Reinforcement Learning
Homework 1

Margarida Campos 77039
margarida.moreira.campos@gmail.com

November 29, 2023

1 Multi-armed bandits

Setup

In this exercise we compare different action-value methods to solve the 10-armed bandit problem: $a \in \{1, ..., 10\}$.

To better assess the differences between methods we run each one for 1.000 time steps and we repeat this for 2.000 10-armed-bandit problems, and then average the results. For each problem we sample the actual values $Q(a)$ from a standard normal distribution: $\mathcal{N}(0, 1)$; the reward function for each action and time step, $R_t(a)$, is sampled from $\mathcal{N}(Q(a), 1)$. In Figure 1 we can see an example of the reward distributions for different actions in one of the experiments.

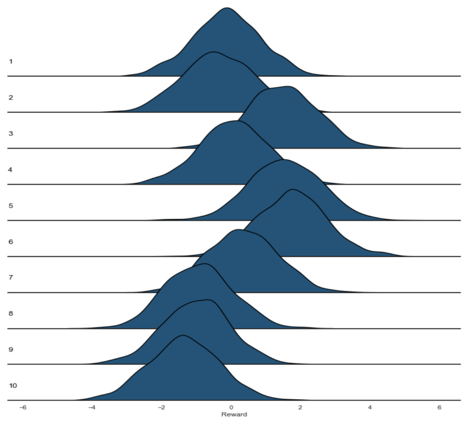

Figure 1: Example of the reward distribution for all ten actions for one of the bandit problems.

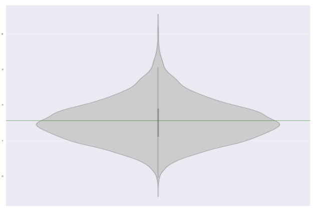

Figure 2: Distribution of maximum Q(a) for the 2000 runs, with highlighted expected value ≈ 1.559

Before we compare the algorithms, it is useful to think what is the maximum average reward we expect to achieve. For each problem, the agent would - ideally - be able to identify the optimal action - a^* , *i.e.*, the one with the maximum $Q(a)$ and would on average get $\mathbb{E}[R_t(a^*)] = Q(a^*)$. The expected value of the maximum of n *i.i.d.* samples $(x_1, .., x_n)$ of a standard normal distribution is approximated by:

$$\mathbb{E}[max(x_1, ..., x_n)] = \Phi^{-1}\left(\frac{n - \frac{\pi}{8}}{n - \frac{\pi}{4} + 1}\right),$$

which for $n = 10$ yields approximately 1.559. One can see in Figure 2 the distribution of the maximum value of each action over 2000 runs. x

Let us look at the results of 5 different methods which allow us to think of the trade-off between exploitation and exploration:

- **Greedy:** uses the greedy policy of always choosing the action with the maximum estimated value
- **Greedy with OIV:** uses the greedy policy, but starts with Optimistic Initial Values (OIV) - motivating exploration by the initial drop of value estimates when taking any action
- **ϵ-Greedy:** uses the greedy policy $(1-\epsilon)\%$ of the time and chooses a random action the remaining times - ensuring some exploration. We will compare $\epsilon = 0.1$ and $\epsilon = 0.01$
- **UCB:** chooses action according to

  $$\text{argmax}\left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}\right],$$

  where $c > 0$ is a user chosen constant wich controls the degree of exploration and $N_t(a)$ is the number of times action a has been taken at time step t. Note that when $N_t(a) = 0$ (unexplored action), the action is considered to be optimal. We use $c = 2$ as a default parameter.

Results

In Figure 3, we can see the average reward (over the 2.000 experiments), each of the methods yields at each time step.

