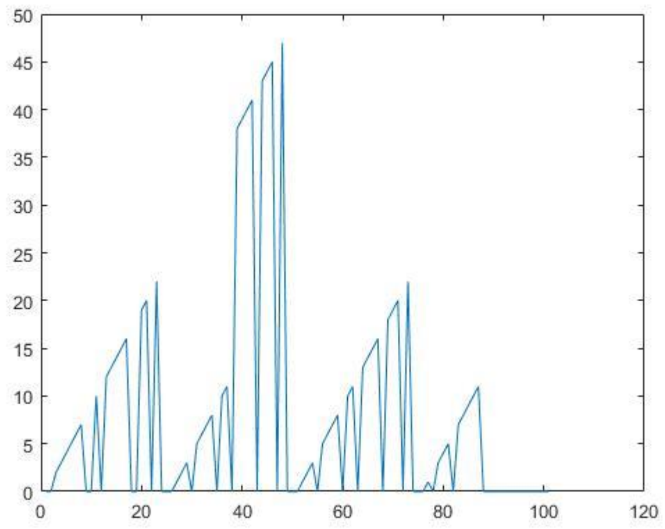
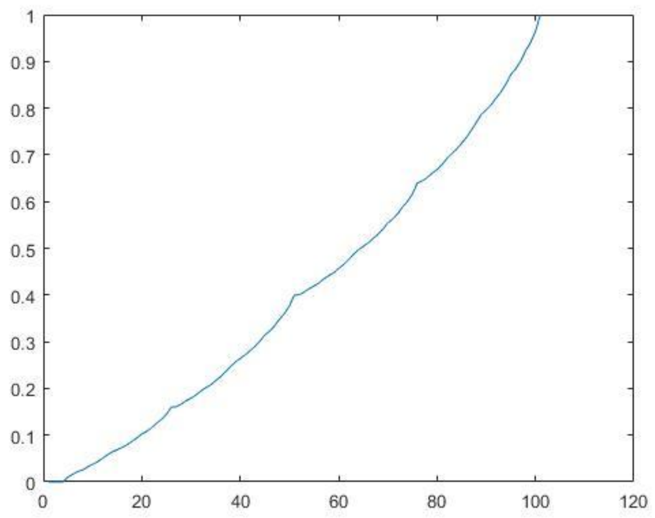
# Assignment 2

5) Here are the plots of state values and policy when I include 0 as an action. I run the code GamblersProblem_with_zero.cpp, which will generate final values and policy on the console. I copy them and make a MATLAB vector using v = [<--copy paste from console-->]; plot(v). Similarly for policy plot.

