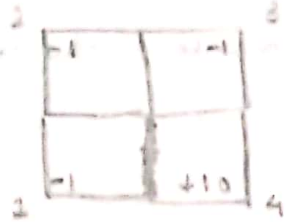
* Monte Carlo Methods.

↓

another gridworld example



↓

state space (4 grid cells)

↓

action space (any among ↑, ↓, ←, →)

↓

$\gamma = 1$

$s_0^1, \uparrow, -1, s_1^2, \leftarrow, -1, s_2^2, \rightarrow, -1, s_3^3, \uparrow, +10, s_4^4$

↓

rewards -1 for NT state, +10 for T state

↓

one step dynamics: the selected action yields the corresponding state with 70% prob, and 30% for other, 10% each for all other actions.

↓

Initially no knowledge of the environment, hence random action to be selected.

↓

hence we follow equi-probable random policy.

↓

let's consider an episode using this policy

↓

$s_0^1, \uparrow, -1, s_1^2, \leftarrow, -1, s_2^2, \rightarrow, -1, s_3^3, \uparrow, +10, s_4^4$

↓

i) to be able to carve an optimal policy, requires knowledge about a lot of episodes, so that all permutations of states and actions are tried out multiple times.

↓

more experience → more knowledge. (or more learning)

ii) for each state, figure out the best action (in terms of maximum ~~cu~~ cummulative reward received)

The two mentioned points are the basic gist of monte carlo approach.

↓

how do we consolidate the data to choose the apt actions to form a policy.

↓

done by maintaining a table for nontemporal states vs actions, where the value for that particular entry is the reward earned for the state, if that particular action is taken. This table is known as Q table

	↑	↓	←	→
1	+7	+6	+5	+6
2	+8	+7	+8	+9
3	+10	+8	+9	+9

Q table.

↓

avg. of the state, action pair over episodes.

← if state, action pair multiple times, over an episode?

↓ First occurrence (first visit MC)

↓ avg. of all occurrence (every visit MC)

↓

Based on 100s or 1000s of episodes, lots of data has been collected, which will improve the decision making for the agent.

↓

once lot of data is collected, action with max. reward value for each state can be chosen

↓

this might yield a better or equivalent policy. *'

↓

it may or maynot be the optimal policy

↓

but it may be one of the some small steps towards the optimal policy.

* The Q table allows us to estimate the action value function for the equi-probable random (here) policy i.e. it allows us to estimate the expected return if the agent states in the particular state and picks the corresponding cell action, following the same policy π., which can be used to find a better policy.

↓

The problem of estimating the value function given a policy is called the prediction problem.

↓

Monte Carlo methods applied for this are called monte carlo approaches.

Monte carlo Approach is basically, when the agent has a policy in mind, it follows the policy to collect a lot of episodes. Then for each state, to figure out which action is best, the agent can look for which action, the max. cummulative reward was generated.

based on the way multiple occurrences in an episode are handled, we have two approaches, i.e.

a) first occurrence (first visit MC prediction)

b) avg. of all occurrence. (every visit MC prediction)

→ policy → collect a lot of episodes

↓

from this data, choose best action for each state.

* Both FV and EV method are guaranteed to converge to the action-value function, as the no. of visits approach ∞.

↓

FV MC convergence follows law of large numbers.

↓

* EVMC is biased, whereas FVMC is unbiased.
* Initially EVMC has lower MSE, but as exp ↑, FVMC attains better MSE.

* Greedy & epsilon greedy policies.

↓

if while moving to a better policy from the Q table of the current policy, by deciding the best possible action on the basis of the cumulative reward generated from that action, we call this approach greedy policies.

↓

general process → moving from a bad policy to an optimal policy will be the continuous process of estimating the Q table, and using that to find a new policy

↓

collect episodes with π to estimate a Q table

$\pi' \leftarrow \text{greedy}(Q)$

$\pi \leftarrow \pi'$

↓

the greedy approach might not be the best ~~po~~ approach to estimate the optimal policy. (exploration.-exploitation dilemma)

↓

Here, there is no scope for further exploration, & the agent keeps exploiting the already garnered knowledge and that increases the importance of the initial episodes very much, and hence it might move into ~~an~~ non-optimum policy.

An example of the same, can be the 2 door problem, where we have two doors, opening any one yields some reward, and that is also the terminal step.

↓

opening door A gives +1, +3 reward with equal probability.

↓

whereas opening door B, gives 0, +100 reward with equal probability.

↓

Now, if initially we open door B, after choosing randomly and we get reward 0, then we decide the next randomly, and this time we open door A and get reward +1. Now since door A yielded higher return, the policy would get modified to open only A all the time as per greedy policy, which is not optimal

↓

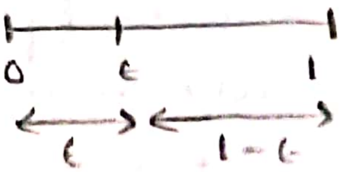
Hence we need to add a stochastic element to this decision making process where the greedy action gets chosen with high probability, but other action also has some probability of being chosen, which adds the element of exploration in the policy estimation process.

↓

This factor is ϵ, and its value lies b/w 0, 1, and gives the probability of choosing any of the actions. (includes the greedy action), whereas $1-\epsilon$ for the greedy action

↓

This process is the epsilon greedy policy.



