Reinforcement Learning: Tutorial 7

MC learning with approximation

Week 4
University of Amsterdam

Milena Kapralova
September 2024

Check-in

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

Outline

1. Admin
2. MC methods with approximation exercises
3. Ask anything about HW3

Admin

- Please direct any questions about grading to Pieter Pierrot
- Any questions?

Tutorial 7 Overview

1. MC methods with approximation exercises
2. Ask anything about HW3

Tutorial 7 Overview

1. MC methods with approximation exercises
   - Questions 6.1-6.2
2. Ask anything about HW3

Q 6.1 Basis functions

1. Tabular methods can be seen as a special case of linear function approximation. Show that this is the case and give the corresponding feature vectors.

Q 6.1 Basis functions

1. Tabular methods can be seen as a special case of linear function approximation. Show that this is the case and give the corresponding feature vectors.

   Let s be a state index, $\vec{s}$ its feature vector and $\vec{w}$ a weight vector. Then for linear function approximation, $v(s;\vec{w}) = \vec{s} \cdot \vec{w}$. If we let the feature vector $\vec{s}$ be a vector that is zero everywhere, except at the index corresponding to the state's tabular index, calling $v(s;\vec{w})$ for state i will simply return the i'th weight, which will correspond to that state's value.

Q 6.1 Basis functions

2. You want to design the feature vectors for a state space with $s = [x, y]$. You expect that x and y interact in some unknown way. How would you design a polynomial feature vector for s?

Q 6.1 Basis functions

2. You want to design the feature vectors for a state space with $s = [x, y]$. You expect that x and y interact in some unknown way. How would you design a polynomial feature vector for s?

   Any feature vector of the form $[1, x, y, xy, \ldots]$ should be fine, as long as they have interaction variables.

Q 6.1 Basis functions

3. What happens to the size of the polynomial feature vector if the number of variables in your state space increases?

Q 6.1 Basis functions

3. What happens to the size of the polynomial feature vector if the number of variables in your state space increases?

   It grows exponentially in the number of state space variables.

Q 6.1 Basis functions

4. You are working on a problem with a state space consisting of two dimensions. You expect that one of them will have a larger impact on the performance than the other. How might you encode this prior knowledge in your basis function?

Q 6.1 Basis functions

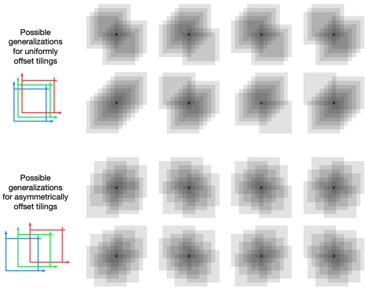

Figure 9.11: Why tile asymmetrical offsets are preferred in tile coding. Shown is the strength of generalization from a trained state, indicated by the small black plus, to nearby states, for the case of eight tilings. If the tilings are uniformly offset (above), then there are diagonal artifacts and substantial variations in the generalization, whereas with asymmetrically offset tilings the generalization is more spherical and homogeneous.

- Asymmetric tile coding (p.219 Fig 9.11)
- By using asymmetric tile coding with respect to the offsets of tilings—applying varied offsets in the more important dimension across multiple tilings while keeping offsets consistent or minimal in the less important dimension—you create a feature representation that is more sensitive to changes in the impactful dimension. This approach encodes your prior knowledge into the basis function, enhancing the learning algorithm's focus on the dimension that has a larger impact on performance.

Q 6.1 Basis functions

5. You can view coarse coding as a special case of Radial Basis Functions. Why?

Q 6.1 Basis functions

5. You can view coarse coding as a special case of Radial Basis Functions. Why?

   In coarse coding, everything within a feature's receptive field is 1, and everything outside of it is 0. Radial basis functions soften this approach, giving a value between 0 and 1 depending on the degree that the feature is present. More generally, any function that only depends on the distance is an RBF.

Q 6.2 Function approximation and state distribution

1. Consider the state distribution, $\mu(s)$. How does it depend on the parameters of the value function approximator if we update the policy to e.g. the ϵ-greedy one?

Q 6.2 Function approximation and state distribution

1. Consider the state distribution, $\mu(s)$. How does it depend on the parameters of the value function approximator if we update the policy to e.g. the ϵ-greedy one?

   $\mu(s)$ is dependent on the policy, which is controlled by the value function approximator. Thus, when the parameters change, the policy changes and so $\mu(s)$ does too.

Q 6.2 Function approximation and state distribution

2. How does this differ from the data distribution in standard (un-)supervised learning problems?

Q 6.2 Function approximation and state distribution

2. How does this differ from the data distribution in standard (un-)supervised learning problems?

   In standard ML, the data distribution is independent of the learned parameters (e.g., in an image classification task, the type of images you encounter do not depend on the classifier learned so far). In RL, the states encounter do depend on the current policy.

Q 6.2 Function approximation and state distribution

3. What does this mean for the weighting of the errors (such as in e.g. Eq. 9.1)?
   $\overline{VE}(\mathbf{w}) \doteq \sum_{s \in S} \mu(s)[v_\pi(s) - \hat{v}(s, \mathbf{w})]^2$

Q 6.2 Function approximation and state distribution

3. What does this mean for the weighting of the errors (such as in e.g. Eq. 9.1)?
   $\overline{VE}(\mathbf{w}) \doteq \sum_{s \in S} \mu(s)[v_\pi(s) - \hat{v}(s, \mathbf{w})]^2$

   While we change the parameters, we also change $\mu(s)$ and thus which states contribute most to the error we care about.


WELL DONE!

Tutorial 7 Overview

1. MC methods with approximation exercises
2. Ask anything about HW3
   - Questions 5.2-5.3, 6.3

Ask anything about HW3

- 5.2: Coding (+ Little bit of theory)
  - Tip: Check out the openai's gym documentation, especially env.step(action) and env.reset() are useful
- 5.3: Theory
- 6.3: Theory

That's it!

See you on tomorrow!