Errata and Notes for:

**Reinforcement Learning: An Introduction**
by Richard S. Sutton and Andrew G. Barto

---

### Errata:

- p. xviii, Ben Van Roy should be acknowledged only once in the list. (Ben Van Roy)
- p. 29 (Figure 2.1). In the upper graph, the third line is unlabeled, but should be labeled "epsilon=0 (greedy)".
- p. 29 (Figure 2.1). If you try to reproduce these graphs, you will find that in order to get the detailed structure at the very beginning to match, you will have to use a policy that is epsilon-greedy (as described in the text) and that breaks ties randomly. The text does not say how ties (multiple actions with the maximal action value) were handled, but in fact they were chosen among randomly, which significantly affects the very beginning of the run. Random tiebreak is generally a good idea, though of course it only has an effect at the beginning of time, which happens rarely ;-). You can get a similar effect, with less programming, just by initializing the action values not to zero, but to very small random numbers. (Garett Hunter)
- p. 61. The upper limit in Equation 3.3 should be T-t-1.
- p. 78. In the 2nd max equation for V*(h), at the end of the first line, "V*(h)" should be "V*(l)". (Christian Schulz)
- p. 98, Figure 4.3. For some MDPs the given policy iteration algorithm never terminates. The problem is that there may be small changes in the values computed in step 2 that cause the policy to forever be changing in step 3. The solution is to terminate step 3 not when the policy is stable, but as soon as the largest change in state value due to a policy change is less than some epsilon. (Stuart Reynolds)
- p. 105, Figure 4.7. The caption should end "with each other" (the each is missing). (Nikos Vlassis)
- p. 121, Figure 5.5. The two 3D graphs, above and below, should be switched. (Terence Schauenberg)
- p. 127, Figure 5.7. The first two lines of step (c) refer to pairs s,a and times t at or later than time tau. In fact, it should only treat them for times later than tau, not equal. (Thorsten Buchheim) No, wait, on further consideration, it is right as printed. The action at tau may be non-greedy, but everything after it is, so we can use it to learn its value. In fact, this is the only way the value of non-greedy actions are learned. (Ashique Mahmood)
- p. 128, Exercise 5.4, second line from the end in the exercise, "on-policy" should be "off-policy".
- p. 129, Exercise 5.5, the word "stationary" should be "sample". (Kenichi Kato)
- p. 146, the windy gridworld example apparently used alpha=0.5 rather than alpha=0.1 as stated. (Yuxi Li)
- p. 147, the second line should read "17 steps, two more than the minimum of 15" instead of "two less" as stated. (Barry D. Nichols)
- p. 151, second line of the equation, pi(s_t,a_t) should be pi(s_{t+1},a_t), and the conditioning on s_t in the first line should similarly be conditioning on s_{t+1}. (Dan Bernstein) This algorithm, called Expected Sarsa, has since been extensively studied by Harm van Seijen et al. in a 2009 paper published the Proceedings of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning.
- p. 155, the parameter alpha was 0.01, not 0.1 as stated. (Abinav Garg)
- p. 157, first sentence of section 6.9. "shown" should be "showed".
• p. 174, 181, 184, 200, 212, 213: in the boxed algorithms on all these pages, the setting of the eligibility traces to zero should appear not in the first line, but as a new first line inside the first loop (just after the "Repeat..."). (Jim Reggia)
• p. 197, bottom formula last theta_t(2) should be theta_t(n). (Dan Bernstein)
• p. 201, the two MSEs should be RMSEs, that is, they should be square roots of the MSEs. (Tao Wang and Dale Schuurmans)
• p. 212-213. There are a number of small problems with these algorithms. Instead of describing minimal fixes, I would prefer to rewrite them as here for Sarsa(lambda) with FA and here for Watkins’s Q(lambda) with FA.
• p. 215, Figure 8.11, the y-axis label. "first 20 trials" should be "first 20 episodes".
• p. 215. The data shown in Figure 8.11 was apparently not generated exactly as described in the text, as its details (but not its overall shape) have defied replication. In particular, several researchers have reported best "steps per episode" in the 200-300 range.
• p. 215. Two-thirds down the page, "alpha = 0.05 (0.1/m)” should be “alpha = 0.05 (0.5/m)”. m here is 10, the number of tilings.
• p. 218, Figure 8.13. The funny symbol between the two thetas at the top should be just a dash, as in theta(1) through theta(5), which all behave as shown in this line.
• p. 220, the last term of the last equation should have a gamma before it (before the expectation).
• p. 233, last line of caption: "ne-step" should be "one-step". (Michael Naish)
• p. 245, the analysis whose results are shown in Figure 9.13 is slightly in error. The text on the next page says that the error is reduced according to 1/sqrt{t} times (b-1)/b, but it should be 1/sqrt{t} times sqrt{(b-1)/b}. The corrected version of Figure 9.13 (which is slightly changed) is available here.
• p. 259, the reference to McCallum, 1992 should be to Chrisman, 1992. And in the references section, on p. 302, the (incorrect) listing for McCallum, 1992 should not be there. (Paul Crook)
• p. 267, Table 11.1. The number of hidden units for TD-Gammon 3.0 is given as 80 but should be 160. (Michael Naish)
• p. 282, middle paragraph, it says Fig. 11.10 shows mean arrival rates of 150, 200, and 300, but the last number should be 350 rather than 300. (Susan Overstreet)
• p. 309, the reference for Tsitsiklis and Van Roy (1997b) should be to Technical Report LIDS-P-2390, Massachusetts Institute of Technology. (Ben Van Roy)
• p. 322, in the index entry for TD error, the range listed as "174-165" should be "174-175". (Jette Randlov)

