Value function of a state following a stochastic policy :-

$$V^{\pi}(s) = \underset{\tau \sim \pi}{E}\left[R(\tau) \mid s_0 = s\right]$$

$$\therefore V^{\pi}(A) = \underset{\tau \sim \pi}{E}\left[R(\tau) \mid s_0 = A\right]$$

$$= \sum_i \underbrace{R(\tau_i)}_{\text{Return of each trajectory } \tau_i} \pi(a_i \mid A)$$

gives the prob of each action a_i in state A

$a_i = \{down, right\}$

$\tau_i = \{\tau_1, \tau_2\}$

$$= R(\tau_1)\,\pi(down \mid A) + R(\tau_2)\,\pi(right \mid A)$$

$$= 4(0.8) + 2(0.2) = 3.6.$$

Ex for value function (Pg: 33)

Assume we have 2 policies π_1 and π_2. Let $V(s)$ value of state s, using policy π_1 is $V^{\pi_1}(s) = 13$, $V^{\pi_2}(s) = 11$. $\therefore$ Optimal value of s = 13. The policy that gives the optimal value for a state s is the optimal policy π^*. Hence π_1 is the optimal policy.

If we have the value table of two such states giving us the max (optimal) value in each of them

State	Value
s_0	7
s_1	11

$\therefore$ Its better to be in s_1. $\because$ s_0 has less state value.

$\therefore$ s_1 is the optimal state.

$\therefore$ we can find the optimal state from a value table.

Q function:- $Q^{\pi}(s,a) = [R(\tau) | s_0 = s, a_0 = a]$

In value fn, we find the value of a state,
" Q fn, we find the value of a state-action pair.

Ex: Pg: 34: $Q^{\pi}(A, down) = [R(T) | s_0 = A, a_0 = down]$

$= 1+1+1+1 = 4.$

$Q^{\pi}(D, right) = [R(T) | s_0 = D, a_0 = right]$

$= 1+1+1 = 3.$

Expected Q function:-

$$Q^{\pi}(s,a) = \underset{\tau \sim \pi}{E} [R(\tau) | s_0 = s, a_0 = a]$$

Q fn depends on the policy. $\therefore$ there will be different Q-values for a (s,a) pair, depending on the policy. $\therefore$ The optimal Q fn for a (S,a) pair is the one that gives the max (optimal) Q-fn value among all.

$$\therefore \quad Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

$\therefore$ The optimal policy π, for a (S,a) is the one that gives the max. Q-fn.

- Q fns can be visualized using a Q-table.

Ex: find the optimal policy π, for the given Q-table.

State	action	Value
s_0	0	9
s_0	1	11
s_1	0	17
s_1	1	13

$\longrightarrow$ optimal policy

State	Action
s_0	1
s_1	0

$\therefore$ $\boxed{\text{Recap}}$:- $V^{\pi}(s) = [R(\tau) \mid s_0 = s]$; det. policy values

for stocastic policy

$$V^{\pi}(s) = \mathop{E}_{\tau \sim \pi}[R(\tau) \mid s_0 = s]$$

$$= \sum_i R(\tau_i)\, \pi(a_i, s)$$

$$V^*(s) = \max_{\pi} V^{\pi}(s) \quad \cdots \left\{ \begin{array}{l} \text{optimal value of} \\ \text{a state } s \end{array} \right.$$

given value table of each state, we can find the optimal state.

Q - function.

$$Q^{\pi}(s,a) = [R(\tau) \mid s_0 = s, a_0 = a]$$

$$Q^{\pi}(s,a) = \underset{\tau \sim \pi}{E} [R(\tau) \mid s_0 = s, a_0 = a]$$

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

Given Q-table, we can find the optimal policy.