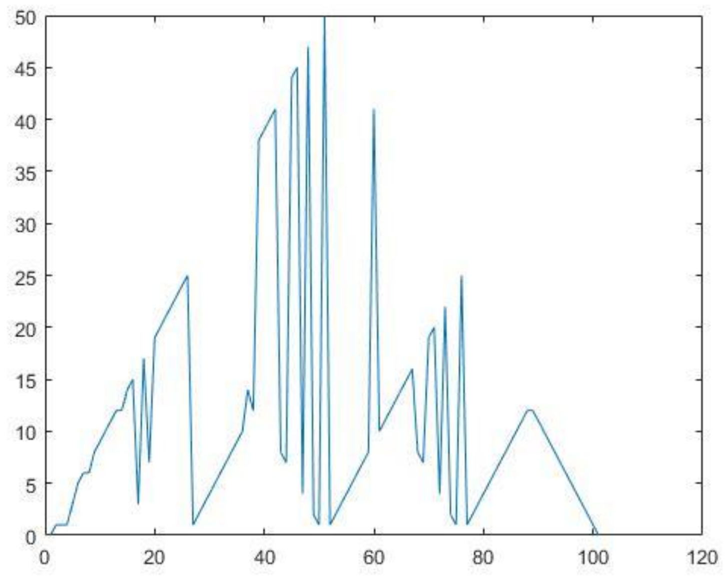


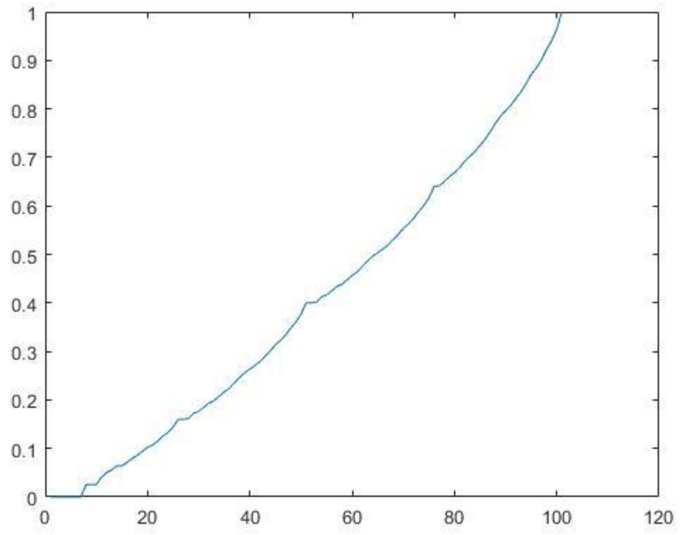
Policy plot



Value plot

When I exclude 0 as an action I get these plots. The code for these is GamblersProblem_without_zero.cpp.

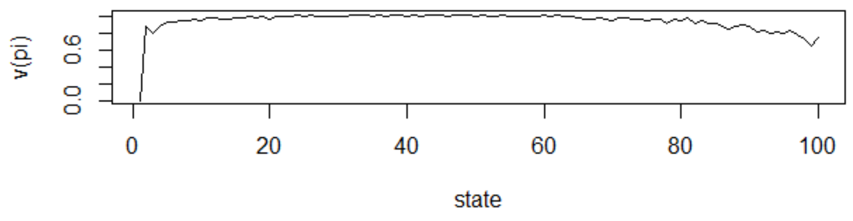


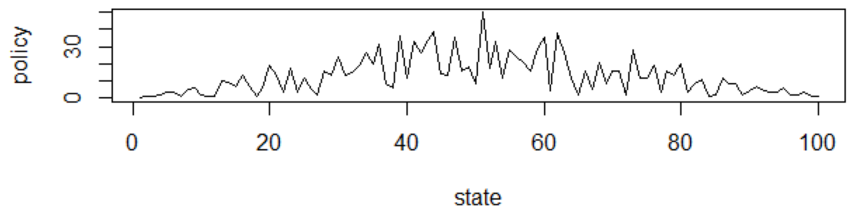


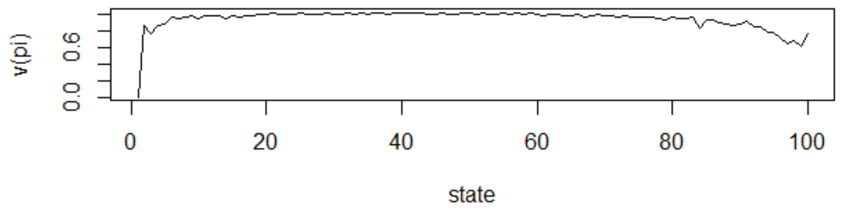
The plots for policy is inconsistent with the one presented in the book. This is because I used *'max_element'* function to select the policy with maximum action value. When there are 2 maximum values in an array this function returns the first occurrence of the maximum value. So my code always selects the action with minimal action value as optimal action. This is why the plot I got is different from the book which must have used a better tie breaking mechanism.

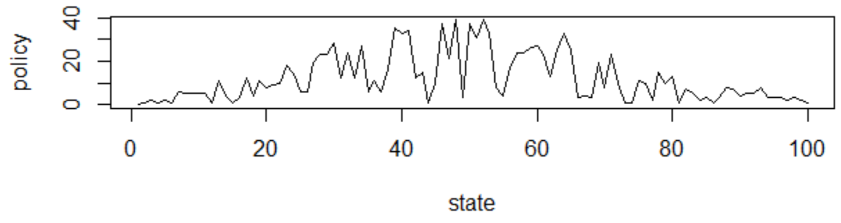
9) The code can just be run with make and make run. It returns v_pi.dat and policy.dat which are used to plot from R using plot.r file.

Here are the plots for 10000 and 1000000 episodes averaged over 30 steps.









