Reinforcement Learning Course Project

Partha Sarathi Mohapatra
EE18D703

13th March 2020

Policy Gradients

To parameterize the policy with parameter θ, we choose multivariate Gaussian distribution with identity covariance matrix with mean $\mu = \theta^T(s_1, s_2, 1)^T$. So, $\log\pi_\theta(a|s) = -\log 2\pi - (a-\mu)^T(a-\mu)$ and the gradient of the policy is given as:

$$\begin{aligned}\nabla_\theta \log\pi_\theta(a|s) &= -\nabla_\theta \left((a-\mu)^T(a-\mu)\right) \\ &= a^T(s_1, s_2, 1)^T - \theta(s_1, s_2, 1)^T(s_1, s_2, 1)\end{aligned}$$

We used this above equation to calculate the gradient for updating the θ values in each iteration.

1 Answers and Observations

1.1

For tuning the hyper-parameters like batch size(N), learning rate (α_θ and α_w), discount factor (γ), maximum iterations, we tried different combinations (within suitable range of each parameters) and conclude the followings:

(i) Increasing the batch size (N), accelerate the learning process, but the program running time increases. Similarly we have to limit the maximum iterations for getting good accuracy with reasonable program running time. From many trial and error we fixed the batch size, $N = 500$ and the maximum iteration as 100 (after 50 iterations the results are also very good).

(ii) If the learning rate (α_θ) for the update of the parameter (θ) of the policy is made large, then it will increase the oscillation around the sub-optimal solution (trajectories will have oscillation near the center). On the other hand decreasing α_θ, makes the convergence sluggish. Considering the above dilemma we try to get an itermediate value for α_θ for optimal performance and fixed its value as 0.0005.

(iii) We tuned the discount factor (γ) as 0.9 and the learning rate (α_w) for the value function (a rough estimate used for baseline calculation) as 0.001.
So the hyper-parameters values are chosen as follows:

$$\begin{aligned}N &= 500 \\ \text{maximum iteration} &= 100 \\ \alpha_\theta &= 0.0005 \\ \alpha_w &= 0.001 \\ \gamma &= 0.9\end{aligned}$$

Some observations and effects of different parameters during tuning are discussed above. Another important thing we observe was regarding baseline which is described as follows:

(iv) If we choose the baseline as average reward rather than 0, the learning process become faster. The learning process can even be made more efficient by using a rough value estimate of the states.

(v) Yes, we can make the policy learning process faster by introducing a rough parameterized value estimate, $v(S, w)$ (with w as parameter) as baseline for this algorithm.

(vi) As the reward is proportional to the distance of the state from the origin (skewed distance for VishamC), we can take square of the x-coordinate and y-coordinate as the basis function (square of the states). This will also take advantage of the symmetry of the reward values. So we choose the baseline as follows:

$$v(S, w) = (w_1, w_2, w_3)(s_1^2, s_2^2, 1)^T$$
$$\nabla_w v(S, w) = (s_1^2, s_2^2, 1)^T$$

1.2

Chakra world:

(i) For the chakra world as the rewards are negative of the distance from the origin, the rough value function for the learned policy must have circular contour with the value decreasing (becoming more and more negative) as the distance from the origin increases.

(ii) All the points equi-distance from the origin must have same value for the value function.

(iii) The highest value attained by the value function is zero and is at the center.

VishamC world:

(i) In case of vishamC the reward function is skewed and is represented by elliptical equation i.e by $-(0.5x^2 + 10 \times 0.5y^2)$, so the resulting value function must also have elliptical contours.

(ii) All the states, for which $(0.5x^2 + 10 \times 0.5y^2)$ is same, must have same value for the value function and the value function must decrease from the center according to this equation.

(iii) The highest value of the value function is zero and is at the center.

