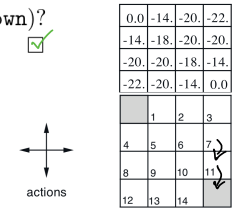
Exercise 4.1 In Example 4.1, if π is the equiprobable random policy, what is $q_\pi(11, \texttt{down})$? What is $q_\pi(7, \texttt{down})$?

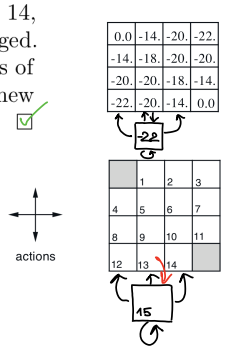


$$q_\pi(11, \text{down}) = -1 + \underbrace{1 \cdot q_\pi(T, \text{exit})}_{=0} = -1$$

$$q_\pi(7, \text{down}) = -1 + \sum_{\forall a} \pi(a \mid 11)\, q(11, a) =$$

$$= -1 + v_\pi(11) = -15$$

Exercise 4.2 In Example 4.1, suppose a new state 15 is added to the gridworld just below state 13, and its actions, `left`, `up`, `right`, and `down`, take the agent to states 12, 13, 14, and 15, respectively. Assume that the transitions *from* the original states are unchanged. What, then, is $v_\pi(15)$ for the equiprobable random policy? Now suppose the dynamics of state 13 are also changed, such that action `down` from state 13 takes the agent to the new state 15. What is $v_\pi(15)$ for the equiprobable random policy in this case?



$$v_\pi(15) = -1 + \sum_{\forall a} \sum_{s' \in \{12\text{-}15\}} \underbrace{\pi(a \mid 15)}_{=1/4} v_\pi(s') = -1 + \frac{1}{4} \sum_{s' \in \{12\text{-}15\}} v_\pi(s')$$

$$v_\pi(15) = -4/3 + \frac{1}{3}\left(v_\pi(12) + v_\pi(13) + v_\pi(14)\right) = -20$$

Allowing the down transition from state 13 couples the Bellman equations for states 13 and 15. Coupling creates a linear system

$$v_\pi(13) = -1 + \frac{1}{4}\left(v_\pi(9) + v_\pi(12) + v_\pi(14) + v_\pi(15)\right)$$

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

$$v_\pi(15) = -4/2 + \frac{1}{2}\left(\frac{1}{5} v_\pi(9) + v_\pi(12) + v_\pi(14)\right) = -22 \text{ (approx.)}$$

The calculation above is a "2nd neighbor" approximation. It assumes that v_π of these states $(9, 12, 13)$ won't change. Getting the precise solution would require re-solving the system of Bellman equations with state 15 because adding a new state introduces a new way for the random agent to get "lost" in it's wandering and increases the mean number of steps required to stumble upon an exit.

Exercise 4.3 What are the equations analogous to (4.3), (4.4), and (4.5) for the action-value function q_π and its successive approximation by a sequence of functions $q_0, q_1, q_2, \ldots$?

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] =$$

$$= \mathbb{E}_\pi\left[R_{t+1} + \gamma \sum_{\forall a'} \pi(a' \mid S_{t+1}) q_\pi(S_{t+1}, a') \mid S_t = s, A_t = a\right] =$$

$$= \sum_{\forall s' \forall r} p(s', r' \mid s, a)\left(r + \gamma \sum_{\forall a'} \pi(a' \mid s') q_\pi(s', a')\right)$$

$$q_{k+1}(s,a) = \mathbb{E}_\pi\left[R_{t+1} + \gamma \sum_{\forall a'} \pi(a' \mid S_{t+1}) q_k(S_{t+1}, a') \mid S_t = s, A_t = a\right] =$$

$$= \sum_{\forall s' \forall r} p(s', r' \mid s, a)\left(r + \gamma \sum_{\forall a'} \pi(a' \mid s') q_k(s', a')\right)$$

Exercise 4.4 The policy iteration algorithm on page 80 has a subtle bug in that it may never terminate if the policy continually switches between two or more policies that are equally good. This is ok for pedagogy, but not for actual use. Modify the pseudocode so that convergence is guaranteed. □

Termination should be based on the number of iterations that the policy evaluation step needs to reduce maximal update size below Δ. If it's only one pass then we know the policy has converged to one of the optimal ones.