Other Notes:

• p. 28. The description of the 10-armed testbed could be clearer. Basically there are 2000 randomly generated 10-armed bandits. The Q*(a) of each of these were selected from a normal distribution with mean 0 and variance 1. Then, on each play with each bandit, the reward was determined by adding to Q*(a) another normally distributed random number with mean 0 and variance 1.
• The Gambler’s Problem, pages 101-103. The optimal policy shown in Figure 4.6 is not unique. There are many other optimal policies, all of which share the optimal function also shown in the figure. Which optimal policy you get from your value iteration algorithm depends on how it breaks ties within the numerical resolution of your computer. An email discussion of this issue is here. (Daniel Loeb, Daniel Polani, Francesco Decomite, and many students)
• p. 127, Figure 5.7. This algorithm is only valid if all policies are proper, meaning that they produce episodes that always eventually terminate (this assumption is made on the first page of the chapter). This restriction on environments can be lifted if the algorithm is modified to use epsilon-soft policies, which are proper for all environments. Such a modification is a good exercise for the reader! Alternative ideas for off-policy Monte Carlo
learning are discussed in this recent research paper.

- Chapter 5. John Tsitsiklis has obtained some new results which come very close to solving "one of the most important open theoretical questions in reinforcement learning" -- the convergence of Monte Carlo ES. See here.

- Exercise 7.3. This is a difficult exercise. Igor Karpov has made a thorough answer available here.

- Chapter 8. The overall conclusions of this chapter have to be re-evaluated in light of the recent development of true gradient-descent TD methods. The current view is much more straightforward and positive. In particular, we now have stable and efficient algorithms for off-policy TD learning with linear function approximation, for nonlinear function approximation, for off-policy control (in preparation), and for general option-conditional predictions. It seems likely that these new ideas will result in a whole new generation of reinforcement learning algorithms for which function approximation is much more straightforward.

- p. 212-213. In these two algorithms, it is implicit that the set of features for the terminal state (and all actions) is the empty set. It would be better if this case was treated explicitly in the pseudo-code, and this is now done for the rewritten versions of these figures mentioned in the errata.

- The last equation on page 214 can be a little confusing. The minus sign here is meant to be grouped with the 0.0025 (as the spacing suggests). Thus the consecutive plus and minus signs have the same effect as a single minus sign. (Chris Hobbs)

- Dyna-Q+ involved two other changes not mentioned in the book. First, in planning steps actions were considered from states even if those actions had never previously been taken. Second, the initial model for these actions was that they produced a reward of zero and returned the agent to the same state. The number of steps since last taking them was of course the total number of steps, as if they had all been taken on step 0. (Kavosh Asadi)

- Discounting and function approximation. Since writing the book, it has become apparent to me that discounting as it is widely used is inconsistent with function approximation. In particular, the formulation of the problem given in Section 3.8 breaks down whenever there is a restriction on the policy space (such as occurs to greedy policies when the value function is approximated). In these cases, one cannot necessarily chose a policy that optimizes the value of all states simultaneously, as the best policy for one state may not be the best from another. The only way out is to specify which states one cares about. For example, one may specify a start state (or start state distribution) as the state one cares about optimizing. Alternatively, one might hope to use the distribution of states actually experienced under the policy. Interestingly, however, in this case one can show that the discount rate has no effect on the policy ordering, and that the effective result is the same as the undiscounted (or "average reward") case outlined in Section 6.7. The lesson, I think, is that we should switch from discounted reward to average reward as our base case in thinking about reinforcement learning and artificial intelligence.