The plots of the value functions for both chakra and vishamC world are shown below:

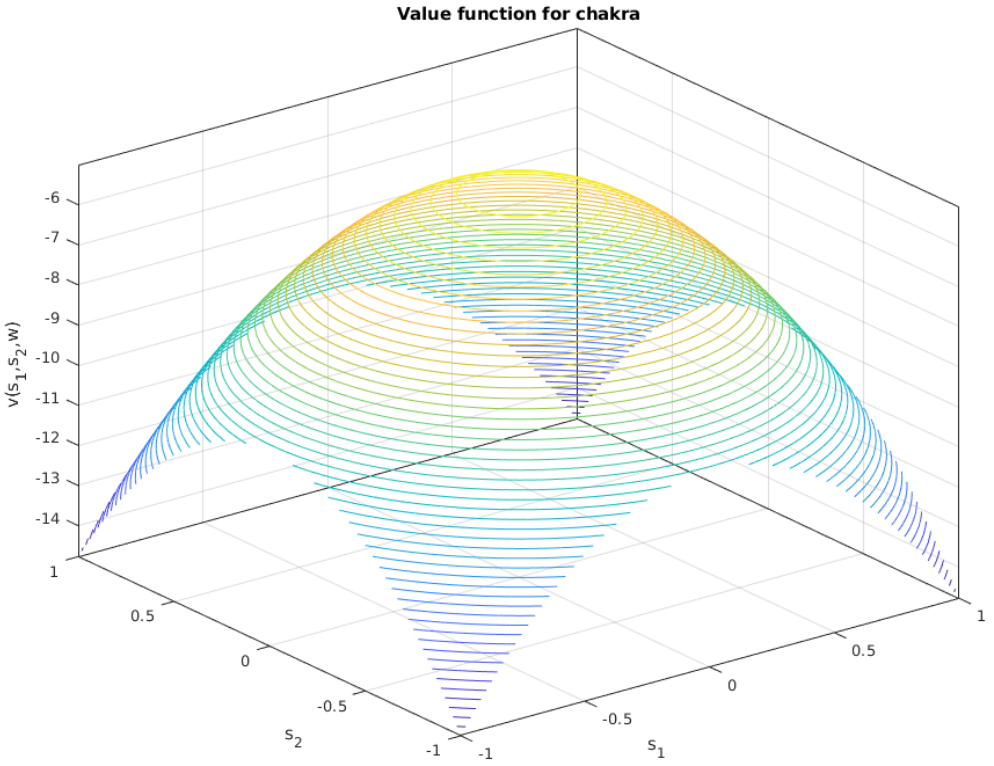

Figure 1: Value function for chakra world

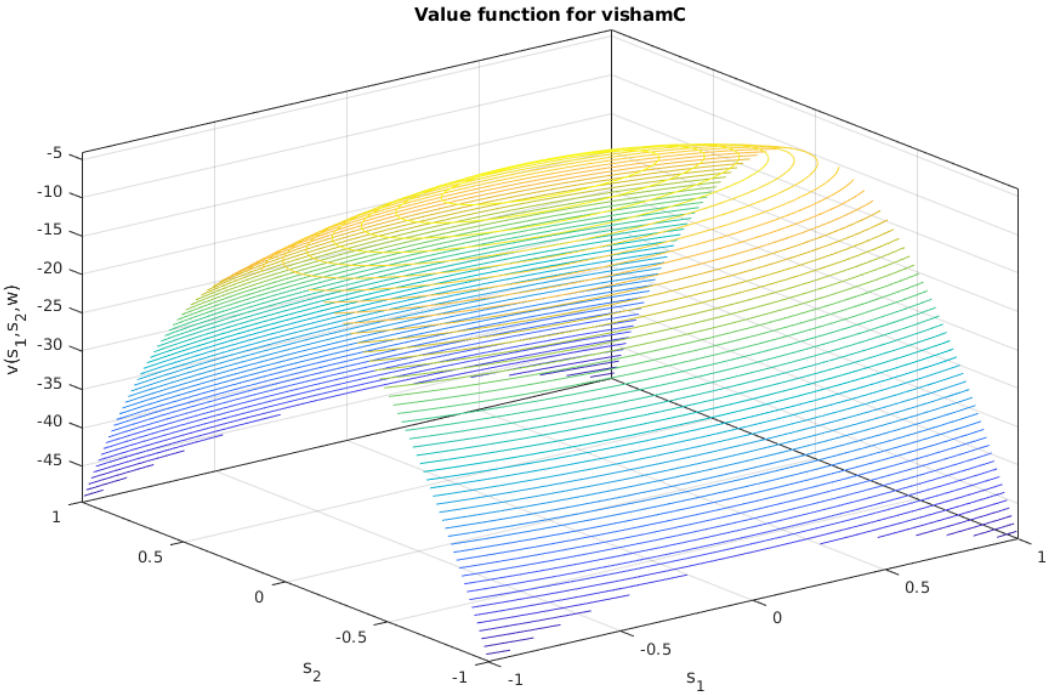

Figure 2: Value function for vishamC world

1.3

(i) Yes, there is a pattern in how the agent reach the origin in both cases. The policy trajectories for the chakra world are radial as the increase in gradient to the center is same in both state directions.

(ii) For the case of vishamC world the increase in gradient to the center is highest in y-axis (for state s_2) as compared to x-axis (state s_1), so the the optimal trajectories are skewed as shown in the figure below.

