Reinforcement Learning: Tutorial 8

On-policy TD learning with approximation

Week 4
University of Amsterdam

Milena Kapralova
September 2024

Check-in

- How is it going?
- How is HW3?
- Are you ready to start HW4?
- If you have any feedback so far, please mail me at *m.kapralova@uva.nl*

Outline

1. Admin
2. On-policy TD learning with approximation exercises
3. Ask anything about HW3 or 7.4 (HW4)

Admin

- Reminder that HW3 deadline is tomorrow @ 17:00
- Any questions?

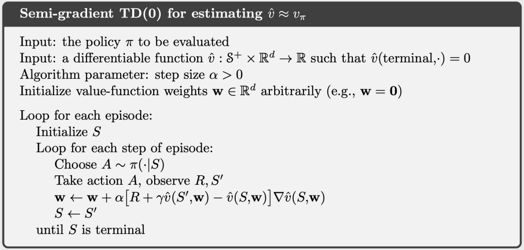
Tutorial 8 Overview

1. On-policy TD learning with approximation exercises
2. Ask anything about HW3 or 7.4 (HW4)

Tutorial 8 Overview

1. On-policy TD learning with approximation exercises
    - Questions 7.1-7.3
2. Ask anything about HW3 or 7.4 (HW4)

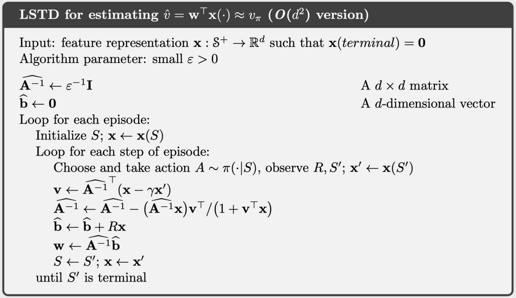
Theory Intermezzo: Semi-gradient TD(0), LSTD



Semi-gradient TD(0) for estimating $\hat{v} \approx v_\pi$

```
Input: the policy $\pi$ to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$
Algorithm parameter: step size $\alpha > 0$
Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
   Initialize $S$
   Loop for each step of episode:
      Choose $A \sim \pi(\cdot|S)$
      Take action $A$, observe $R, S'$
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})\big]\nabla\hat{v}(S,\mathbf{w})$
      $S \leftarrow S'$
   until $S$ is terminal
```