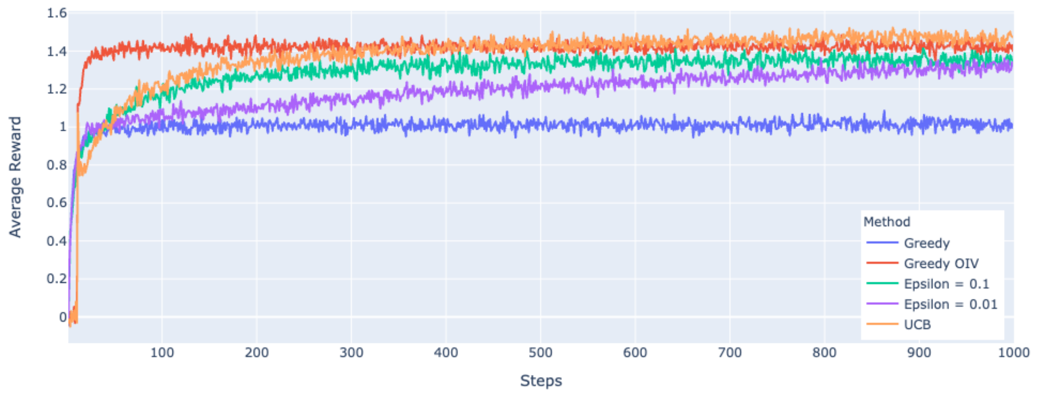

Figure 3: Average performance of all tested methods on the 10-armed testbed.

As expected, all methods start with near zero reward and improve over time. One can see that the greedy method performs the worst in the long run - tending to an average reward of 1, as it gets stuck in suboptimal solutions - since it will stop choosing the optimal action, whenever its initial estimates were poor.

ϵ-greedy methods on the other hand, ensure a degree of exploration which allow the average reward to surpass the greedy's one over time. Note that in the beginning the greedy policy performs better since it is not taking random (eventually bad) actions. It is also interesting to compare $\epsilon = 0.1$ and $\epsilon = 0.01$: the first - exploring more - increases the average reward faster, but the latter will in time surpass it, since $\epsilon = 0.1$ will continue to take a random action 90% of the times as the latter will only choose a suboptimal action 1% of the times. Continuing to apply the ϵ-greedy policy we will never get the optimal reward. In a real life scenario, one would remove the chance of random action after learning was complete.

Outperforming ϵ-greedy policies in the long run, we have greedy OIV which ensured exploration by setting initial estimates of $\hat{Q}(a)$ to 5, hence the agent was forced to iterate over all actions. We can see in figure 4 a zoom in on the first 50 time steps - greedy OIV is stuck near 0 until it explores all 10 actions and then surpasses the greedy approach.

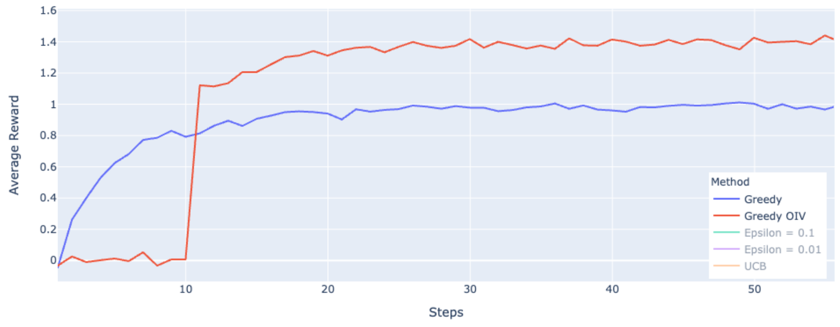

Figure 4: Zoom in: first 50 time steps for Greedy and Greedy with optimistic initial values methods.

The best performing policy in the long run was UCB with $c = 2$ - even though it increases slower than greedy OIV since its exploration is maintained through time and decreases only with the number of times an action is taken. Note, in Figure 5, that similarly to OIV, UCB

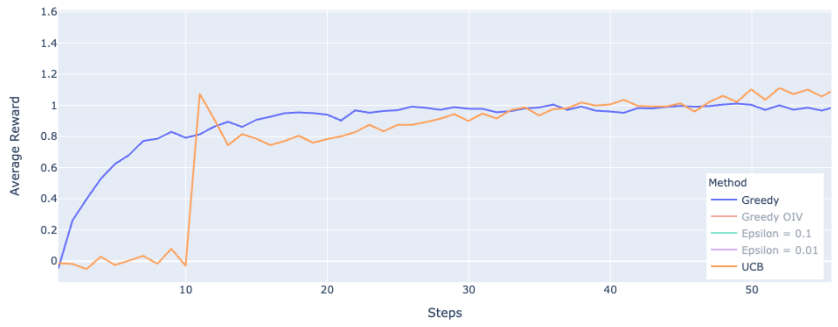

Figure 5: Zoom in: first 50 time steps for Greedy and UCB methods.

