UPPSALA UNIVERSITET

Reinforcement Learning

Lecture 4 - Model-free prediction

Per Mattsson

2022

Department of Information Technology

Repetition

Important concepts

UPPSALA UNIVERSITET

- **States, actions and rewards:** $s \in \mathcal{S}$, $a \in \mathcal{A}$, $r \in \mathcal{R}$.
- **Dynamics/model:** $p(s', r|s, a)$.
- **Policy:** $\pi(a|s)$ (For deterministic policy also $a = \pi(s)$.)
- **The return:**

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$

  Is in general a stochastic variable.
- **State-value function:**

  Expected return when starting in s and following policy π,

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t | S_t = s \right].$$

- **Action-value function:**

  Expected return when starting in s, taking action a and *then* follow π,

$$q_\pi(s, a) = \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a \right]$$

- **Bellman equation:**

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')], \quad \text{for all } s \in \mathcal{S}.$$

- **Policy evaluation:**

$$v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')]$$

then $v_k(s) \to v_\pi(s)$.

- **Policy improvement:**

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

$$\pi'(s) = \arg\max_a q_\pi(s, a)$$

- **Policy iteration:** $\pi_0 \xrightarrow{\text{E}} v_{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} v_{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi_* \xrightarrow{\text{E}} v_*.$

- **Bellman optimality equation:**

$$v_*(s) = \max_a q_*(s, a) = \max_a \sum_{r,s'} p(s', r|s, a)[r + \gamma v_*(s')].$$

- **Value iteration:** (Based on Bellman optimality equation)

$$v_{k+1}(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')]$$

  then $v_k(s) \to v_*(s)$.

- **Optimal policy:**

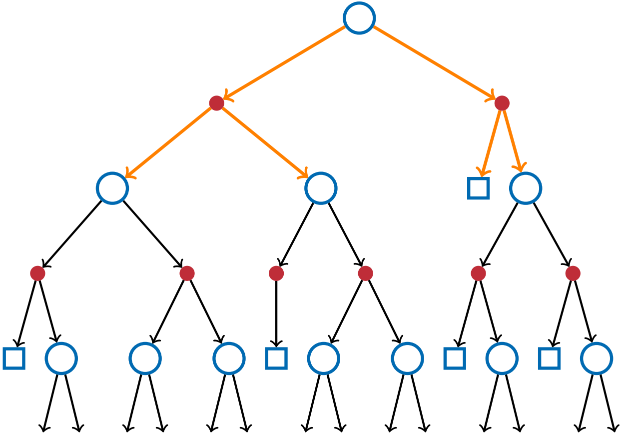
$$q_*(s, a) = \sum_{r,s'} p(s', r|s, a)[r + \gamma v_*(s')]$$

$$\pi_*(s) = \arg\max_a q_*(s, a).$$

Dynamic Programming Backup Diagram

$$V(s) \leftarrow \mathbb{E}_\pi \left[R_{t+1} + \gamma V(S_{t+1}) | S_t = s\right]$$

We need $p(s', r|s, a)$ to compute the expected value.



Monte-Carlo Methods

Example: Expected sum of throw with two dice

UPPSALA UNIVERSITET

- **Problem:** We throw two dice, and call their sum G. What is $V = \mathbb{E}[G]$?
- **By hand:**
  - Each dice has 6 sides, so we can get $6 \times 6 = 36$ combinations.
  - There is no way to get $G = 1$, so $p(1) = 0$.
  - There is one way to get $G = 2$, so $p(2) = 1/36$.
  - There are two ways to get $G = 3$, so $p(3) = 2/36$.
  - Etc.
  - Finally, $\mathbb{E}[G] = \sum_{g=1}^{12} g p(g) = 7$.
- **Monte-Carlo:**
  - Throw many times and get independent observations $G_1, G_2, G_3, \ldots, G_n$.
  - Use the empirical mean to estimate $V = \mathbb{E}[G]$:

$$\hat{V}_n = \frac{1}{n} \sum_{k=1}^{n} G_k$$

  - Do not need any information about how a dice works!
- **Law of large numbers:** $\hat{V}_n \to \mathbb{E}[G]$ as $n \to \infty$.