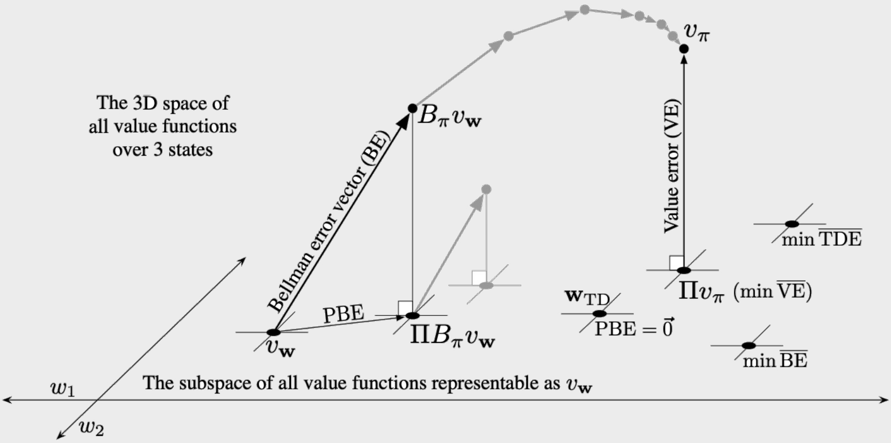


LSTD for estimating $\hat{v} = \mathbf{w}^\top \mathbf{x}(\cdot) \approx v_\pi$ ($O(d^2)$ version)

```
Input: feature representation $\mathbf{x} : \mathcal{S}^+ \to \mathbb{R}^d$ such that $\mathbf{x}(terminal) = \mathbf{0}$
Algorithm parameter: small $\varepsilon > 0$

$\widehat{\mathbf{A}^{-1}} \leftarrow \varepsilon^{-1}\mathbf{I}$                                    A $d \times d$ matrix
$\widehat{\mathbf{b}} \leftarrow \mathbf{0}$                                    A $d$-dimensional vector
Loop for each episode:
   Initialize $S$; $\mathbf{x} \leftarrow \mathbf{x}(S)$
   Loop for each step of episode:
      Choose and take action $A \sim \pi(\cdot|S)$, observe $R, S'$; $\mathbf{x}' \leftarrow \mathbf{x}(S')$
      $\mathbf{v} \leftarrow \widehat{\mathbf{A}^{-1}}^\top(\mathbf{x} - \gamma\mathbf{x}')$
      $\widehat{\mathbf{A}^{-1}} \leftarrow \widehat{\mathbf{A}^{-1}} - \big(\widehat{\mathbf{A}^{-1}}\mathbf{x}\big)\mathbf{v}^\top / \big(1 + \mathbf{v}^\top\mathbf{x}\big)$
      $\widehat{\mathbf{b}} \leftarrow \widehat{\mathbf{b}} + R\mathbf{x}$
      $\mathbf{w} \leftarrow \widehat{\mathbf{A}^{-1}}\widehat{\mathbf{b}}$
      $S \leftarrow S'$; $\mathbf{x} \leftarrow \mathbf{x}'$
   until $S'$ is terminal
```

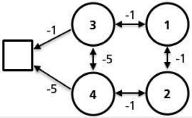
Theory Intermezzo: Geometry of value functions


The 3D space of
all value functions
over 3 states
$B_\pi v_{\mathbf{w}}$
v_π
Bellman error vector (BE)
Value error (VE)
PBE
$v_{\mathbf{w}}$
$\Pi B_\pi v_{\mathbf{w}}$
$\mathbf{w}_{\text{TD}}$
$\text{PBE} = \vec{0}$
Πv_π $(\min \overline{\text{VE}})$
$\min \overline{\text{TDE}}$
$\min \overline{\text{BE}}$
w_1
w_2
The subspace of all value functions representable as $v_{\mathbf{w}}$

Q 7.1 Semi-gradient TD and the TD fixed point

We are considering how to travel to a goal location from various locations labeled 1, 2, 3, and 4. There are different travel costs between these locations. A "map" for this problem (showing the possible actions per state) and the associated costs are summarized in the Figure below. We model the problem as an MDP (Figure below), with discount factor $\gamma = 1$. To use only 2 parameters to represent the value function, approximation can be used. We use a linear approximation $\hat{v}(s, \boldsymbol{w}) = \boldsymbol{w}^T \phi(s)$, For the four states and the terminal (goal) state, we use the following features respectively:

$$\phi(s1) = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \phi(s2) = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \phi(s3) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \phi(s4) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \phi(T) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$



Q 7.1 Semi-gradient TD and the TD fixed point

1. Assume the parameter for the value function approximator in the last time step is $w_t = [0.5, 0.5]^T$, the agent take a new step and receive a transition sample $(s_2, -1, s_4)$, please compute the one step updated value of parameters w_{t+1} with a learning rate α.

Q 7.1 Semi-gradient TD and the TD fixed point

1. Assume the parameter for the value function approximator in the last time step is $w_t = [0.5, 0.5]^T$, the agent take a new step and receive a transition sample $(s_2, -1, s_4)$, please compute the one step updated value of parameters w_{t+1} with a learning rate α.

Following the semi-gradient equation, we need to evaluate

$$\boldsymbol{w_{t+1}} \leftarrow \boldsymbol{w_t} + \alpha[R + \gamma \hat{v}(s', \boldsymbol{w_t}) - \hat{v}(s, \boldsymbol{w_t})]\nabla \hat{v}(s, \boldsymbol{w_t}).$$

With a single sample $(s_2, -1, s_4)$ at hand, we can have $w_{t+1} = [0.5, 0.5]^T + \alpha \cdot (-1 + \gamma * (0.5) - 1.0) \cdot [0, 2]^T = [0.5, 0.5 - 3\alpha]^T$.