Bellman Eqn of the value fn of a state for a deterministic envt:-

$$V^{\pi}(s) = R(s,a,s') + \gamma V(s')$$

for a stochastic envt. - Each state has different next states for an action a'

$$V^{\pi}(s) = \sum_{s'} \underset{\substack{\Downarrow \\ \text{trans. prob} \\ \text{of next state}}}{P(s'|s,a)} \underbrace{R(s,a,s') + \gamma V(s')}_{\text{Backup}}$$

Pg: 4

$$V(s_1) = \left[P(s_2|s_1,a_1) R(s_1, a_1, s_2) + \gamma V(s_2)\right] +$$

$$\left[P(s_3|s_1,a_1) \quad P(s_1, a_1, s_3) + \gamma V(s_3)\right]$$

$$= 0.7 \quad R(s_1, a_1, s_2) + \gamma V(s_2) +$$

$$0.3 \quad R(s_1, a_1, s_3) + \gamma V(s_3)$$

$$\therefore \boxed{V^{\pi}(s) = \sum_{s'} P(s'|s,a) \left[R(s, a, s') + \gamma V(s')\right]}$$

$\Downarrow$ for a stochastic envt but deterministic policy

What ~~if the~~

next state is associated with the envt.

Policy is associated with an action

Stochastic envt:? — When we take an action 'a' in a particular state 's', the next state s' is random.

Stochastic policy:? When we are in a state 's', the ~~next~~ action we take is random.

With stochastic Policy: Action in each state has randomness.

$$V^{\pi}(s) = \sum_{a} \underbrace{\pi(a|s)}_{\text{stochas. policy}} \sum_{s'} \underbrace{P(s'|s,a)}_{\text{stochas. envt}} \underbrace{R(s,a,s') + \gamma V(s')}_{\text{backup}}$$

Using expectations:

$$V^{\pi}(s) = \underset{\substack{a \sim \pi \\ s' \sim P}}{E} \left[R(s,a,s') + \gamma V(s') \right]$$

Bellman Equation of the Q function:-

- It is the sum of the immediate reward and the discounted value of the next state-action pair.

for det. envt ↗

$$Q^{\pi}(s,a) = R(s,a,s') + \gamma Q(s',a')$$

for stocastic envt:-

$$Q^{\pi}(s,a) = \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma Q(s',a') \right]$$

for stochastic policy:-

$$Q^{\pi}(s,a) = \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma \sum_{a'} \pi(a'|s') Q(s',a') \right]$$

$\therefore$ Bellman Expectation Equation ~~of~~: the Q function ~~is~~ for a (s,a) pair is:

$$Q^{\pi}(s,a) = \sum_{s'} \overset{P(s'|s,a)}{\wedge} \left[R(s,a,s') + \gamma \sum_{a'} \pi(a'|s') Q(s',a') \right]$$

$$\therefore Q^{\pi}(s,a) = \mathop{E}_{s' \sim P} \left[R(s,a,s') + \gamma \mathop{E}_{a' \sim \pi} Q(s',a') \right]$$

Bellman Optimality Theorem:-

Recap:

$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma V(s') \right]$$

$$Q^{\pi}(s,a) = \sum_{s'} \overset{P(s'|s,a)}{\wedge} R(s,a,s') + \gamma \sum_{a'} \pi(a'|s') Q(s',a')$$

Recap:

1) Bellman Eqn for the Value fn of a State s is -

$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma V(s') \right]$$

2) Above in expectation form:-

$$V^{\pi}(s) = \mathop{E}_{\substack{a \sim \pi \\ s' \sim P}} \left[R(s,a,s') + \gamma V(s') \right]$$

3)

3) Bellman Eqn for the Q fn of (s, a) pair is:-

$$\cancel{Q^{\pi}(s,a) = P(s'|s,a)\ R(s,a,s') + \gamma \sum_{a'} \pi(a'|s') Q(s',a')}$$

$$Q^{\pi}(s,a) = \underbrace{\sum_{s'} P(s'|s,a)}\left[R(s,a,s') + \gamma \underbrace{\sum_{a'} \pi(a'|s')}\ Q(s',a')\right]$$

4) Above eqn in expectation form:-

$$Q^{\pi}(s,a) = \mathop{E}_{s' \sim P}\left[R(s,a,s') + \gamma \mathop{E}_{a' \sim \pi} Q(s',a')\right]$$

Bellman Optimality Theorem:-

We know, the Bellman eqn of the Value and Q fn in expectation form is

$$V^{\pi}(s) = \mathop{E}_{\substack{a \sim \pi \\ s' \sim P}}\left[R(s,a,s') + \gamma V(s')\right]$$

$$Q^{\pi}(s,a) = \mathop{E}_{s' \sim P}\left[R(s,a,s') + \gamma \mathop{E}_{a' \sim \pi} Q(s',a')\right]$$

To find the optimal Bellman Value fn:-

The Bellman optimality eqn, gives the optimal

Bellman value and Q fn:-

W.K.T, the value fn of a state depends on the policy.

