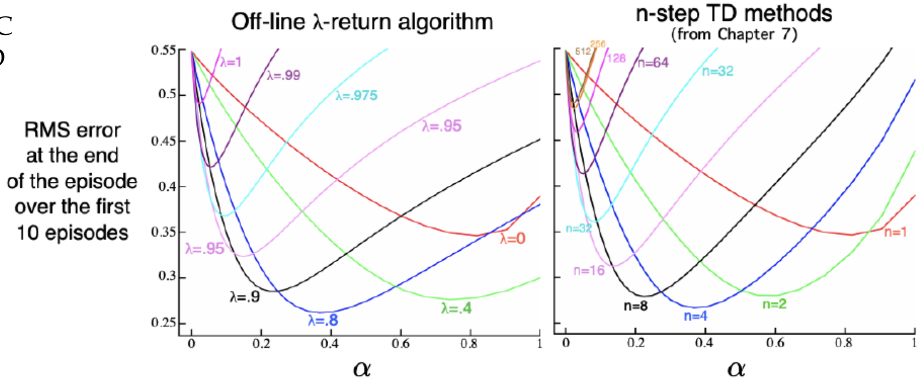


$$G_t^{\lambda} = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

Relation to TD(0) and MC

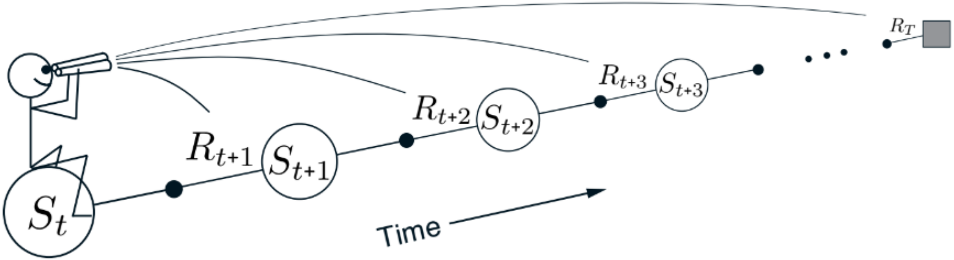
When

- λ=1, MC
- λ=0, TD



Intuition: $1/(1-\lambda)$ is the 'horizon'. E.g., $\lambda = 0.9 \approx n = 10$.

Forward-view TD(λ)



- Update value function towards the λ -return
- Forward-view looks into the future to compute G
- Like MC, can only be computed from complete episodes

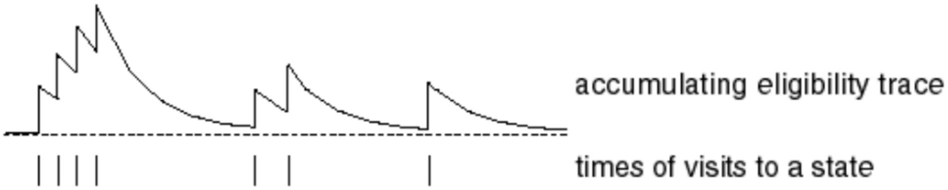
Backward View

- The forward view provides theory
- Backward view provides a mechanism
- Update online, every step, from incomplete sequences

Eligibility Traces

- Frequency heuristic: assign credit to the most frequent states
- Recency heuristic: assign credit to the most recent states
- Eligibility traces combine both heuristics

$$E_0(s) = 0$$
$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbf{1}(S_t = s)$$

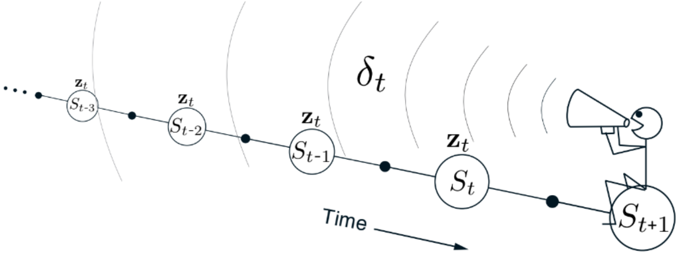


Backward View

- Keep an eligibility trace for every state s
- Update value V(s) for every state s
- In proportion to TD-error δ_t and eligibility trace $E_t(s)$

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$

$$V(s) \leftarrow V(s) + \alpha \delta_t E_t(s)$$



- Shout the TD error backwards
- The traces fade with temporal distance by γλ

TD(λ) and TD(0)

- When λ = 0, only current state is updated

$$E_t(s) = \mathbf{1}(S_t = s)$$
$$V(s) \leftarrow V(s) + \alpha \delta_t E_t(s)$$

- This is exactly equivalent to a TD(0) update

$$V(S_t) \leftarrow V(S_t) + \alpha \delta_t$$

- When λ = 1, credit is deferred until the end of episode

$$E_t(s) = \gamma \lambda E_{t-1}(s) + \mathbf{1}(S_t = s)$$

Summary

- Generalize Temporal-Difference and Monte Carlo learning methods, sliding from one to the other as n increases:

    - $n = 1$ is TD as in Chapter 6
    - $n = \infty$ is MC as in Chapter 5

- An intermediate n is often much better than either extreme applicable to both continuing and episodic problems

End of lecture