Using the estimate

$$\hat{V}_n = \frac{1}{n}\sum_{k=1}^{n} G_k$$

Trail	$n = 1$	$n = 10$	$n = 100$	$n = 1000$
1.	12.00	7.10	6.94	7.07
2.	9.00	7.20	7.30	6.92
3.	7.00	7.50	6.75	6.95
4.	4.00	4.90	7.02	7.01
5.	5.00	8.10	7.62	7.09

Bias and variance

- Let $\hat{\theta}_n$ be an estimate of θ using n random samples.
- Since the samples are random, $\hat{\theta}_n$ is a stochastic variable.
- We can thus talk about the expected value and variance of $\hat{\theta}_n$.
- **Bias:** (Unbiased if bias$=0$)

$$\text{Bias}(\hat{\theta}_n) = \mathbb{E}[\hat{\theta}_n] - \theta.$$

- **Variance:**

$$\text{Var}(\hat{\theta}_n) = \mathbb{E}[(\hat{\theta}_n - \mathbb{E}[\hat{\theta}_n])^2]$$

- **The mean squared error (MSE):**

$$\text{MSE}(\hat{\theta}_n) = \mathbb{E}[(\hat{\theta}_n - \theta)^2] = \text{Var}(\hat{\theta}_n) + \text{Bias}(\hat{\theta}_n)^2$$

- **Consistent** if $\hat{\theta}_n \to \theta$ as $n \to \infty$.

Example: Throwing two dice

- We want to estimate $V = \mathbb{E}[G]$.
- We use observations $G_1, \ldots, G_n$ to form the estimate

$$\hat{V}_n = \frac{1}{n}\sum_{k=1}^{n} G_k.$$

- **Bias:**

$$\text{Bias}(\hat{V}_n) = \mathbb{E}[\hat{V}_n] - V = \mathbb{E}\left[\frac{1}{n}\sum_{k=1}^{n} G_k\right] - V = \frac{1}{n}\sum_{k=1}^{n}\mathbb{E}[G] - V = \mathbb{E}[G] - V = 0$$

  so the estimate is unbiased for all n.
- **Variance:**

$$\text{Var}(\hat{\theta}_n) = \mathbb{E}[(\hat{V}_n - \mathbb{E}[\hat{V}_n])^2] = \frac{35}{6n} \approx \frac{5.83}{n}$$

- So, as $n \to \infty$ the variance goes to zero.

Incremental updates

- If we have already computed

$$\hat{V}_{n-1} = \frac{1}{n-1} \sum_{j=1}^{n-1} G_j,$$

and I get one more observation G_n:

$$\hat{V}_n = \frac{1}{n} \sum_{j=1}^{n} G_j = \frac{1}{n} \left(G_n + \sum_{j=1}^{n-1} G_j \right)$$
$$= \frac{1}{n} \left(G_n + (n-1)\hat{V}_{n-1} \right) = \hat{V}_{n-1} + \frac{1}{n}(G_n - \hat{V}_{n-1}).$$

- So, we can start from $\hat{V} = 0$, $n = 0$ and for each new observation G do

$$n \leftarrow n + 1$$
$$\hat{V} \leftarrow \hat{V} + \frac{1}{n}(G - \hat{V}).$$

- Fast update, and don't need to remember all old observations!

Incremental updates

$$\underbrace{\hat{V}}_{\text{New Estimate}} \leftarrow \underbrace{\hat{V}}_{\text{Old Estimate}} + \underbrace{\alpha_n}_{\text{Step size}} \left[\underbrace{G}_{\text{Target}} - \underbrace{\hat{V}}_{OldEstimate} \right]$$

- In each step, move the estimate a bit closer to the observed "target".
- For empirical mean the step size $\alpha_n = \frac{1}{n}$.
- For i.i.d observations of G this converge to $\mathbb{E}[G]$ if

$$\sum_{n=1}^{\infty} \alpha_n = \infty, \quad \sum_{n=1}^{\infty} \alpha_n^2 < \infty.$$

- **Non-stationary** problems: Can use a constant $\alpha \in (0, 1)$ to "forget" old observations. Variance do not go to zero, but can adjust to changing probabilities.
- **Extreme cases:**
  - $\alpha = 0$ gives $\hat{V} \leftarrow \hat{V}$. We learn nothing.
  - $\alpha = 1$ gives $\hat{V} \leftarrow G$. We forget all past observations!