V(s) for a particular state s, varies with the policy.

∴ the optimal Bellman V(s) for a particular state s, is the one that gives the max value among all values of ~~that~~ state.

How to find this $V^*(s)$?

We compute the optimal Bellman $V^*(s)$, by selecting the action that gives the max value in that state.

We don't know which action gives the max value

∴ ~~we~~ instead of using a policy π, ~~to~~ select an action, we find the value of the state using all possible actions, and then select the max value of all.

∵ no policy is used, remove the expectation over policy π, add the max over action.

∴ optimal Bellman Value fn of a state is:-

$$\boxed{V^*(s) = \max_a E_{s' \sim P}\left[R(s,a,s') + \gamma V(s')\right]}$$

Example:

In state s, and have 2 possible actions 0 and 1.

$$\therefore V^*(s) = \max \left(\begin{array}{l} \underset{s' \sim P}{E} \left[R(s, 0, s') + \gamma V^*(s') \right] \\ \underset{s' \sim P}{E} \left[R(s, 1, s') + \gamma V^*(s') \right] \end{array} \right)$$

Optimal Bellman Q function:-

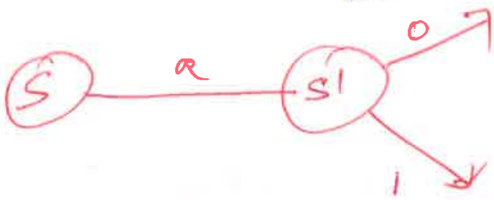
w.r.t.

$$Q^{\pi}(s,a) = \underset{s' \sim P}{E} \left[R(s,a,s') + \gamma \underset{a' \sim \pi}{E} Q(s',a') \right]$$

Instead of using a policy π for ~~choosing~~ the action a' in state s', we choose all possible actions in s', and find the max Q value.

$$\therefore Q^*(s,a) = \underset{s' \sim P}{E} \left[R(s,a,s') + \gamma \max_{a'} Q^*(s',a') \right]$$

Example:



$$\therefore Q^*(s,a) = \underset{s' \sim P}{E} \left[R(s,a,s') + \gamma \max \left(\begin{array}{l} Q^*(s',0) \\ Q^*(s',1) \end{array} \right) \right]$$

To summarize:-

1). The Bellman optimality eqn for value fn of a state s, is:-

$$V^*(s) = \max_a \underset{s'\sim P}{E} \left[R(s,a,s') + \gamma V^*(s')\right]$$

2) The Bellman optimality eqn for Q fn of a (s,a) pair is

$$Q^*(s,a) = \underset{s'\sim P}{E} \left[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right]$$

3) We can also expand the expectation & rewrite

$$V^*(s) = \max_a \sum_{s'\sim P} \left[\xrightarrow{P(s'|s,a)} (R(s,a,s') + \gamma V^*(s))\right]$$

$$Q^*(s,a) = \sum_{s'\sim P} P(s'|s,a)\left[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right]$$

The relationship between the value and Q fns:-

In chapter 1, w.k.t, the value of a state s, is the expected return starting from that state s, following a policy π.

$$V^\pi(s) = \underset{\tau\sim\pi}{E}\left[R(\tau) \mid s_0 = s\right]$$

Illy, the Q value of a (s,a) pair, is the expected return of starting from state s, by performing action, a.

$$\therefore \quad Q^{\pi}(s,a) = \mathop{E}_{\tau \sim \pi}\left[R(\tau) \mid s_0 = s, a_0 = a\right]$$

w.k.t, the optimal value fn of a state, s gives its maximum state-value.

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

~~the~~ optimal Q fn of a (s,a):-

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

Is there a relation betn. $V^*(s)$ and $Q^*(s,a)$?

w.k.t. the optimal value fn. of a state is the max expected return ~~of~~ when we start from a state. The optimal Q fn is the max. expected ~~Q value~~ return when we start from a state and perform an action 'a'.