We used the converged values of the θ (after 100 iterations) as given below for getting the trajectories for both cases. Below are a rough representation of the optimal trajectories.

$$\theta_{\text{chakra}} = \begin{pmatrix} -17.50489398 & -0.0934462 & -0.07491875 \\ -0.1526158 & -17.66515993 & 0.03056671 \end{pmatrix}$$

$$\theta_{\text{vishamC}} = \begin{pmatrix} -5.19339945e+00 & 3.43873506e-02 & -1.36560944e-01 \\ -4.17138345e-02 & -2.43173858e+01 & -1.24982045e-02 \end{pmatrix}$$

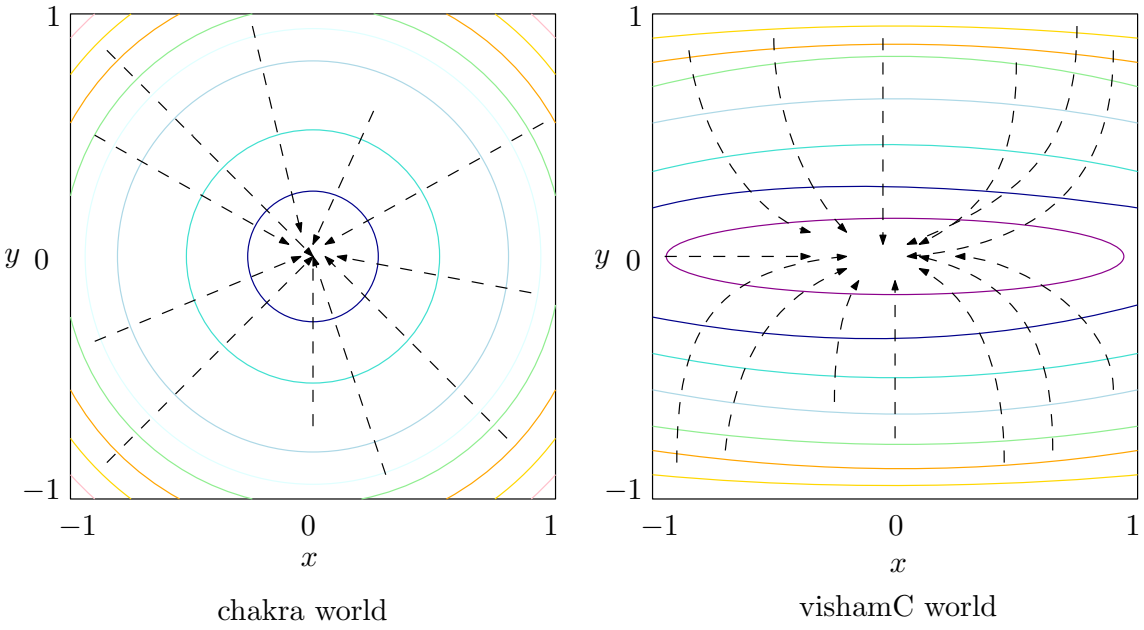

Figure 3: Optimal policy trajectories for chakra and vishamC world