Q 7.1 Semi-gradient TD and the TD fixed point

2. What is the relation between the solution found by LSTD and the Semi-gradient TD method?

Q 7.1 Semi-gradient TD and the TD fixed point

2. What is the relation between the solution found by LSTD and the Semi-gradient TD method?

LSTD finds the TD fixpoint. Semi-gradient TD, if it converges, converges to this same TD fix point.

Q 7.1 Semi-gradient TD and the TD fixed point

3. For the MDP, you have access to the following set of trajectories (with actions not shown):

$$\{(s_1, -1, s_3, -1, T), (s_2, -1, s_4, -5, T)\}$$

where the end of an episode means you reached a terminal state. What solution do TD algorithms converge to when repeatedly trained on this dataset with the given feature function ? Hint: consider the previous sub-question.

Q 7.1 Semi-gradient TD and the TD fixed point

Use LSTD on the data. Since there is only a single solution (the 'A' matrix is well conditioned), there is no need to use ϵ. (Also, the TD algorithms wouldn't use such regularization). Compute the sample estimates by considering all steps for each of the two trajectories:

$$
\begin{aligned}
\hat{A}_t &= \phi(s_1^{(1)}) \left(\phi(s_1^{(1)}) - \phi(s_3^{(1)})\right)^T + \phi(s_3^{(1)}) \left(\phi(s_3^{(1)}) - \phi(T^{(1)})\right)^T \\
&+ \phi(s_2^{(2)}) \left(\phi(s_2^{(2)}) - \phi(s_4^{(2)})\right)^T + \phi(s_4^{(2)}) \left(\phi(s_4^{(2)}) - \phi(T^{(2)})\right)^T \\
&= \begin{bmatrix} 2 \\ 0 \end{bmatrix} \left(\begin{bmatrix} 2 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right)^T + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right)^T \\
&+ \begin{bmatrix} 0 \\ 2 \end{bmatrix} \left(\begin{bmatrix} 0 \\ 2 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right)^T + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right)^T \\
&= \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}
\end{aligned}
$$

Q 7.1 Semi-gradient TD and the TD fixed point

$$\hat{b}_t = -\phi(s_1^{(1)}) - \phi(s_3^{(1)}) + 0 - \phi(s_2^{(2)}) - 5\phi(s_4^{(2)}) + 0 = \begin{bmatrix} -3 \\ -7 \end{bmatrix}$$

Then the solution is,

$$w_t \doteq \hat{A}_t^{-1}\hat{b}_t = \begin{bmatrix} -1 \\ -\frac{7}{3} \end{bmatrix}$$

The value functions can then be computed as:

$$\hat{v}(s_1, w_t) = \left[-1, -\tfrac{7}{3}\right] \begin{bmatrix} 2 \\ 0 \end{bmatrix} = -2$$

$$\hat{v}(s_2, w_t) = \left[-1, -\tfrac{7}{3}\right] \begin{bmatrix} 0 \\ 2 \end{bmatrix} = -\frac{14}{3} \approx -4.67$$

$$\hat{v}(s_3, w_t) = \left[-1, -\tfrac{7}{3}\right] \begin{bmatrix} 1 \\ 0 \end{bmatrix} = -1$$

$$\hat{v}(s_4, w_t) = \left[-1, -\tfrac{7}{3}\right] \begin{bmatrix} 0 \\ 1 \end{bmatrix} = -\frac{7}{3} \approx -2.33$$

Q 7.1 Semi-gradient TD and the TD fixed point

4. Comment on the solution you found under 2. Where is the solution good or bad? Where the solution seems to be bad, can you understand why that is the case?

Q 7.1 Semi-gradient TD and the TD fixed point

4. Comment on the solution you found under 2. Where is the solution good or bad? Where the solution seems to be bad, can you understand why that is the case?

   In the ‘top route’ of the MDP the features are able to capture the value function perfectly. In the 'bottom route' of the MDP, the features capture the value function badly. The value at s_2 should be -3. The only way to do that with the given features is to set w_2 to $-3/2$. However, the value at s_4 clearly needs to be -4. The only way to do is to set w_2 to -4. The solution given by the algorithm makes a trade-off of taking into account the on-policy distribution μ, ending up somewhere in the middle.