also starts by exploring all 10 actions since $N_t(a)$ is 0 for all actions. Then, we can see a spike followed by drop before starting to increase again. The spike happens since all actions have the same $N_t(a) = 1$ and so independently of the value of c, the summed confidence bound is the same for all and the agent will choose the action with maximal Q(a). After that, the optimal action gets added a lower confidence bound since $N_t(a^*) = 2$ and other actions might get chosen even if they have a lower value. The higher the c - parameter that explores the degree of exploration by increasing the size of the intervals considered - the bigger will be the drop, as can be seen in Figure 6. Eventually, as t increases, so will $N_t(a)$ and the agent will tend to choose the actual optimal action.

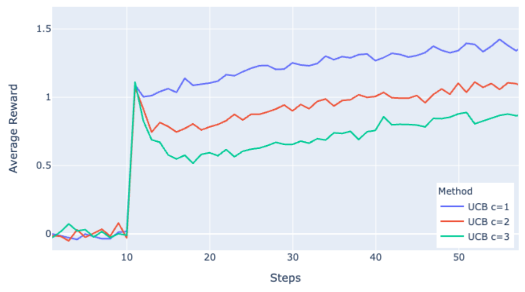

Figure 6: Zoom in: first 50 time steps for UCB methods with different values of c.

2 The Gambler Problem

MDP

We start by modelling the Gambler Problem as an MDP.

- States are the gambler's capital: $S_t \in \mathcal{S}^+ = \mathcal{S} \cup \{0, 100\} = \{1, ..., 99\} \cup \{0, 100\}$ including terminal states
- Actions are the stakes of the agent in the bet: $A_t(s) \in \{0, ..., min(s, 100 - s)\}$
- Reward is 0 except for when agent reaches state 100 (his goal) where the reward is +1.
- Trnasition probabilities are as follows:

  $p(s', 0|s, a) = p_h$, if $s + a = s' \neq 100$;

  $p(s', 1|s, a) = p_h$, if $s + a = s' = 100$

  $p(s', 0|s, a) = 1 - p_h$, if $s - a = s'$

  $p(s', r|s, a) = 0$, othwerwise,

  with p_h as the probability of landing heads, *i.e.*, the player winning the bet.

Setting p_h to 0.4, we solve the problem by using value iteration and a discount rate of $\gamma = 1$, hence for each $s \in \mathcal{S}$, estimated value $V(s)$ will be updated by:

$$V(s) \leftarrow \max_a V(s, a),$$

with

$$V(s,a) = \begin{cases} p_h + (1 - p_h)V(s - a) & \text{if } s + a = 100 \\ p_h V(a + s) + (1 - p_h)V(s - a) & \text{if} s + a \neq 100 \end{cases} \tag{1}$$

Iteration stops when the maximum delta between estimates, $|V^{(k+1)}(s) - V^{(k)}(s)|$, for all states is smaller than a chosen threshold $\theta = 10^{-8}$.

After getting the estimate for the value function ($\hat{V}(s)$) we can find an optimal policy, by choosing π such that $\underset{a}{\text{argmax}}\, \pi(s, a)$, where:

$$\pi(s,a) = \begin{cases} p_h + (1 - p_h)\hat{V}(s - a) & \text{if } s + a = 100 \\ p_h \hat{V}(a + s) + (1 - p_h)\hat{V}(s - a) & \text{if} s + a \neq 100 \end{cases} \tag{2}$$

Results

In Figure 7 we can see the results of applying the aforementioned value iteration procedure with initialization $V^{(0)}(S) = 0$ for all states. The program ran for 15 iterations, having achieved the stopping criteria.