$\therefore$ the optimal value fn of a state, is the max. of ~~all~~ optimal Q value over all possible actions ~~over~~ from a state s, $\therefore$ we can derive V from Q:

$$\boxed{V^*(s) = \max_{a}\left[Q^*(s,a)\right]}$$

Recap :-

Bellman eqn ~~of~~ the value and Q fns :-

$$V^{\pi}(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^{\pi}(s')\right]$$

$$Q^{\pi}(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma Q^{\pi}(s',a')\right]$$

$$\sum_{a'} \pi(a'|s')$$

Bellman optimality eqn of the value & Q fns :-

↳ (irrespective of any policy)

$$V^*(s) = \max_a \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right] \quad \text{(1)}$$

~~If~~ we have optimal value fn $V^*(s)$, we can use it to derive optimal Bellman Q fn, [we can derive Q from V).

$\therefore$ w.k.t: $V^*(s) = \max_a Q^*(s,a)$ ... (2)

~~: Subs (2) in (1), we get~~

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right] \quad \text{... (3)}$$

~~Also~~ ~~we~~ ~~can~~ ~~de~~

Subs (3) in (2), we get

$$V^*(s) = \max_a \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

Dynamic Programming: In RL, we use DP to find the optimal policy, using 2 methods: 1) value iteration and 2) policy iteration.

DP is a model-based method, ie, to find an optimal policy using DP, we should know the model dynamics [station transn prob, and reward functions].

1) Value iteration method: -

a) An optimal policy tells the agent to perform a correct action in each state.

b) To find this policy, first we find the optimal value fn of each state; $\hat{V}^*(s)$

c) Later use ~~this~~ optimal value for $V^*(s)$, find the optimal policy [by finding the Q function]

W.K.T the Bellman's optimal value and Q fns are

$$V^*(s) = \max_a \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma \underbrace{\max_{a'} Q^*(s',a')}\right]$$

$$\Downarrow$$

$\therefore$

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

$$\because V^*(s) = \max_a Q^*(s,a)$$

& if we know the Q-values of all (s,a) pairs of a particular state, s, using this, we can find the optimal value fn of that state.

<u>Ex:</u> Q-values of all state-action pairs

State are:-

State	Action	Value
s_0	0	2.7
s_0	1	3
s_1	0	4
s_1	1	2

Using this, we can find the optimal state values of each state as

State	Value
s_0	3
s_1	4

This is the outline of the value itn algo

The value iteration Algom:-

Step1: Compute the optimal value fn of each state iteratively, by taking the max of the Q fns (of all possible action ~~over~~ in a state), ie,

$$V^*(s) = \max_a Q^*(s,a)$$

Step2: Extract the optimal policy from the computed optimal value fn

An informal explan of the value itn algm:-

1. Initialize the value table of all states to zero.

2. ~~Compute &~~ ~~for~~ each state: s, do:

   ~~2.1 find the Q-fn of all possible actions~~

   2.1 for each possible action a in state s do:

   2.1.1 find the Q-value of this (s,a) pair as

$$Q(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V(s')\right]$$

(or)

$$Q(s,a) = \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V(s')\right]$$

2.1.2 find the max among these Q values and update that as the value of that state. $V^*(s) = \max_a Q^*(s,a)$

3. If the value table of 2 consecutive iteration doesn't change, then, go to the next step to find the optimal policy; Else repeat step 2, using the updated value table of this recent iteration.

4. find the Q value of each (s,a) pair using the optimal value table obtained in step 3 using:

$$Q(s,a) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V(s') \right]$$

5. Extract the optimal policy from the Q-value table got in step 4.

Ex: Given 

; ~~we are~~ . Goal: From state A, reach state C, without visiting B.

2 actions;

0 – left/right

1 – up/down

What is the optimal policy? Ans: perform action 1 in state A.

Given the model dynamics of state A:

State (s)	Action (a)	Next state (s')	Transn Prob $P^a_{ss'}$	Reward fn $R^a_{ss'}$
A	0	A	0.1	0
A	0	B	0.8	-1
A	0	C	0.1	1
A	1	A	0.1	0
A	1	B	0.0	-1
A	1	C	0.9	1

Step 1: Compute the optimal value function:

W.K.T. $V^*(s) = \max_a Q^*(s,a)$

1) Initialise value table to zero

2) for each state s, find $Q(s,a) = \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V(s')\right]$ for all possible actions

3) find the $\max_a Q^*(s,a)$, update the value table.