(30)

* Mathematically; in order to construct a policy π that is ϵ-greedy with respect to the current action-value function, we set :

↓

chosen because greedy policy

$$\pi(a|s) \rightarrow 1-\epsilon+\frac{\epsilon}{|A(s)|}$$

random choosing factor

for a which maximises over $Q(s,a)$

&

$\frac{\epsilon}{|A(s)|}$ for all other actions in $A(s)$

control problem
↓
estimating optimal policy.

↓

sum of the above should be 1,

$$(|A(s)|-1)\left(\frac{\epsilon}{|A(s)|}\right)+1-\epsilon+\frac{\epsilon}{|A(s)|}$$

$$\cancel{\epsilon}-\cancel{\frac{\epsilon}{|A(s)|}}+1-\cancel{\epsilon}+\cancel{\frac{\epsilon}{|A(s)|}}=1$$

↓

estimating Q table for π (policy evaluation)

$\pi' \leftarrow \epsilon\text{-greedy}(Q)$
$\pi \leftarrow \pi'$
(policy improvement)

↓

This approach of alternating b/w policy evaluation & policy improvement, is used to solve the control problem and is referred to the Monte Carlo control method

* Being able to converge to an optimal policy as quickly as possible is very important & for the agent to be able to do this, it should be able to balance exploration & exploitation quite well.

↓

the exploration factor induced by the introduction of ϵ greedy policies.

↓

These requirements can be balanced by the gradually modifying the value of ϵ when constructing ϵ-greedy policies.

* At initial time steps, the agent should favour exploration over exploitation, to be able to collect, or consolidate more knowledge (nearer to 1), and as time steps grow, at later steps, the agent should favour exploitation over exploration (nearer to 0) to be able to maximise reward.

↓

Theoretically setting ϵ, done using greedy in the limit with ∞ exploration (GLIE)

↓

In order to guarantee that MC control converges to the optimal policy π_*, two conditions need to be met:

a) every (s, a) visited ∞ many times.

b) policy converges to a policy that is greedy wrt to the Q table of π

↓

These conditions ensure that: a) agent explores for all time steps.

b) agent gradually exploits more (& explore less)

(3.2) These can theoretically be realised by modifying ϵ as time ↑.

↓

Let ϵ_i correspond to ϵ at ith time step.

both these conditions are met if $\epsilon_i > 0$ for all time steps i, & ϵ_i decays to zero in the limit as time step i approaches ∞

↓

this can be done by setting $\epsilon = \frac{1}{i}$

* setting ϵ_i practically

↓

Setting ϵ_i in accordance with the GLIE may not always be good enough, since it might take million or billion episodes for the policy to converge

↓

Letting ϵ to get too small might lead to later episodes without optimal policy converge to operate very slowly.

↓

So practically, one of the following approach is used;

i) using fixed ϵ

ii) letting ϵ_i decay to a small positive number, like 0.1

↓

in paper explaining DQN, ϵ-greedy was used with ~~epsil~~ epsilon annealed linearly from 1 to 0.1 over the first million frames, and fixed at 0.1 thereafter

* Incremental Mean.

↓

updating Q table after each episode and using that improved Q table for the improved policy. This can be used to estimate action values efficiently after each episode

↓

(discounted return)

$$Q_{n+1} \leftarrow Q_n + \frac{1}{N}\left(G_{n+1} - Q_n\right) \quad \ldots (i)$$

After each episode, new action value estimate is found from : i) old action value estimate (Q_n)

ii) most recently sampled return (G_{n+1})

iii) total number of visits (N) to the state, action (s, a) pair.

(refer to 5.6 of the textbook)

↓

$Q_2 \leftarrow Q_1 + \frac{1}{2}(G_2 - Q_1)$

$\leftarrow 2 + \frac{1}{2}(6)$

$\leftarrow 5$

N	1	2	3	4
G	2	8	11	3
Q	2	5	7	6

$Q_4 = Q_3 + \frac{1}{4}(3 - 7)$

$= 7 + \frac{1}{4}(-4)$

$= 6.$

* Constant α

↓

corresponding return for state, action pair

In (i) $\frac{1}{N}\left(\underbrace{G_{n+1} - Q_n}\right)$

error term → measure of its deviation from the expected estimate.

↓

$\delta(t)$

↓

if $\delta(t) > 0$ → our estimate is less than what was expected, and needs to be inc, viceversa for $\delta(t) < 0$.

← this change, both +ve and -ve is proportional inversely to N.

34

Now since the dependency is on N, then $\frac{1}{N}$ will yield large changes initially, and very small changes in future time steps

↓

$\frac{1}{N}$ replaced by α step size hyper parameter

↓

Now returns that come later are more emphasized, as the agent will trust the more recent episodes, since policy is being improved at each time step, and they should be able to give them more weightage, than episodes that came previously.

$$Q_{t+1} \leftarrow Q_t + \alpha(G_{t+1} - Q_t)$$

$$Q_{t+1} \leftarrow Q_t(1-\alpha) + \alpha G_{t+1} \quad \text{(ii)}$$

* setting the value of α

↓

i) needs to be between 0 & 1

0 → disregard the current estimate and take older estimate into consideration → agent would never learn

1 → only recent return taken into consideration → agent would never learn

(ii) From (ii), smaller values of α encourage the agent to consider longer history of returns when calculating the action-value function estimate.