The value function here represents the probability of the agent winning the game given the current capital. Event though, the plot shows a continuous representation, note that the function is only defined for discrete values. Let us look at the first sweep (update of value estimates) in order to better understand the iterative process:

- the value estimate remains 0 for all values under 50, since there is no bet (action) that leads the agent to win
- for capital 50 the agent updates $V(50)$ to be $p_h = 0.4$. For s between 50 and 74, the value will be 0.4 since there is only one action $(100 - s)$ that makes the agent win and not return to a zero-valued state
- when $s = 75$ the scenario is different, since action $a = 25$ yields a greater probability of winning: since the agent can win by finishing the game immediately with probability p_h or go back to state 50 where he has a probability of winning of p_h, hence the value is updated to be $p_h + p_h(1 - p_h) = 0.64$. This is will be equivalent for all states until $s = 87$

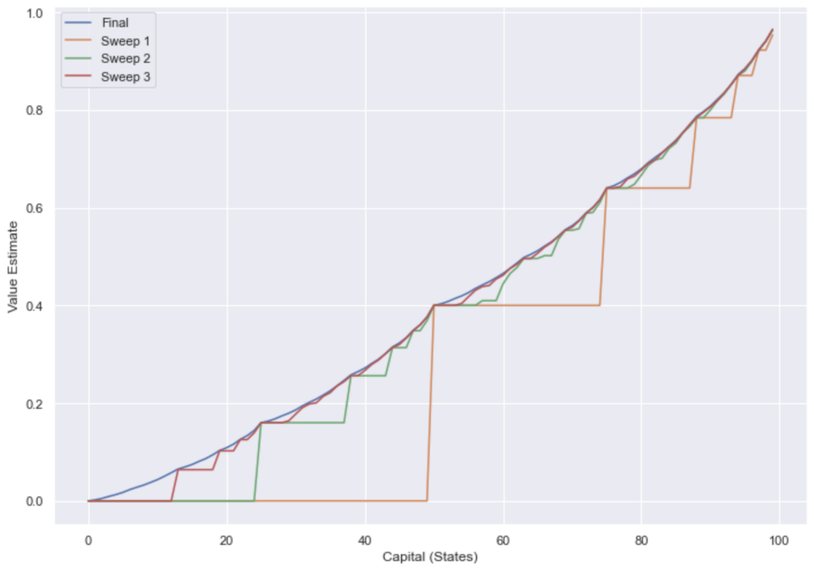

Figure 7: Value function estimate for each state: first, second, third and final estimates.

- for capital $s = 88$ there is a way ($a = 12$) of winning but also to go back to state $s = 76$, so the probability of winning will be $p_h + (1 - p_h)[p_h + p_h(1 - p_h)] = 0.784$
- this will happens similarly for states 94 and 97

In the second sweep, there are distinct value estimates and the iteration will procede in a similar way, getting more accurate estimates of the probabilities. Note that the final estimate does not go below any of the sweeps' values, since previous iterations were always underestimating the value function.

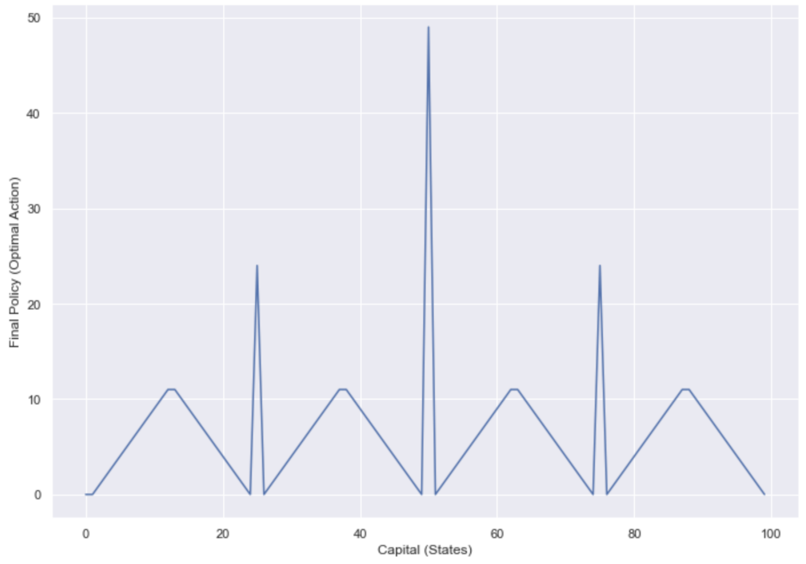

Figure 8: An optimal policy for the estimated value function.