4) . Repeat steps (2) & (3) until convergence

~~Iteration 1~~:

Initialize value table to zero

State	Value
A	0
B	0
C	0

Step 1: In state A: Possible actions are (0, 1)

∴ find $Q(A,0) = \cancel{P^0_{AA}[R^0_{AA} + \gamma V(A)] +}$

$Q(A,1) =$

To using $Q(s,a) = \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$

$$\therefore Q(A,0) = P^0_{AA}[R^0_{AA} + \gamma V(A)] + P^0_{AB}[R^0_{AB} + \gamma V(B)] + P^0_{AC}[R^0_{AC} + \gamma V(C)]$$

$$= 0.1[0+0] + 0.8[-1+0] + 0.1[1+0]$$

$$= 0 - 0.8 + 0.1 = -0.7$$

$$Q(A,1) = P^1_{AA}[R^1_{AA} + \gamma V(A)] + P^1_{AB}[R^1_{AB} + \gamma V(B)] + P^1_{AC}[R^1_{AC} + \gamma V(C)]$$

$$= 0.1[0+0] + 0[-1+0] + 0.9[1+0]$$

$$= 0 + 0 + 0.9 = 0.9$$

∴ updated value table:

State	Value
A	0.9
B	0
C	0

~~for State B:~~

~~$Q(B,0) = P.$~~

~~$Q(B,1) =$~~

Similarly, if we know the model dynamics of state B & C we can find $Q(B,0)$ $Q(B,1)$ $Q(C,0)$ and $Q(C,1)$. finally updated state value table at end of itr1 (assume) is:-

State	Value
A	0.9
B	−0.2
C	0.5

Itr2: Using the value table of itr1, find Q values of all states with all possible actions in each state:

Ex: In state A:-

$$Q(A,0) = P^0_{AA}\left[R^0_{AA} + \gamma V(A)\right] + P^0_{AB}\left[R^0_{AB} + \gamma V(B)\right] + P^0_{AC}\left[R^0_{AC} + \gamma V(C)\right]$$

$$= 0.1(0+0.9) + 0.8(-1 \overset{-0.2}{\cancel{+0.0}}) + 0.1(1+0.5)$$

$$= -0.72$$

$$Q(A,1) = P^1_{AA}\left[R^1_{AA} + \gamma V(A)\right] + P^1_{AB}\left[R^1_{AB} + \gamma V(B)\right] + P^1_{AC}\left[R^1_{AC} + \gamma V(C)\right]$$

$$= 0.1(0+0.9) + 0.0(-1-0.2) + 0.9(1+0.5)$$

$$= 1.44$$

Similarly we find the Q-value of all states & update the state table: as [Assumption]

value table from itn2:

State	Value
A	1.44
B	-0.50
C	1.0

Itn 3: Repeat the same steps. While computing Q values, use the updated value table got from the previous itn.

Assume value table from itn3

State	Value
A	1.94
B	-0.70
C	1.3

Convergence?

Keep repeating, until the value table bet^n 2 consecutive itns does' change or changes by a very small (fraction based on a threshold)

Assume value table from itn4 is

state	value
A	1.95
B	-0.72
C	1.3

~~we t~~ ∵ the diff is small, we take this as the optimal value table.

Next, step 2: to extract the optimal policy from this optimal value table.

Step 2. Extract the optimal policy from the optimal value table (function) from step 1.

Assume Optimal value table is

State	Value
A	1.95
B	-0.72
C	1.3

Use the Q fn to compute the policy for a state s

2.1) find the Q-value of all (s, a) pairs as.

$$\hat{Q}(s,a) = \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V(s') \right]$$

[Use the optimal value fn of step 1 to find $V(s')$]

2.2) Extract the optimal policy, by selecting the action that has the max Q value in a state.

$$\pi^* = \arg\max_a Q(s,a)$$

for state A :-

$$Q(A,0) = P^{0}_{AA}\left[R^{0}_{AA} + \gamma V(A)\right] + P^{0}_{AB}\left[R^{0}_{AB} + \gamma V(B)\right] + P^{0}_{AC}\left[R^{0}_{AC} + \gamma V(C)\right]$$

$$= 0.1\left[0 + 1.95\right] + 0.8(-1 - 0.72) + 0.1\left[1 + 1.3\right]$$

$$= -0.951$$

$$Q(A,1) = 2.21.$$

w.k.t. the optimal value fn ~~is~~ and Q ~~fn~~ is

$$V^*(s) = \max_a \sum P(s'|s,a)\left[R(s,a,s') + \gamma \underbrace{V^*(s')}_{}\right]$$

w.k.t, given value fn, we can derive the Q fn..

