Note: The references mentioned in the exercises refer to the textbook (Sutton and Barto) in the 2nd edition.

9 Functional Approximation & Policy Gradient Methods

46. Autonomous Orchard

    An AI controlled orchard needs to decide when to harvest its trees. To do this it measures the concentration of three chemicals in the air. Each day the orchard can choose to wait or harvest. Waiting costs one credit in operating costs while a harvest ends the process. Once a crop is harvested, packaged and sold, the orchard is told the profit or loss of that harvest. Most experts agree that the function mapping the chemical concentrations to the profit is linear with some error. The orchard has several samples of the profits from other harvests:

Concentration of A (ppm)	Concentration of B (ppm)	Concentration of C (ppm)	Profit/Reward (credits)
4	7	1	3
10	6	0	-15
20	1	15	5
4	19	3	21

    Begin to approximate (by hand) the function that maps the state feature vector to Q(state, harvest) using an MC goal. Do a gradient decent step on each sample. A sensible learning rate would be around 0.01, but feel free to try any value.

47. Exercise 9.1 Show that tabular methods such as presented in Part I of this book are a special case of linear function approximation. What would the feature vectors be?

48. Exercise 10.4 Give pseudocode for a differential version of semi-gradient Q-learning.

49. Exercise 11.1 Convert the equation of n-step off-policy TD (7.9) to semi-gradient form. Give accompanying definitions of the return for both the episodic and continuing cases.

50. Exercise 12.1 Just as the return can be written recursively in terms of the first reward and itself one-step later (3.9), so can the λ-return. Derive the analogous recursive relationship from (12.2) and (12.1).

51. Implementation Task: Reinforce

    Implement the "Reinforce" algorithm (p. 328) and benchmark with an environment of your choice (i.e. from the textbook or a pre-build environment from https://gym.openai.com).

52. Implementation Task: Reinforce with Baseline

    Implement the "Reinforce with Baseline" algorithm (p. 329) and benchmark with an environment of your choice (i.e. from the textbook or a pre-build environment from https://gym.openai.com). Any approximation may be used and the use of existing software libraries is also allowed for the baseline.

53. Implementation Task: Actor-Critic

    Implement the "One-step Actor-Critic" or the "Actor-Critic with Eligibility Traces" algorithm (p. 332) and benchmark with an environment of your choice (i.e. from the textbook or a pre-build environment from https://gym.openai.com).

54. Implementation Task: Step-Size Parameter

    Recover a well-known result in stochastic approximation theory gives the conditions for convergence with probability 1:

$\sum_{n=1}^{\infty} \alpha_n(a) = \infty$ and $\sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$

Play around with the step-size parameter of any past implementations and observe the impact on convergence and learning speed. Were the above conditions met for previous examples? What do you notice? For which types of scenarios constant step sizes are suitable and for which not?