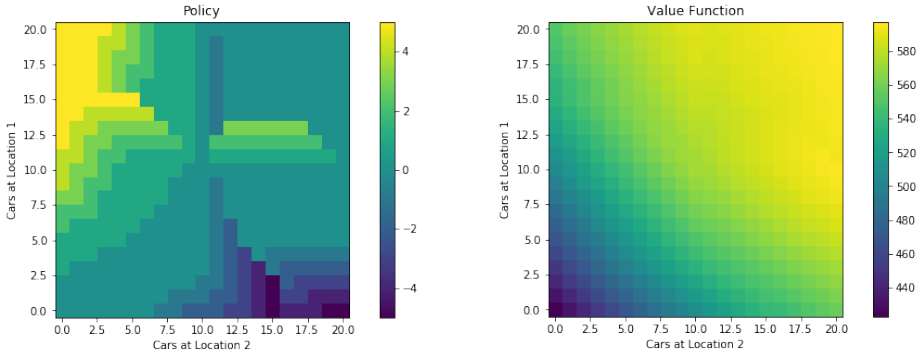
Exercise 4.5 How would policy iteration be defined for action values? Give a complete algorithm for computing q_*, analogous to that on page 80 for computing v_*. Please pay special attention to this exercise, because the ideas involved will be used throughout the rest of the book. ☑

1. Arbitrary initialization of $Q(s,a)$ and $\pi(s)$
2. Policy evaluation
   Loop $s \in S$:
   Loop $a \in A$:
   $$Q(s,a) = \sum_{s',r} p(s',r|s,a)\left(r + \gamma\, Q(s', \pi(s'))\right)$$
   Same accuracy criteria termination check
3. Policy improvement
   $\pi(s) = \underset{a}{\text{argmax}}\ Q(s,a)$
   Same stability termination check

Exercise 4.6 Suppose you are restricted to considering only policies that are ε-*soft*, meaning that the probability of selecting each action in each state, s, is at least $\varepsilon/|\mathcal{A}(s)|$. Describe qualitatively the changes that would be required in each of the steps 3, 2, and 1, in that order, of the policy iteration algorithm for v_* on page 80. ☑

1. Initialize policy to $\pi(a|s) = \frac{1}{A(s)} \quad \forall s$
2. Value function update needs to consider expectation over actions
   $$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left(r + \gamma V(s')\right)$$
3. The improved policy has to satisfy the ε-soft constraint
   $$\pi(a|s) = \begin{cases} 1 - \varepsilon|A(s)| & \text{if } a = \underset{a}{\text{argmax}} \sum_{s',r} p(s',r|s,a)\left(r + \gamma V(s')\right) \\ \varepsilon/|A(s)| & \text{otherwise} \end{cases}$$

Exercise 4.7 (programming) Write a program for policy iteration and re-solve Jack's car rental problem with the following changes. One of Jack's employees at the first location rides a bus home each night and lives near the second location. She is happy to shuttle one car to the second location for free. Each additional car still costs $2, as do all cars moved in the other direction. In addition, Jack has limited parking space at each location. If more than 10 cars are kept overnight at a location (after any moving of cars), then an additional cost of $4 must be incurred to use a second parking lot (independent of how many cars are kept there). These sorts of nonlinearities and arbitrary dynamics often occur in real problems and cannot easily be handled by optimization methods other than dynamic programming. To check your program, first replicate the results given for the original problem. ☑



Exercise 4.8 Why does the optimal policy for the gambler's problem have such a curious form? In particular, for capital of 50 it bets it all on one flip, but for capital of 51 it does not. Why is this a good policy? ☑

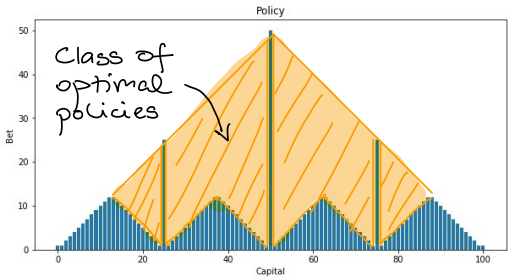
The trick to optimal policy when the stakes are against the agent (i.e. probability of head is less than 1/2) is to not play the game. Or come as close as possible to not playing as possible. Because the problem is not discounted, all losses are equivalent and it doesn't matter how many betting rounds were played prior to losing. Equivalently all wins achieve the same score independently of the number of betting rounds prior to winning but each bet lowers the chance of winning.