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

∴ Q table is

State	Action	Value
A	0	-0.95
A	1 ✓	2.28 ✓
B	0	-0.5
B	1 ✓	0.5 ✓
C	0	-1.1
C	1 ✓	1.4 ✓

from this Q table, pick the action in each state that has the max. value as an optimal policy.

∴ In state A, the optimal policy is action 1, i.e., moving down..

Solving the frozen lake problem with value itn.

Start →

S	F	F	F
F	H	F	H
F	F	F	H
H	F	F	G

← Goal

S . . starting state

F . . frozen state

H . . . . hole state

G . . goal state.

states are encoded from 0 to 15.

Actions:

left	0
down	1
right	2
up	3

step1. Compute the optimal value fn.

```
def value_itn(env):
    num_itns = 1000
    threshold = 1e-20
    gamma = 1.0
    value_table = np.zeros(env.observation_space.n)
    for i in range(num_itns):
        updated_value_table = np.copy(value_table)
        for s in range(env.observation_space.n):
```

```
Q_values = [

        sum([Prob * (r + gamma * updated_value_table[s_]
             for prob, s_, r, _ in env.P[s][a]])
                 for a in range(env.action_space.n)]

    value_table[s] = max(Q_values)

if (np.sum(np.fabs(updated_value_table -
                          value_table))) <= threshold):

        break

return value_table.
```

step 2: finding optimal policy: -

```
def extract_policy(value_table):
    gamma = 1.0
    policy = np.zeros(env.observation_space.n)
    for s in range(env.observation_space.n):
        Q_values = [sum([Prob * (r + gamma * value_table[s_]
                      for prob, s_, r, _ in env.P[s][a]])
                          for a in range(env.action_space.n)]
        policy[s] = np.argmax(np.array(Q_values))
return policy
```

flow diagram..

value_itn(env)

main()

env = - - -

optimal_ = value_itn(env)

```
def value_iteration(env):
    :
    :
    return value_table

def extract_policy(value_table)
    return policy

main()
    // initialize env

optimal_value_fn = value_iteration(env)

optimal_policy = extract_policy(optimal_value_fn)

print(optimal_policy)
```

Policy iteration Method:-

~~1 compute the optimal value fn, using~~

In the value itn method:

1. Compute the optimal value fn iteratively, by taking the max over the Q fn (Q values)

2. Extract the optimal policy from the optimal value fn, got in step 1.

In the policy itn method:

1. Compute the optimal value fn iteratively, by using the policy.

2. Extract the optimal policy from the optimal value fn, got in step 1. [this is the same policy that generated the optimal value fn]

How to find the value fn of a state for a given policy π ?

If π is stochastic:

$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma V^{\pi}(s') \right]$$

If π is deterministic:

$$V^{\pi}(s) = \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma V^{\pi}(s') \right]$$

(or) $$V^{\pi}(s) = \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

Assume the policy is deterministic!

1. Start with a random policy π_0.
2. Initialize the value fn (table) of all states to zeros.
3. Use π_0, to find an optimal value table iteratively V^{π_0}. [This will not be optimal ~~policy~~ value table, since it is generated from a random policy]
4. Use V^{π_0} to generate a policy π_1.
5. compute V^{π_1} using π_1. Check if V^{π_1} is optimal. If so, stop. Else generate π_2 from V^{π_1}.
6. Use π_2 to generate a V^{π_2}. If V^{π_2} is optimal stop. Else generate a new policy π_3 .. and so on.

$$\underset{\text{random policy}}{\pi_0} \to V^{\pi_0} \to \pi_1 \to V^{\pi_1} \to \pi_2 \to V^{\pi_2} \dots \pi^* \to \underset{\text{optimal policy}}{V^{\pi^*}}$$

How to decide that the value fn is optimal?