Q 7.1 Semi-gradient TD and the TD fixed point

5. The TD fixed point is independent of the learning rate and certain algorithms based on finding it are said to "never forget". Elaborate what is meant by this and provide one advantage and one disadvantage of "never forgetting".

Q 7.1 Semi-gradient TD and the TD fixed point

5. The TD fixed point is independent of the learning rate and certain algorithms based on finding it are said to "never forget". Elaborate what is meant by this and provide one advantage and one disadvantage of "never forgetting".

Advantage: If you never forget a datapoint your method is more sample efficient as you do not throw away any data. Hence you need less data overall.
Disadvantage: If the MDP or the policy changes, never forgetting is a disadvantage since in this case we might want to overwrite older experience with newer experience to correct for this and 'forget' old datapoints to a certain extent. This is talked about in the recording of Lecture 5.

Q 7.1 Semi-gradient TD and the TD fixed point

6. (Deep) neural networks are popularly used as function approximators (instead of linear function approximators). In this case $\hat{v}(s, \boldsymbol{w}) = \mathrm{NN}_w(s)$, where NN_w is a neural network with parameters (weights and biases) w. Assuming you have access to a 'autograd()' function that stores $\partial\, \mathrm{NN}_w(s)/\partial w$ to w.grad, how would you implement an update of the v-function for a transition (s, a, r, s', a')?

Q 7.1 Semi-gradient TD and the TD fixed point

6. (Deep) neural networks are popularly used as function approximators (instead of linear function approximators). In this case $\hat{v}(s, \boldsymbol{w}) = \text{NN}_w(s)$, where NN_w is a neural network with parameters (weights and biases) w. Assuming you have access to a 'autograd()' function that stores $\partial \, \text{NN}_w(s)/\partial w$ to w.grad, how would you implement an update of the v-function for a transition (s, a, r, s', a')?

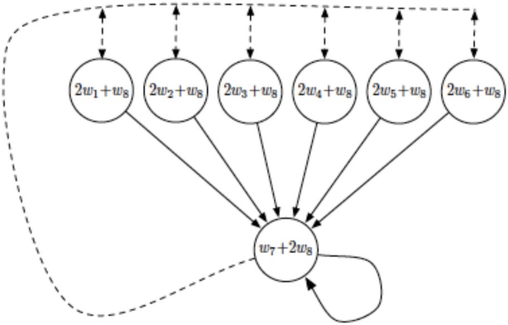
   - val $\leftarrow \text{NN}_w(s)$ (forward pass)
   - valprime $\leftarrow \text{NN}_w(s')$ (forward pass)
   - val.backward() (backward pass)

   Then evaluate:

   - $\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha[R + \gamma \text{valprime} - \text{val}]\boldsymbol{w}$.grad.

Q 7.2 Preparatory question: Off-policy approximation

Off-policy learning with approximation is a tricky topic. Before we'll dive into it in the next chapter, we'll investigate what happens if we apply the methods you know so far in this setting. On Canvas, you'll find a notebook prepared with an exercise on a problem called 'Baird's Counterexample'.



$\pi(\text{solid}|\cdot) = 1$

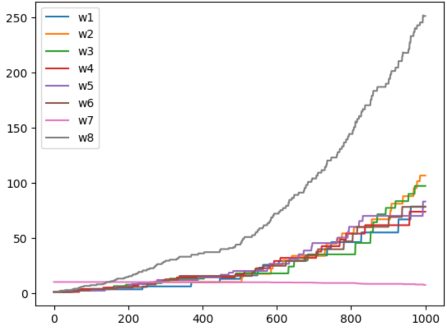
$b(\text{dashed}|\cdot) = 6/7$

$b(\text{solid}|\cdot) = 1/7$

$\gamma = 0.99$

Q 7.2 Preparatory question: Off-policy approximation

Off-policy learning with approximation is a tricky topic. Before we'll dive into it in the next chapter, we'll investigate what happens if we apply the methods you know so far in this setting. On Canvas, you'll find a notebook prepared with an exercise on a problem called 'Baird's Counterexample'.