Best strategy is therefore to win the game in as few bets as possible. This translates into the following policy. Above 50% of the target capital the agent should attempt to end the game with a precise bet to get to the target. Betting above the target means lower chances of recovery in cases when the bet fails.

Below 50% the goal is to leap-frog to a state with capital above 50% of the target. Again, in as few bets as possible. There many correct choices to leap-frog to above 50% goal from any given score. For instance, starting from capital 30, all these strategies have the same probability of achieving the goal because they need two wins in a row:

```
30 Bet: 20 -> 50 Bet: 50 -> 100
...
30 Bet: 30 -> 60 Bet: 40 -> 100
```

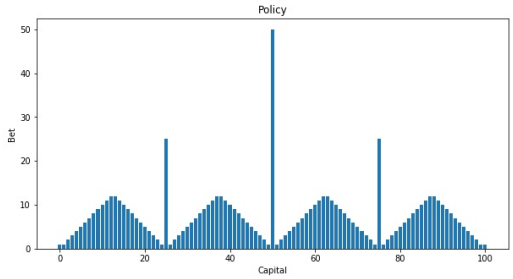
Also, the agent goes bust, if one of the bets fails. The "common denominator" of these polices is that they need to somehow bet and win the 70 remaining capital points. And of course, it's not possible to do that without risking 70 capital sooner or later in the game.

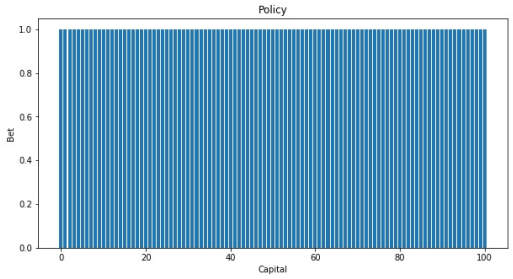


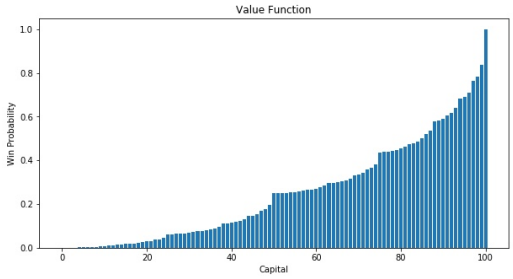
The "common denominator" gives rise to an entire class of optimal policies with their bets distributed arbitrarily in the shaded area in between "large triangle" policy and "tiny triangles" optimal policy as chosen in the book.

The policy displayed in the book is probably the consequence of a common strategy of breaking ties with the first found maximal value. This behavior is a common implementation detail of many argmax-style functions across many programming languages.

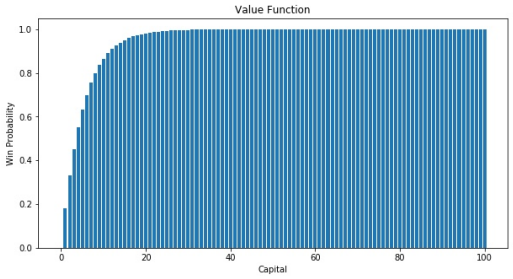
Exercise 4.9 (programming) Implement value iteration for the gambler's problem and solve it for $p_h = 0.25$ and $p_h = 0.55$. In programming, you may find it convenient to introduce two dummy states corresponding to termination with capital of 0 and 100, giving them values of 0 and 1 respectively. Show your results graphically, as in Figure 4.3. Are your results stable as $\theta \to 0$?







$p_h = 0.25$



$p_h = 0.55$

Exercise 4.10 What is the analog of the value iteration update (4.10) for action values, $q_{k+1}(s,a)$?

$$q_{k+1}(s,a) = \sum_{s',r} p(s',r \mid s,a)\left(r + \gamma \max_{a'} q_k(s',a')\right)$$