If it doesn't change over iterations?

Since the value fn is generated from a policy, over a series of iterns, if the policy doesn't change betn 2 consecutive iterations, then it is an optimal policy π^*.

The value fn generated from π^* is the optimal value fn V^{π^*}

Psuedo-code:-

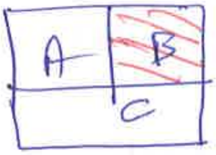
```
policy = random-policy
for i in range(num_itns):
    value_fn = compute-value-fn(policy)
    new_policy = extract-policy(value_fn)
    if policy == new policy:
        break
    else
        policy = new-policy.
```

Algorithm for policy itn:-

1. Initialize a random policy
2. Compute the value fn iteratively, from this policy
3. Extract a new-policy from the value fn of step(2)

4. If the extracted policy is the same as the policy generated in step 2, then stop, else goto step(2) with this new-policy & repeat steps 2 to 4.

Ex: [Pg: 118]



States: A ... 0
B ... 1
C ... 2

Actions: 0 – left/right; 1 – up/down.

Goal: from A, reach C, without visiting B.

Given model dynamics of state A.

State s	Action a	Next state s'	$P^a_{ss'}$	$R^a_{ss'}$
A	0	A	0.1	0
A	0	B	0.8	−1
A	0	C	0.1	1
A	1	A	0.1	0
A	1	B	0.0	−1
A	1	C	0.9	1

Step1:- Initialize a random policy:–

$A \to 1$; $B \to 0$; $C \to 1$.

Step2: Compute the value fn using this random policy.

$$V^{\pi}(s) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^{\pi}(s') \right]$$

~~Step(3): Initialize the value table of all states to~~ zero.

State	Value
A	0
B	0
C	0

Initial value table

$\boxed{\gamma = 1}$

~~step~~ Itn 1: find the $V(s)$ for a state, only for ~~those~~ actions mentioned in the policy, π.

$\therefore$ $V(A)$ . . only for action 1.

$$\therefore V(A) = P^1_{AA}\left[R^1_{AA} + \gamma V(A)\right] + P^1_{AB}\left[R^1_{AB} + \gamma V(B)\right]$$

$$+ P^1_{AC}\left[R^1_{AC} + \gamma V(C)\right]$$

$$= 0.1[0+0] + 0[-1+0] + 0.9[1+0]$$

$$= 0.9$$

for state B, ~~find~~ only action 0; if we know the model dynamics

$\therefore V(B) = P_B$

Illy, from the model dynamics of states B and C, find $V(B)$ for action 0 and $V(C)$ for action 1.

Assume the value table from itn 1 is

State	Value
A	0.9
B	−0.2
C	0.1

This will not be optimal.

Itn2: find V(S) using the value table ~~of~~ from

Itn1. ~~the~~

$$V(A) = P'_{AA^*}[R'_{AA^*} + \gamma V(A)] + P'_{AB}[R'_{AB} + \gamma V(B)]$$

$$+ P'_{AC}[R'_{AC} + \gamma V(C)]$$

$$= 0.1[0 + 0.9] + 0[-1 - 0.2] + 0.9[1 + 0.1$$

$$= 1.08$$

Illy keep & find V(B) and V(C). Assume value table from itn2 is

State	Value
A	1.08
B	-0.5
C	0.5

value table from itn3 is

A	1.45
B	-0.9
C	0.6

value table from itn4 is

A	1.45
B	-0.9
C	0.6

∵ No change the ~~optimal~~ final value table is ~~from the~~

Step3: Extract a new policy using this value fn from step2:

∵ policy ~~is generated~~ tells us which is the correct action in each state. This is given by

$$Q(s,a) = \sum_{s'} P_{ss'}^{a} \left[R_{ss'}^{a} + \gamma V(s') \right]$$

for all (s,a) pairs of a state s.

for state A:-

∴ find $Q(A,0)$: $P_{AA}^{0}\left[R_{AA}^{0} + \gamma V(A)\right] + P_{AB}^{0}\left[R_{AB}^{0} + \gamma V(B)\right] + P_{AC}^{0}\left[R_{AC}^{0} + \gamma V(C)\right]$