'Deadly triad':

- Function approximation
- Semi-gradient bootstrapping
- Off-policy training

Q 7.3 *Exam question: Errors and function approximation

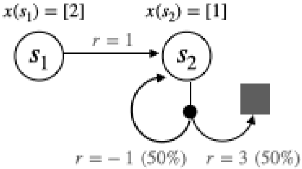
1. Indicate which of the following statements are true:
   - If the mean-squared value error is zero, the mean-squared Bellman error is zero.
   - If the mean-squared Bellman error is zero, then the mean-squared value error is zero.
   - Regardless of features, with linear function approximation there is always a $\hat{v}_w$ such that the MSBE is zero.
   - If the mean squared value error is zero, the mean squared TD error is zero.

Q 7.3 *Exam question: Errors and function approximation

1. Indicate which of the following statements are true:
    - If the mean-squared value error is zero, the mean-squared Bellman error is zero.
    - If the mean-squared Bellman error is zero, then the mean-squared value error is zero.
    - Regardless of features, with linear function approximation there is always a $\hat{v}_w$ such that the MSBE is zero.
    - If the mean squared value error is zero, the mean squared TD error is zero.

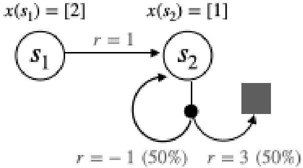
Items 1 and 2 are correct, 3 and 4 are incorrect.

Q 7.3 *Exam question: Errors and function approximation



2. Consider the MDP in the Figure above. With all trajectories starting in s_1, the on-policy distribution μ for the MDP shown is given by $\mu(s_1) = 1/3, \mu(s_2) = 2/3$. Briefly explain what this means and why these values are correct.

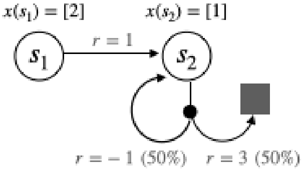
Q 7.3 *Exam question: Errors and function approximation



2. Consider the MDP in the Figure above. With all trajectories starting in s_1, the on-policy distribution μ for the MDP shown is given by $\mu(s_1) = 1/3, \mu(s_2) = 2/3$. Briefly explain what this means and why these values are correct.

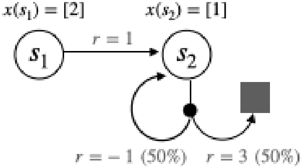
In expectation, 1/3 of the time is spend in s1 and 2/3 of the time is spend in s2. Each episodes starts in s1 and stays there for 1 step. s2 will have at least 1 visit, or ≥ 2 visits with probability 1/2, or ≥ 3 visits with probability 1/4. We know that this sequence adds up to 2 visits on average. So that makes 1/3 steps in s1 and 2/3 steps in s2.

Q 7.3 *Exam question: Errors and function approximation



3. Consider the same MDP. What is the mean squared temporal difference error ($\overline{\text{TDE}}$) in the above example when $\mathbf{w} = [1]$?

Q 7.3 *Exam question: Errors and function approximation



The MSTD error is given by $\sum_s \mu(s) \sum_a \pi(a|s) \sum_{s'} \delta(s, a, s')^2$. So, in this case we get (adding over the three kinds of transitions):

$$1/3 * \delta(s_1, a_1, s_2)^2 + 1/3 * \delta(s_2, a_1, s_2)^2 + 1/3 * \delta(s_2, a_1, T)^2$$

$$\rightarrow \quad \begin{aligned} \delta(s_1, a_1, s_2) &= 1 + 1 - 2 = 0 \\ \delta(s_2, a_1, s_2) &= -1 + 1 - 1 = -1 \\ \delta(s_2, a_1, T) &= 3 + 0 - 1 = 2 \end{aligned}$$

Which gives a total MSTDE of: $1/3 * 0 + 1/3 * 1 + 1/3 * 4 = 5/3$

WELL DONE!

Tutorial 8 Overview

1. On-policy TD learning with approximation exercises
2. Ask anything about HW3 or 7.4 (HW4)
    - Questions 5.2-5.3, 6.3, 7.4

Ask anything about HW3 or 7.4 (HW4)

- 5.2: Coding (+ Little bit of theory)
- 5.3: Theory
- 6.3: Theory
- 7.4: Theory

That's it!

Good luck with the HW and see you on Monday!