Monte-Carlo Prediction

UPPSALA UNIVERSITET

- Lets consider episodic tasks (that terminate after a finite number of steps).
- **Goal:** Learn $v_\pi(s)$ from experience under π,

$$S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_T \sim \pi.$$

- **The return:**

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

- **The value function:**

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t | S_t = s \right]$$

- **Monte-Carlo:** Estimate $v_\pi(s)$ using the *empirical mean* return of many episodes instead of the *expected* return.
- With MC, we do not need to know what $p(s', r|s, a)$ is!

First-visit vs every-visit

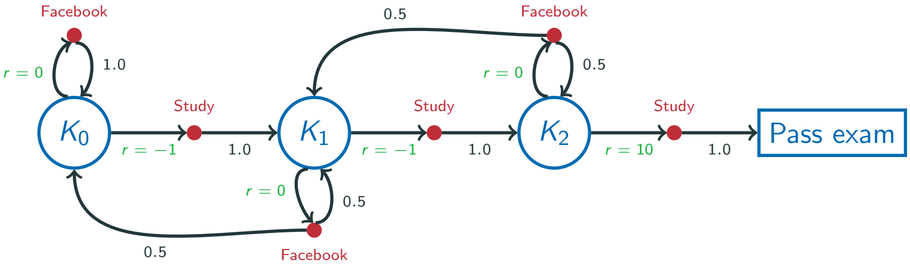
First-visit

1. Sample an episode, $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$ using policy π.
2. **The first** time-step t that state s is visited, add G_t to $Returns(s)$.
3. Let $V(s) = \text{average}(Returns(s))$.
4. Go back to step 1.

Every-visit

1. Sample an episode, $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$ using policy π.
2. **Every** time-step t that state s is visited, add G_t to $Returns(s)$.
3. Let $V(s) = \text{average}(Returns(s))$.
4. Go back to step 1.

Example: Study or Facebook?



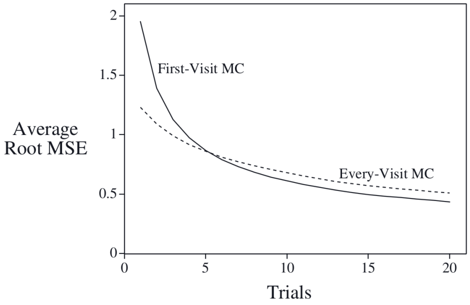
Trajectory:

$$S_1 = K_0, R_2 = -1, S_2 = K_1, R_3 = -1, S_3 = K_2, R_4 = 0,$$
$$S_4 = K_1, R_5 = -1, S_5 = K_2, R_6 = 10, S_6 = \text{Pass}$$

- First visit: For K_1 only use $G_2 = -1 + 0\gamma - 1\gamma^2 + 10\gamma^3$.
- Every visit: For K_1 use G_2 and $G_4 = -1 + 10\gamma$.

Properties of MC

- $+$ **Consistent:** $V(s) \to v_\pi(s)$ as $N(s) \to \infty$.
- $+$ First-visit MC is **unbiased**. (Every-visit can be biased).
- $+/-$ Does not make use of the Markov-property.
- $-$ Generally high variance, and reducing it may require a lot of experience.
- $-$ Must wait until end of episode to compute G_t and update V.



"Reinforcement Learning with Replacing Eligibility Trace", Singh and Barto, 1996.

- In the MC method we compute the average of all observed returns G_t seen in each state.
- **Incremental updates:**
    1. Collect a trajectory $S_0, R_1, S_1, R_2, \cdots, S_T$ following policy π.
    2. For (the first-visit/every-visit) S_t compute G_t and let