$= 0.1[0 + 1.46] + 0.8(-1 - 0.9) +$

~~Q(A,?)~~ : $0.1[1 + 0.61]$

$= -1.21.$

$$Q(A,1) = P_{AA}^{1}\left[R_{AA}^{1} + \gamma V(A)\right] + P_{AB}^{1}\left[R_{AB}^{1} + \gamma V(B)\right] + P_{AC}^{1}\left[R_{AC}^{1} + \gamma V(C)\right]$$

$= 0.1[0 + 1.46] + 0.0[-1 - 0.9] + 0.9[1 + 0.61]$

$= 1.59$

Illy do for states B and C for all actions.

∴ Q table is

State	Action	Value
A	0	-1.21
A	1	1.59
B	0	0.1
B	1	0.0
C	0	0.5
C	1	0.0

from this Q table, pick the action in each state, that gives the max. value. This ~~gives~~ gives a new policy.

A → 1; B → 0; C → 0.

Step 4: check the new policy

If the extracted new policy from step (3) is same as the policy used in step (2), then stop: else goto step 2 with this new policy & repeat steps 2 to 4.

Solving the Frozen Lake Problem with policy iteration

main()

```
import gym
import numpy as np
env = gym.make('FrozenLake-v0')
optimal_policy = policy_itn(env)
print(optimal_policy)
```

```
def policy_itn(env):

    num_itns = 1000
    // initialize policy to zero in all states
    policy = np.zeros(env.observation_space.n)

    for i in range(num_itns):

        val_fn = compute_val_fn(policy)

        new_policy = extract_policy(val_fn)

        if (np.all(policy == new_policy)):
            break

        policy = new_policy

    return(policy)

def compute_val_fn(policy):
    num_itns = 1000
    threshold = 1e-20
    gamma = 1.0
    // init. val. table of all states to zero
    val_tab = np.zeros(env.observation_space.n)

    for i in range(num_itns):

        updated_val_tab = np.copy(val_tab)

        for s in range(env.observation_
                                 space.n):
            // find value of a state only for action 'a' as in
                                                    the policy
            a = policy[s]
```

// $V^{\pi}(s) = \sum_{s'} P^{a}_{ss'} [R^{a}_{ss'} + \gamma V^{\pi}(s')]$

```
val_tab[s] = sum(
        [prob * (r + gamma * updated_val_tab[s_])
              for prob, s_, r, _ in env.P[s][a]])

if (np.sum(np.fabs(updated_val_tab - val_tab)) <=
                                   threshold:
    break
return(val_tab)

def extract_policy(val_tab):
```

// policy is generated using $Q(s,a)$, for all
// actions 'a' in a state 's'; not with resp. to any π.
// $Q(s,a) = \sum_{s'} P^{a}_{ss'} [R^{a}_{ss'} + \gamma V(s')]$

```
    gamma = 1.0
```

// Init. policy of all states to zero.

```
    policy = np.zeros(env.observation_space.n)
```

~~for i in range(num.~~

```
    for s in range(env.observation_space.n):
        Q_values = [sum([prob * (r + gamma *
                                      val_tab[s_])
                  for prob, s_, r, _ in env.P[s][a]])
                      for a in range(env.action_space.n)]
```

policy[s] = np.argmax(np.array(Q-values))
// this returns the index of max Q-value ~~that~~

return policy.

Limitations of DP :-

DP is a model-based method to find the optimal policy. In both value & policy itn methods, we need to know the model dynamics [transn probabilities] and reward functions.

In value iteration:-

$$V^*(s) = \max_a Q^*(s,a)$$

1. find the optimal val fn of a state 's', by finding the max over all Q functions (s,a) of all actions in state 's'.

$$Q(s,a) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V(s') \right]$$

2. Extract the optimal policy from this optimal val fn

In policy iteration:-

1. find the optimal val fn of a state 's', by ~~finding the~~ using the policy iteratively

$$V^\pi(s) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right]$$

start with a random policy and find the value fn.

2. find The policy that generated this optimal value fn, is the optimal policy.

$\therefore$ in both methods, we should know $P_{ss'}^{a}$.

To find the optimal policy ~~is a stochastic env~~, without knowing model dynamics, we use model-free methods like the Monte Carlo methods.