For the gamblers problem Monte Carlo is not a suitable method compared to the dynamic programming which works perfectly. Some of the hurdles while implementing Monte Carlo method are finding a good initial policy. The final value plot will change depending on the initial policy we take. The plots I generated are for policy with uniform random probability.

Other problem with the method is that we have to run the program for many number of episodes to get good estimates. While in case of dynamic programming we can do it in much simpler way.

If the experiment has more probability then 0.25 the value function reaches 1 sooner compared to when probability is less than 0.25.

6) The estimated value function jump up for the last two rows in the rear as these ones correspond to player having sums of 20 and 21. As these hand values are large enough to guarantee a win to the player and hence he gets a reward of 1.

The drop on the last row on the left is because these rows correspond to the dealer showing an ace. Hence the dealer has a high probability of winning which would lead to -1 reward.

The foremost values are higher in case where the player has usable ace. This is because when the player has an ace he knows that the probability of dealer having an ace drops a little since there are only 3 more aces left.

1)

$$q_\pi(s,a) = \sum_{s'} r(s,a,s') \times P(s,a,s') + \gamma \sum_{s'} P(s,a,s')\, v_\pi(s')$$

① $r(s,a,s') = -1$ for all states

for terminal states $s^*$ $\quad r(s,a,s^*) = 0$

(reward can be assumed as 0).

② Given that ~~Assume~~ actions are deterministic.

③ I will use policy iteration values given in book.

④ Terminal state values taken zero.

$$q_\pi(4, up) = r(4, up, s^*) \times 1 + \gamma \times 1 \times v_\pi(s^*).$$

(1 because deterministic).

$$= 0 \times 1 + \gamma \times 1 \times 0 = 0.$$

(0 because we assumed reward for entering terminal state = 0)

(0 because value of terminal state is 0).

$$q_\pi(\overset{5}{~~4~~}, \overset{left}{~~up~~}) = r(5, left, 4) \times 1 + \gamma \times 1 \times v_\pi(4)$$

$$= -1 + -14 = -15.$$

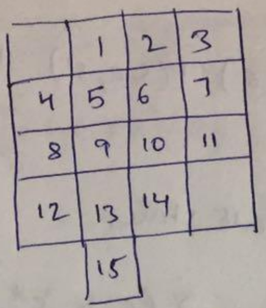
(Assuming discount rate $\gamma = 1$).

⑤

If $\gamma = 0.9$

$$q_\pi(5, left) = -1 + -14 \times 0.9 = -13.6$$

2)



I will use original state values for grid assuming that they won't change after adding (15).

$$v_\pi(15) = \sum_a \pi(15,a) \sum_{s'} p(15,a,s')\left(r(15,a,s') + \gamma V(s')\right)$$

Given $\pi(15, a) = 1/4$ $\quad a =$ up, down, left, right

~~&~~ ~~π(15, up)~~ $p(15, up, 13) = 1$ $\quad$ & $r = -1$ for all transitions

$p(15, left, 12) = 1$

$p(15, down, 15) = 1$

$p(15, right, 14) = 1.$

$$v_\pi(15) = \tfrac{1}{4}(-1-22\gamma) + \tfrac{1}{4}(-1-20\gamma) + \tfrac{1}{4}(-1-14\gamma) + \tfrac{1}{4}(-1+\gamma V(15)).$$

$$= -1 - 12.6 + \frac{0.9}{4}V(15) \quad \text{(taking } \gamma = 0.9\text{)}$$

$$V(15) = \frac{4}{3.1} \times -13.6 = -17.54839$$

If we assume dynamics of 13 has changed

$$v_\pi(13) = \tfrac{1}{4}\Big[-1 + \gamma v^\pi(9) + \gamma v^\pi(15) - 1 - 1 + \gamma v^\pi(14) - 1 + \gamma v_\pi(12)\Big]$$

$$= \tfrac{1}{4}\big(-4 + \gamma v_\pi(9) + \gamma v_\pi(15) + \gamma v_\pi(12) + \gamma v_\pi(14)\big)$$

$$= -1 + \frac{\gamma}{4}\big(-20 - 14 - 22 + v_\pi(15)\big) = -1 + \frac{\gamma}{4} v_\pi(15) - 14\gamma \quad \text{— } \textcircled{1}$$