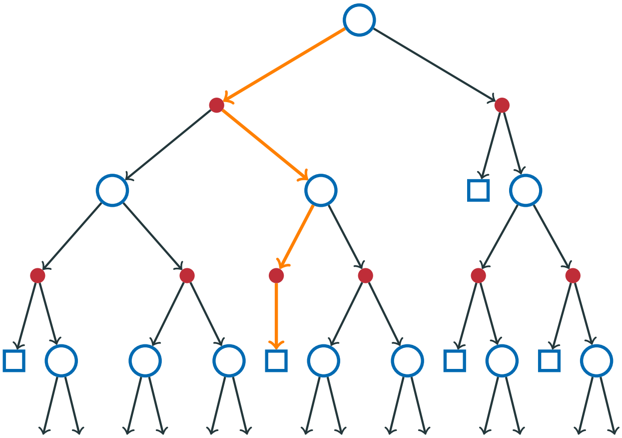
$$\begin{aligned} N(S_t) &\leftarrow N(S_t) + 1 \\ V(S_t) &\leftarrow V(S_t) + \alpha_n(G_t - V(S_t)). \end{aligned}$$

- **For empirical mean:** $\alpha_n = \frac{1}{N(S_t)}$.
- **For e.g. non-stationary environment** we may instead use a constant α.

Monte Carlo Backup Diagram

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}\left(G_t - V(S_t)\right)$$

The observations G_t can be computed from experience.



Monte-Carlo vs Dynamic Programming

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s]$$

Dynamic Programming:

$$V(s) \leftarrow \mathbb{E}_\pi[R_{t+1} + \gamma V(S_{t+1}) | S_t = s].$$

- **Bootstrapping:** Each new estimate is based on a previous estimate.
- Computes expectation exactly, but estimate since based on estimate $V(S_{t+1})$.
- Needs model $p(s', r | s, a)$ to compute expectation.

Monte Carlo:

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

- We do not bootstrap, since we use the full return G_t.
- Is an estimate because we use empirical mean of G_t, and not $\mathbb{E}_\pi[G_t | S_t = s]$.
- No model needed, since samples G_t can be computed from experience.

Can we combine bootstrapping and learning from experience?

Temporal-Difference (TD) Learning

Temporal-Difference (TD) Learning

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi[G_t | S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s]
\end{aligned}
$$

- **MC:** We use the *target* G_t:

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

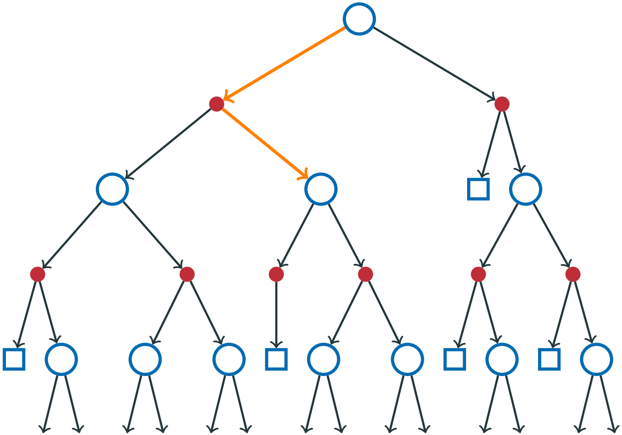
- **TD:** We use the *TD-target* $R_{t+1} + \gamma V(S_{t+1})$:

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$

- Often called TD(0), since it is a special case of TD(λ) (with $\lambda = 0$).
- TD **bootstraps**: The new estimate $V(S_t)$ is based on the estimate $V(S_{t+1})$.

$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$

We do not have to complete a full episode to make the update!



TD-learning

UPPSALA UNIVERSITET

- Initialize the estimate V (e.g. $V(s) = 0$ for all s)
- Start in some state S.
- Loop
  1. Take action A according to policy $\pi(a|S)$.
  2. Observe reward R and new state S'.
  3. $V(S) \leftarrow V(S) + \alpha[R + \gamma V(S') - V(S)]$
  4. $S \leftarrow S'$.
- (If the task is episodic, we would have to re-run the above loop for several episodes).

Note: We do not have to complete the episode before we start learning, and we can even learn in continuing tasks.

The bias

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t | S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] \end{aligned}$$

The MC-target G_t

- **Unbiased** estimate of $v_\pi(S_t)$.
- Not based on previous estimates. (No bootstrapping)

The "true TD-target": $R_{t+1} + \gamma v_\pi(S_{t+1})$

- **Unbiased** estimate of $v_\pi(S_t)$.
- Cannot be computed when we do not know $v_\pi(S_{t+1})$.