Before looking at the found optimal policy represented in Figure 8 we need to take into account some important factors:

- The problem is numerically unstable and so roundings, namely rounding up values to 4 decimal places, were done to get more clear and interpretable results
- There are multiple optimal policies

- The problem is undiscounted ($\gamma = 1$), hence the number of bets made before winning the game is not taken into consideration. One can check for example that taking action $a = 0$ at all times is an optimal policy
- We do not consider action $a = 0$ as a possibility so as to avoid allowing the agent to get suck in not betting

Remembering the reasoning we saw above regarding the sweeps, we can think about the reported optimal policy and its peaks. With capital 50 ($s = 50$) the probability of winning by gambling 50 is $p_h = 0.4 = \hat{V}(50)$. If we have a capital of 51, if we were to gamble 49, our chances of winning were 0.4 and of loosing 0.6, but by betting 1 we either go back to having 50 with probability 0.6 or go up to 52 where the probability of winning is higher. Since there is no penalization for the number of bets the agent opts for the lower bet. After $s = 51$, we can see the optimal bet increases, so has to always be able to return to $s = 50$. After $s = 63$ it is more worth while to gamble less again to get to $s = 75$ where we have a peak in the probability of winning. Note how the peaks of the policy are aligned with peaks in the estimated value function.

3 Convergence of value iteration

Let:

- $\mathbf{v}_\pi, \mathbf{v^{(k)}} \in \mathbb{R}^{|\mathcal{S}|}$ be the vector forms of the value function and its estimate at iteration k
- $\mathbf{r}_\pi \in \mathbb{R}^{|\mathcal{S}|}$ be the vector of rewards
- $\mathbf{P}_\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ be the matrix of transition probabilities, which is doubly stochastic
- $\gamma \in \mathbb{R}$ and $\gamma < 1$

We can write value iteration as:

$$\mathbf{v}^{(k+1)} = \mathbf{r}_\pi + \gamma \mathbf{P}_\pi \mathbf{v}^{(k)}$$

Let us define operator $T : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ as:

$$T(v) = \mathbf{r}_\pi + \gamma \mathbf{P}_\pi v$$

Let us prove that T is a contraction:

Proof.

$$\begin{aligned} ||T(v_1) - T(v_2)|| &= ||\mathbf{r}_\pi + \gamma \mathbf{P}_\pi v_1 - \mathbf{r}_\pi - \gamma \mathbf{P}_\pi v_2|| \\ &= ||\gamma \mathbf{P}_\pi (v_1 - v_2)|| \\ &\leq |\gamma| \, ||\mathbf{P}_\pi|| \, ||v_1 - v_2|| && \text{Norm is submultiplicative} \\ &\leq |\gamma| ||v_1 - v2||, && ||\mathbf{P}_\pi|| \leq 1 \\ \forall v_1, v_2 \in \mathbb{R}^{|\mathcal{S}|} \end{aligned} \tag{3}$$

□

By Banach's fixed point theorem, we can say T has a unique fixed point for any $v_0 \in \mathbb{R}^{|\mathcal{S}|}$ the sequence defined by the point obtained recursively by:

$$v_{(k+1)} = T v_k, \quad k \geq 0$$

converges to unique fixed point. We know by the Bellman equation for v_π that this is indeed a fixed point of the recursion. Hence, value iteration will converge to v_π regardless of the starting point.

4 The temporal difference operator

Considering the same definitions of the previous section, we are now interested in showing that the operator $T^{(\lambda)} : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ defined by:

$$T^{(\lambda)} v = \sum_{n=0}^{\infty} (\lambda\gamma\mathbf{P}_\pi)^n \left[\mathbf{r}_\pi + \gamma\mathbf{P}_\pi v - v\right] + v$$

is a contraction with respect to the sup-norm, so let $||.||$ denote $||.||_\infty$. With $0 < \lambda < 1$ and $0 \leq \gamma \leq 1$.