Same for $v_\pi(15)$

$$v_\pi(15) = -1 + \frac{\gamma}{4}\big(v_\pi(12) + v_\pi(13) + v_\pi(15) + v_\pi(14)\big)$$

$$= -1 + \frac{\gamma}{4}\big(v_\pi(13) + v_\pi(15) - 14 - 22\big)$$

$$= -1 + \frac{\gamma}{4}\big(v_\pi(13) + v_\pi(15)\big) - 9\gamma \quad \text{— } \textcircled{1}$$

taking $\gamma = 0.9$

Let $v_\pi(13) = a$
$v_\pi(15) = b$

from ① $a = -1 + \frac{0.9}{4} b - 14 \times 0.9$

$a - 0.215b = -13.6$

from ② $b = -1 + 0.215(a+b) - 8.1$

$0.215a - 0.785b = 8.1$

by solving them

$v_\pi(13) = -17.3$ $v_\pi(15) = -16.7$

3) policy iteration for action values $q(s,a)$.

```
1. Initialization
      $q(s,a)$ and $\pi(s) \in A(s)$ arbitrarily for all $s \in S$, $a \in A(s)$

2. Policy Evaluation
      Repeat ~~until~~ [illegible]
         $\Delta \leftarrow 0$
         for every $(s,a)$ pair
            $q \leftarrow q(s,a)$
            $q(s,a) \leftarrow \sum_{s'} \overset{\text{(reward)}}{r}(s,a,s')\, P(s,a,s') +$
                $\overset{\text{gamma}}{\gamma} \sum_{s'a'} q_\pi(s',a')\, \pi(s',a')\, P(s,a,s')$
            $\Delta \leftarrow \max(\Delta, |q - q(s,a)|)$
      until $\Delta < \theta$ (a small positive number).

3. Policy Improvement
      policy stable $\leftarrow$ true.
      For each ~~$s \in S$~~ s,a ~~action~~ pairs.
         $p \leftarrow \pi(s,a)$
         $\pi(s,a) \leftarrow \arg\max_a q_\pi(s,a)$
         if $p \neq \pi(s,a)$, then policy stable $\leftarrow$ false
      if policy stable ~~[illegible]~~, then stop and return $\pi(s,a) \approx \pi_*$
                                                          $q(s,a) \approx q_*$
      else go to step 2.
```

4)

~~$q_{k+1}(s,a) = \max_a E\left[ r(s,a,s') + \gamma \sum_{s' \in S} p(s,a,s') v_k(s') \right]$~~

$$q_{k+1}(s,a) = \max_a E\left[ \overset{R_{t+1}}{\cancel{R_{k+1}}} + \gamma q_k(S_{t+1}, a_{t+1}) \mid S_t = s, A_t = a \right]$$

$$= \cancel{\max_a \sum_{s' \in S} r(s,a,s')\, p(s,a,s') +}$$

$$\cancel{\gamma \sum_{s',a'} q_\pi(s',a')\, \pi(s',a')\, p(s,a,s')}).$$

$$= \max_a \sum_{s' \in S} p(s,a,s') \left[ r(s,a,s') + \gamma q_k(s',a') \right].$$

9)

$$v_n = \frac{\sum_{k=1}^{n} w_k R_k}{\sum_{k=1}^{n} w_k} \quad - (5\text{-}6)$$

$$v_{n+1} = v_n + \frac{W_n}{C_n}\left[G_n - V_n\right], \quad n \geq 1 \quad (5\text{-}7)$$

$$v_{n+1} = \frac{\sum_{k=1}^{n+1} w_k R_k}{\sum_{k=1}^{n+1} w_k} = \frac{\sum_{k=1}^{n} w_k R_k + w_{n+1} R_{n+1}}{\sum_{k=1}^{n+1} w_k}.$$

$$= \frac{\cancel{\left(\sum_{k=1}^{n} w_k\right)}}{\cancel{\left(\sum_{k=1}^{n} w_k\right)}} \times \frac{\sum_{k=1}^{n} w_k R_k + w_{n+1} R_{n+1}}{\sum_{k=1}^{n+1} w_k}$$

$$= \frac{\sum_{k=1}^{n} w_k}{\left(\sum_{k=1}^{n+1} w_k\right)} \times \overbrace{\left(\frac{\sum_{k=1}^{n} w_k R_k}{\sum_{k=1}^{n} w_k}\right)}^{(V_n)} + \frac{w_{n+1} R_{n+1}}{\left(\sum_{k=1}^{n+1} w_k\right)}.$$

take $W_n = \sum_{k=1}^{n} w_k$.

