Coursework (1) for *Introductory Lectures on Optimization*

Zhou Nan
3220102535

October 29, 2024

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

Solution of Excercise 1:

$$J(\theta) = E_{\tau \sim p_\theta(\tau)}[G(\tau)] = E_{\tau \sim p_\theta(\tau)}[\sum_{t=0}^{T-1} \gamma^t r_{t+1}]$$

The core goal of reinforcement learning is to find a policy π_θ that maximizes the agent's expected total return in a given environment. To achieve this goal, reinforcement learning algorithms adjust the policy parameters θ by optimizing the objective function $J(\theta)$ thereby obtaining a higher expected return.

- Probability Distribution of Trajectories: Under the policy π_θ, the trajectory distribution $p_\theta(\tau)$is influenced by the policy, and different trajectories can be obtained through sampling.
- Return of a Trajectory: The total return $G(\tau)$of each trajectory is the weighted sum of all rewards (discounted rewards).
- In reinforcement learning, the goal is to adjust the policy parameters θ to maximize $J(\theta)$. In other words, to find the optimal policy parameters θ^* such that:

$$\theta^* = \arg\max_\theta J(\theta)$$

□

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

Solution of Excercise 2: It is infeasible to use the Uniform Grid method to train a neural network.

1. Uniform Grid method relies on zero-order information (function evaluations) without leveraging gradient information.
2. Uniform Grid method requires a large number of evaluations per iteration to cover the search space. Each Oracle call (evaluation) involves computing the function value at a specific point.
3. Neural networks, especially deep ones, have large parameter spaces. This results in an impractically large number of grid points, making convergence difficult within reasonable time and computational limits.
4. While theoretically optimal in terms of complexity bounds, the practical overhead renders it ineffective for real-world neural network training.

□

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: since $\epsilon \leq O(\frac{L}{n})$, there exists a constant c and N such that, for any $n \geq N$, we have $\epsilon \leq \frac{1}{4c}\frac{L}{n}$.

$$\frac{1}{2}\lfloor \frac{L}{2\epsilon} \rfloor \geq \frac{1}{2}\lfloor \frac{L}{2\frac{1}{4c}\frac{L}{n}} \rfloor = cn, \forall n \geq N$$

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

$\square$

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: Consider a function

$$F(\tau) = \nabla f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))$$

then, we have $F(0) = \nabla f(\mathbf{x})$ and $F(1) = \nabla f(\mathbf{y})$

$$\begin{aligned}
\frac{dF(\tau)}{d\tau} &= \frac{d}{d\tau}(f_1'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\ f_2'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\ f_3'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) \cdots f_n'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})))^T \\
&= (\frac{d}{d\tau} f_1'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\ \frac{d}{d\tau} f_2'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\ \frac{d}{d\tau} f_3'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\ \cdots \frac{d}{d\tau} f_n'(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})))^T \\
&= (\sum_{j=1}^n f_{1j}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\dot{(}y_j - x_j) \cdots \sum_{j=1}^n f_{nj}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))\dot{(}y_j - x_j))^T \\
&= \begin{bmatrix} f_{11}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) & f_{12}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) & \cdots & f_{1n}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) \\ f_{21}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) & f_{22}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) & \cdots & f_{2n}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) \\ \vdots & \vdots & \ddots & \vdots \\ f_{n1}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) & f_{n2}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) & \cdots & f_{nn}''(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) \end{bmatrix} \begin{bmatrix} y_1 - x_1 \\ y_2 - x_2 \\ \vdots \\ y_n - x_n \end{bmatrix} \\
&= \nabla^2 f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x})
\end{aligned}$$

Then,

$$\begin{aligned}
\int_0^1 \nabla^2 f(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x}))(\mathbf{y} - \mathbf{x}) d\tau &= \int_0^1 \frac{dF(\tau)}{d\tau} d\tau \\
&= F(1) - F(0) \\
&= \nabla f(\mathbf{y}) - \nabla f(\mathbf{x})
\end{aligned}$$

Therefore,

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

□