Proof. For any $v_1, v_2 \; in \mathbb{R}^{|\mathcal{S}|}$ we have:

$$\begin{aligned}
&||T^{(\lambda)}v_1 - T^{(\lambda)}v_2|| = \\
&\quad = ||\sum_{n=0}^{\infty} \{(\lambda\gamma\mathbf{P}_\pi)^n \left[\mathbf{r}_\pi + \gamma\mathbf{P}_\pi v_1 - v_1\right] - (\lambda\gamma\mathbf{P}_\pi)^n \left[\mathbf{r}_\pi + \gamma\mathbf{P}_\pi v_2 - v_2\right]\} + v_1 - v_2|| \\
&\quad = ||\sum_{n=0}^{\infty} \{(\lambda\gamma\mathbf{P}_\pi)^n \left[\mathbf{r}_\pi - \mathbf{r}_\pi + \gamma\mathbf{P}_\pi (v_1 - v_2) - v_1 + v_2\right]\} + v_1 - v_2|| \\
&\quad = ||(\gamma\mathbf{P}_\pi - \mathrm{I})(v_1 - v_2)\sum_{n=0}^{\infty}(\lambda\gamma\mathbf{P}_\pi)^n + v_1 - v_2|| \\
&\quad = ||\left[(\gamma\mathbf{P}_\pi - \mathrm{I})\sum_{n=0}^{\infty}(\lambda\gamma\mathbf{P}_\pi)^n + \mathrm{I}\right](v_1 - v_2)|| \\
&\quad = ||\left[(\gamma\mathbf{P}_\pi - \mathrm{I})(\mathrm{I} - \lambda\gamma\mathbf{P}_\pi)^{-1} + \mathrm{I}\right](v_1 - v_2)|| && \text{Sum of geometric series of matrices} \\
&\quad = ||\left[(\gamma\mathbf{P}_\pi - \mathrm{I})(\mathrm{I} - \lambda\gamma\mathbf{P}_\pi)^{-1} + (\mathrm{I} - \lambda\gamma\mathbf{P}_\pi)(\mathrm{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}\right](v_1 - v_2)|| \\
&\quad = ||(\gamma\mathbf{P}_\pi - \lambda\gamma\mathbf{P}_\pi)(\mathrm{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}(v_1 - v_2)|| \\
&\quad = ||(\gamma - \lambda\gamma)\mathbf{P}_\pi \sum_{n=0}^{\infty}(\lambda\gamma\mathbf{P}_\pi)^n (v_1 - v_2)|| \\
&\quad \leq ||(\gamma - \lambda\gamma)\mathbf{P}_\pi|| \; ||\sum_{n=0}^{\infty}(\lambda\gamma\mathbf{P}_\pi)^n|| \; ||v_1 - v_2|| && \text{Norm is submultiplicative} \\
&\quad \leq ||\gamma - \lambda\gamma|| \; ||\mathbf{P}_\pi|| \; ||\sum_{n=0}^{\infty}(\lambda\gamma\mathbf{P}_\pi)^n|| \; ||v_1 - v_2|| \\
&\quad \leq ||\gamma - \lambda\gamma|| \; ||\mathbf{P}_\pi|| \; \sum_{n=0}^{\infty} |\lambda|^n \; |\gamma|^n \; ||\mathbf{P}_\pi||^n \; ||v_1 - v_2|| && \text{Triangular inequality} \\
&\quad \leq |\gamma| \; |1 - \lambda| \;\; \sum_{n=0}^{\infty} |\lambda|^n \; ||v_1 - v_2|| && ||\mathbf{P}_\pi|| \leq 1, \; ||\gamma|| \leq 1 \\
&\quad \leq |\gamma| \frac{|1 - \lambda|}{1 - |\lambda|} ||v_1 - v_2|| && \text{Geometric series, } \lambda < 1 \\
&\quad = \gamma ||v_1 - v_2|| && 0 < \lambda < 1
\end{aligned}$$

Hence $T^{(\lambda)}$ is a contraction. □

Note that even though the problem statement requested the proof to be done with sup-norm, we did not use any properties that are not shared with the L2 norm, and hence the proof can also extend to $||.||_2$.