$$\Rightarrow V_{n+1} = \frac{W_n}{W_{n+1}} \times V_n + \frac{w_{n+1}}{W_{n+1}} R_{n+1}$$

we have.

$$\frac{W_n}{W_{n+1}} = \frac{W_{n+1} - w_{n+1}}{W_{n+1}} = 1 - \frac{w_{n+1}}{W_{n+1}}.$$

$$V_{n+1} = V_n - \frac{w_{n+1}}{W_{n+1}} V_n + \frac{w_{n+1}}{W_{n+1}} R_{n+1}$$

$$= V_n - \frac{w_{n+1}}{W_{n+1}} \left(R_{n+1} - V_n\right).$$

Hence proved.

8)

$$V_{k+1} = \frac{1}{k+1} \sum_{i=1}^{k+1} G_i$$

$$= \frac{1}{k+1} \left( G_{k+1} + \sum_{i=1}^{k} G_i \right)$$

$$V_k = \frac{\sum_{i=1}^{k} G_i}{k}$$

$$\Rightarrow \sum_{i=1}^{k} G_i = kV_k$$

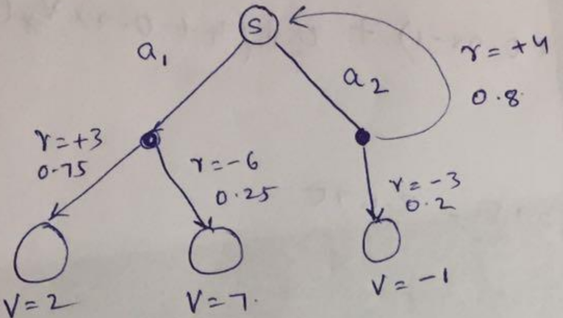
$$= \frac{1}{k+1} \left( G_{k+1} + kV_k \right)$$

$$= \frac{1}{k+1} \left( G_{k+1} + (k+1)V_k - V_k \right)$$

$$V_{k+1} = V_k + \frac{1}{k+1} \left( G_{k+1} - V_k \right)$$

Here $V_{k+1}$ is value after $k+1$ episode
and $G_k$ is returns calculated in that episode.

10)



$\gamma = 0.9$

first if we take $a_1$ and $a_2$ as equiprobable $\Rightarrow$ $\pi(a_1/s) = 0.5$, $\pi(a_2/s) = 0.5$

$$V_\pi(s) = \sum_{a \in A} \pi(a/s)\left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s')\right)$$

$$V(s) = 0.5\left(0.75\underbrace{(+3+0.9\times2)}_{3.6} + 0.25\underbrace{(-6+0.7\times7)}_{0.075}\right) +$$
$$0.5\left(0.2(-3+0.9\times-1) + 0.8(4+0.9V(s))\right)$$

$$2V(s) = 3.6 + 0.075 - 0.68 + 3.2 + 0.72V(s)$$

$$1.28V(s) = 6.175$$

$$\Rightarrow V(s) = 4.839844$$

for optimal policy * we choose either $a_1$ or $a_2$ . based on which leads to maximum reward.

$$V_*(s) = \max_a R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_*(s')$$

$$\text{or} \quad V_*(s) = \max_a q_*(s,a)$$

$$V_*(s) = \max \begin{cases} 0.75(+3 + 0.9\times 2) + 0.25(-6 + 0.7\times 7) & -① \\ 0.2(-3 + 0.9\times -1) + 0.8\left(4 + 0.9\times V_*(s)\right) & -② \end{cases}$$

by solving ①

$$V_*(s) = 3.6 + 0.075 = 3.675$$

by solving ②

$$V_*(s) = -0.68 + 3.2 + 0.72\, V_*(s)$$

$$\Rightarrow 0.28\, V_*(s) = 2.52$$

$$\Rightarrow V_*(s) = 9.$$

We ~~tod~~ get optimal by choosing ②.

so, $V_*(s) = 9$.