The TD-target $R_{t+1} + \gamma V(S_{t+1})$

- **Biased** estimate of $v_\pi(S_t)$.
- Based on old estimate of $V(S_{t+1})$ (Bootstrapping)

UPPSALA UNIVERSITET

Monte Carlo (MC):

- Only in episodic environments.
- High variance, zero bias.
- Converges to $v_\pi(s)$ (if α decrease with a suitable rate)
- (good convergence properties even for function approximation)
- Not very sensitive to initial conditions.
- Usually more efficient in non-MDP environments.

Temporal Differences (TD):

- Both episodic and continuing environments.
- Low variance, but some bias.
- Converges to $v_\pi(s)$ (if α decrease with a suitable rate)
- (but do not always converge with function approximation)
- More sensitive to initial conditions.
- Usually more efficient in MDP environments.

Batch MC and TD

- MC and TD converge as experience $\to \infty$.
- Lets assume that we only have a finite number of episodes of experience:

$$S_0^1, A_0^1, R_1^1, \cdots, S_T^1$$
$$\vdots$$
$$S_0^K, A_0^K, R_1^K, \cdots, S_T^K$$

- Batch learning: Repeatedly apply MC or TD to the episodes we have.
- With constant (but small) α both methods converge, but to different estimates.
- **Batch MC:**
  Minimize the mean-squared error $\sum(G_t - V(S_t))^2$ on training data.
- **Batch TD:**
  Equivalent to finding the maximum likelihood estimate of $p(s', r|s, \pi(s))$, and then computing the value function according to the estimated model.

AB Example

Two states (A and B), $\gamma = 1$.

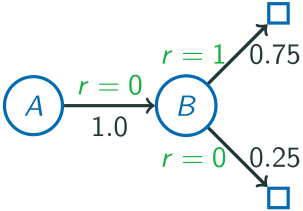
Experience:

$$A, 0, B, 0$$
$$B, 1$$
$$B, 1$$
$$B, 1$$
$$B, 1$$
$$B, 1$$
$$B, 1$$
$$B, 0$$

Batch MC: Converges to

$$V(A) = 0, \quad V(B) = 0.75$$

Batch TD: Converges to



and thus

$$V(A) = 0.75, \quad V(B) = 0.75.$$

Extensions*

n-step return and TD(λ)

- In MC we use the full return as target:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + R_T.$$

- In TD, we only take one step and then use our previous estimate. That is the TD-target

$$G_{t:t+1} = R_{t+1} + \gamma V(S_{t+1}).$$

- In the same way we could define a 2-step return

$$G_{t:t+2} = R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2})$$

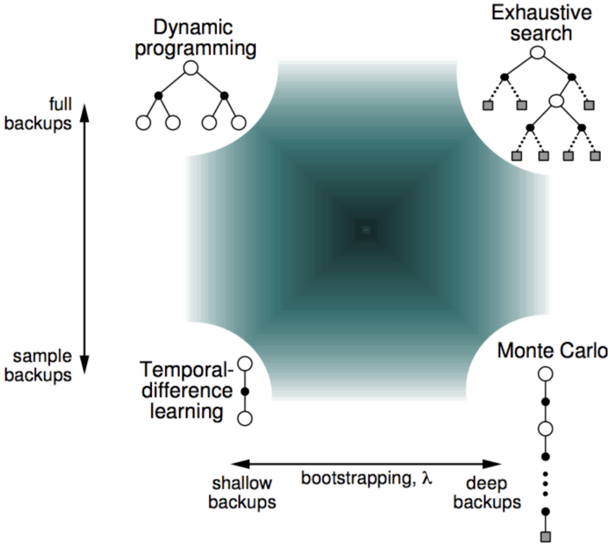
  and so on.

- In TD(λ) we average between different n-step returns and use

$$G_t^\lambda = (1-\lambda) \sum_{n=1}^{T-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t-1} G_t.$$

  Note that $\lambda = 0$ gives the 1-step return, and $\lambda = 1$ gives MC.

Unified view of RL



Summary

- How to estimate v_π from experience using MC and TD(0).
- Both converge to v_π in tabular case (if α decrease with a suitable rate).
- MC - unbiased (first-visit), high variance.
- TD - biased, relatively low variance.
- TD usually more efficient in MDP-environments.

Next:

- How to find a good policy when $p(s', r|s, a)$ is unknown.
- You can now also look at Tinkering Notebook 3a.