Coursework (1) for *Introductory Lectures on Optimization*

Xiaoyu Wang
3220104364

October 20, 2024

Excercise 1. Please write out the optimization objective function for reinforcement learning.

Solution of Excercise 1: Among the many applications of reinforcement learning, we have chosen the familiar Q-Learning as a reference for optimizing the objective function, we choose the familiar Q-Learning as a reference for optimizing the objective function.

1. Q-Learning Update Rule in Bellman Equation

   Q-Learning uses the Bellman equation to update the Q-function.
   The update rule can be expressed as:

   $$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$$

   where:
   α is the learning rate, $0 < \alpha \leq 1$.
   r is the immediate reward.
   γ is the discount factor, ranging from [0,1). It determines the degree to which future rewards affect current decisions. A larger γ (close to 1) means a greater emphasis on long-term future returns.
   s' is the next state.
   a' is the next action.

2. Objective Function

   Now we can try to write the objective function of it.
   To simplify the expression, we replace the increased reward value after transferring the Bellman equation at time t(compared to time $t-1$) with the parameter R_t
   The objective of Q-Learning is to maximize the expected cumulative reward.

   $$\max \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R_t \mid s_0, a_0 \right]$$

   where:
   $\mathbb{E}$ denotes the expected value.
   γ is the discount factor, $0 \leq \gamma < 1$.
   R_t is the increased reward at time t step compared with $t-1$.
   s_0 is the initial state.
   a_0 is the initial action.

□

Excercise 2. Is it feasible to use the Uniform Grid method as an optimizer to train a neural network? Why?

Solution of Excercise 2: No,it is not feasible.the reasons are as follows:

1. The Uniform Grid method divides the search space into a grid and evaluates the objective function at each grid point. In the class,we learned that the Uniform Grid method is a **zero order iterative method**. This method is not suitable for training neural networks because the search space of neural networks is high-dimensional and the grid points are too sparse to cover the search space effectively.
2. In the class,we learned that the Uniform Grid method is **without any influence from the accumulated in formation on the sequence of testpoints.** That is to say,Every time a test point is selected, the results of previous test points are not referenced, but rather selected according to a predetermined uniform grid pattern. Neural networks are usually trained using gradient-based optimization methods like **Gradient-Descen** and **backpropagation algorithm** which uses the pervious information to speeds up the loss function to get the least value,leading to faster convergence.
3. The Uniform Grid method does not use gradient information.This makes it inefficient for the loss function of a neural network. But the gradient-based optimization methods can quickly identify and exploit regions of the parameter space where the loss function decreases rapidly.

□

Excercise 3. For the performance analysis of the Uniform Grid Method, Prove that

$$\left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor + 2\right)^n, \text{ and } \left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor\right)^n,$$

coincide up to an absolute constant multiplicative factor if $\epsilon \leq O(\frac{L}{n})$.

Proof of Excercise 3: The constant is e,now we prove it:

$$\because \epsilon \leq O(\frac{L}{n})$$

$$\therefore \epsilon \leq M\frac{L}{n} \ for \ constant \ M$$

$$\therefore \frac{L}{2\epsilon} \geq \frac{L}{2M\frac{L}{n}} = \frac{n}{2M} \geq 2cn$$

where c is a integer.

$$\frac{\left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor + 2\right)^n}{\left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor\right)^n} = \left(1 + \frac{1}{\frac{1}{2}\left\lfloor \frac{L}{2\epsilon} \right\rfloor}\right)^n \leq \left(1 + \frac{1}{cn}\right)^n = \left(\left(1 + \frac{1}{cn}\right)^{cn}\right)^{\frac{1}{c}}$$

$$\because \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = e$$

$$\therefore \left(\left(1 + \frac{1}{cn}\right)^{cn}\right)^{\frac{1}{c}} \underset{n\to\infty}{=} e^{\frac{1}{c}}$$

So, the two functions coincide up to an absolute constant multiplicative factor $e^{\frac{1}{c}}$. □

Excercise 4. Prove that, for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we have

$$\nabla f(\mathbf{y}) = \nabla f(\mathbf{x}) + \int_0^1 \nabla^2 f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x}) d\tau.$$

Proof of Excercise 4: Now we prove it:
$\because$ Newton-Leibniz formula:

$$\int_a^b f(x)dx = F(b) - F(a) \ where \ F'(x) = f(x) \tag{1}$$

now we set $g(\tau) = \nabla f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))$

$\because$ the theorem(1) we have:

$$\nabla f(\mathbf{y}) - \nabla f(\mathbf{x}) = g(1) - g(0) = \int_0^1 g'(\tau) d\tau$$

$$\because g'(\tau) = \nabla f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))' = \frac{\partial \nabla f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))}{\partial \tau}$$

$$= \frac{\partial \nabla f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))}{\partial (\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))} \frac{\partial (\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))}{\partial \tau} = \nabla^2 f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x})$$

$$\therefore \nabla f(\mathbf{y}) - \nabla f(\mathbf{x}) = \int_0^1 \nabla^2 f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x}) d\tau$$

□