Programming Assignment 1: Tic-Tac-Toe

For my first programming assignment, I implemented the tic-tac-toe extended example from Chapter 1. According to the text, this learner uses a simple form of *temporal distance* learning.

The value function is identical to that given in the text:

$$V(s) = V(s) + \alpha[V(s') - V(s)] \tag{1}$$

In almost all of my results, the metric shown is games won versus an opponent playing randomly. This is equivalent to reward, as in my simulations a reward of 1 is given for winning, while no reward is given for losses or draws. In all results, α is set to 0.9, and the initial value of states is set to 0.5. In all results with fixed-ϵ, ϵ-greedy algorithms, the algorithms learn while exploring.

In Figure 1, I show random X players versus random O players. In this plot, I show the cumulative wins over time for each of five pairs of X and O players. This is an important baseline graph, because it shows that X has a significant advantage just from being the first to move. Though both sides are playing randomly, X ends up with close to double the wins of O (the X players are the higher grouping of runs).

In Figure 2, I plot the same data, only instead of plotting all pairs I plot the mean of five runs. The grouping in the previous graph is tight enough around these means that I use means over five pairs of players in all other graphs.

In Figure 3, I show a learning X player versus a random O player. This learning player does much better than the random X player in 2. The learner is ϵ-greedy, and ϵ is set to 0.1.

In Figure 4, I show a learning O player versus a random X player. This learning player also does much better than its random counterpart and even manages to come out ahead of the X player. The learner is ϵ-greedy, and ϵ is set to 0.1.

In Figure 5, as suggested by an exercise in Chapter 1, I show two learning agents pitted against each other, each ϵ-greedy with $\epsilon = 0.1$. The win margin of X over O is smaller than in the random case.

In Figure 6, I show the effect of ϵ on learning rate and final outcome. I show only the averaged X players, omitting the O players. After 10000 games, the players end up in order by ϵ, with the largest value doing worst and the smallest value doing best. This is as expected, as they all eventually learn the correct policy, but the larger ϵ values choose non-greedily a larger percentage of the time.

In Figure 7, I show how ϵ affects the number of states encountered by the learner. In my implementation, state values are cached as soon as they are observed for possible transition (a learner examines the values of all possible next states at each step and caches their values). The higher ϵ learners converge faster and come out higher at the "knee" of the graph, but they all converge near the end of the runs. The symmetry of the board is not exploited in my implementation. After all runs, all learners are converging to around 5000 states; as a loose upper bound, for the tic-tac-toe board, $3^9 = 19683$ states are possible (though many of these are not reachable in actual gameplay).

Finally, in Figure 8 I show decreasing ϵ over time versus a random player. In this graph, I start ϵ at 0.99 and discount it after every game, so the exploration rate decreases like

$$0.99, 0.99^2, 0.99^3 \cdots \tag{2}$$

This player does not learn while exploring, as appropriate for a player that will eventually stop exploring (essentially). This player does better than all my other examples, in the end winning over 8000 of 10000 games.

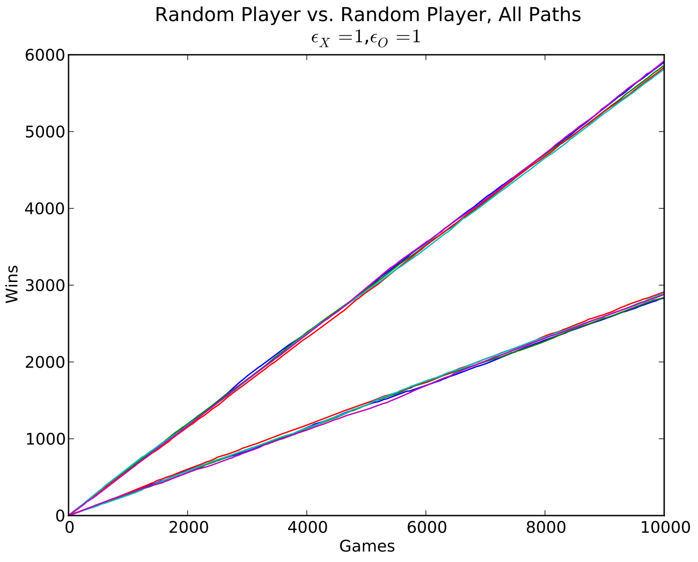

Figure 1:

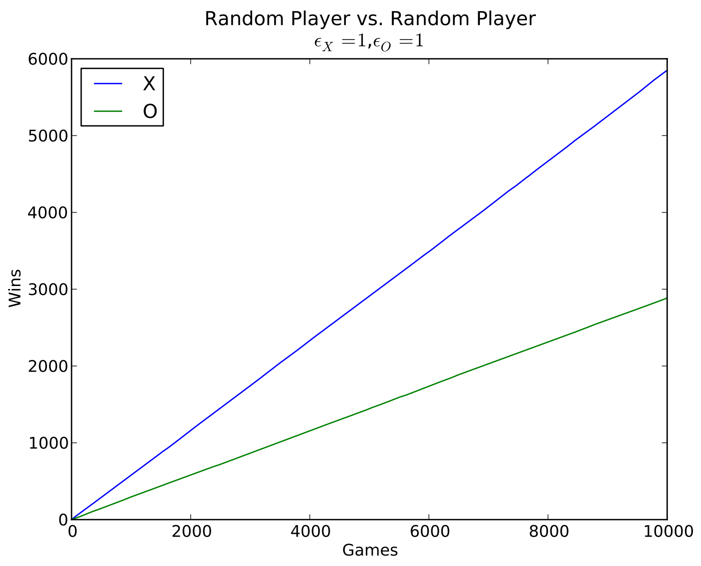

Figure 2:

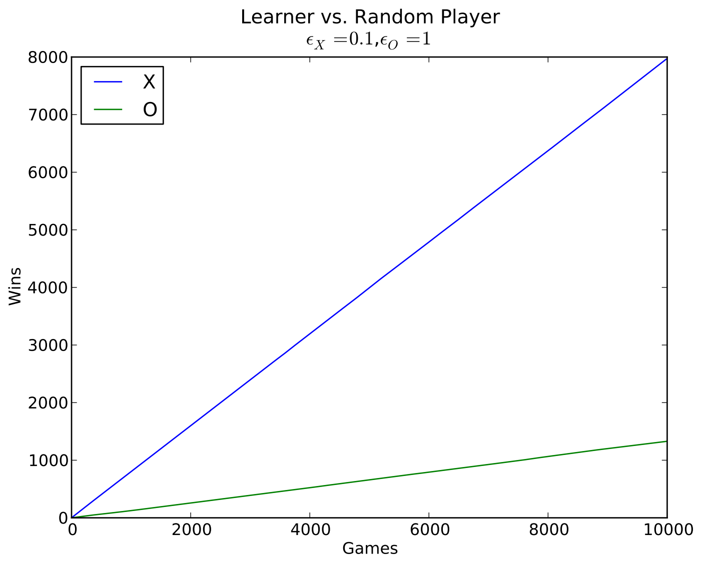

Figure 3:

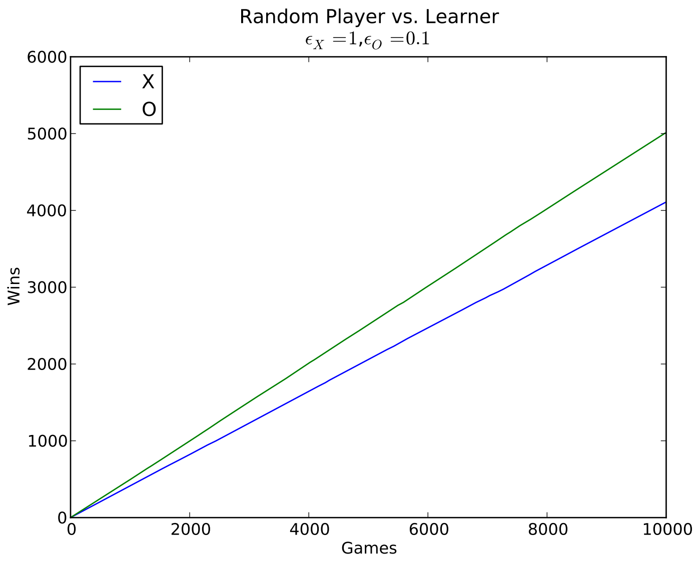

Figure 4:

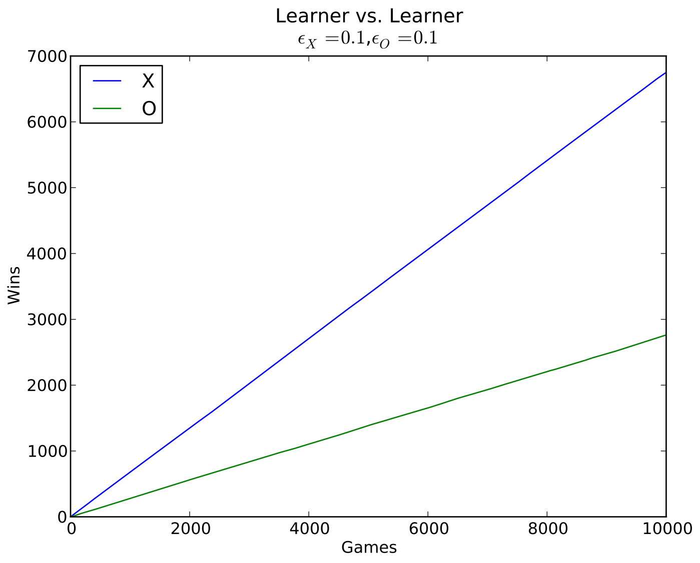

Figure 5:

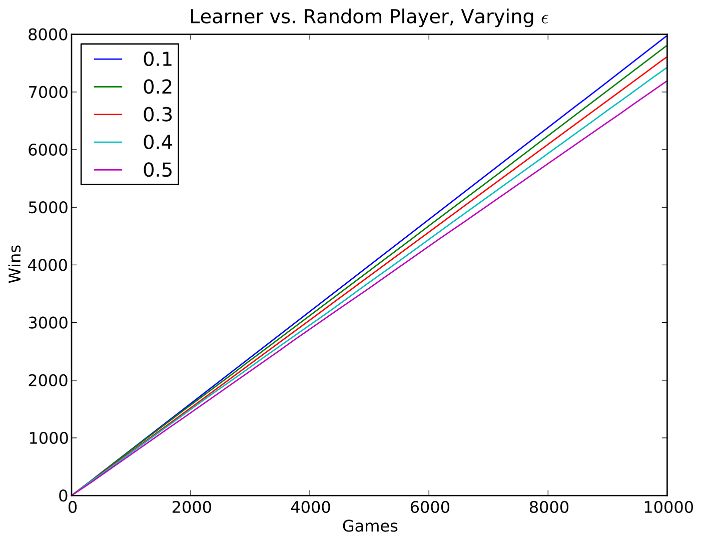

Figure 6:

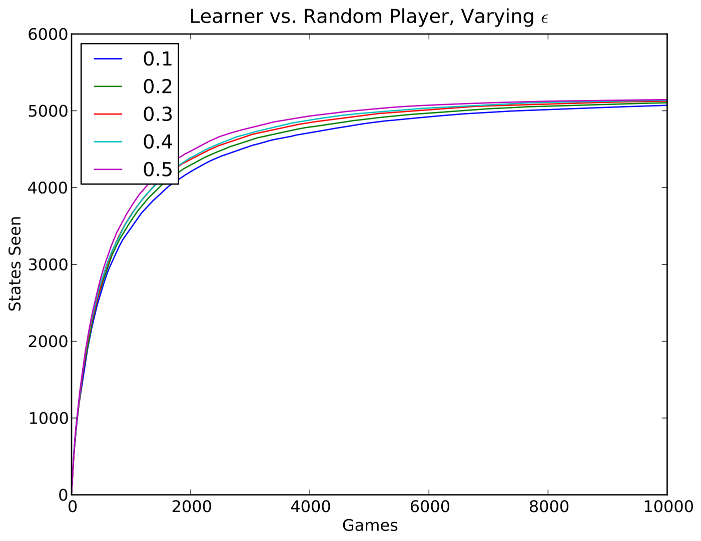

Figure 7:

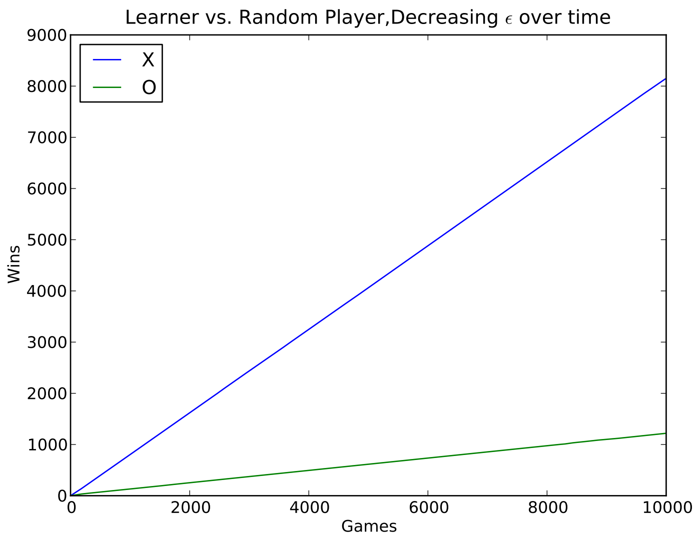
Learner vs. Random Player,Decreasing ϵ over time
X
O
Wins
Games
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
0
2000
4000
6000
8000
10000

Figure 8: