MATH4406 CONTROL THEORY: HW4

Stephen Lynch 42674223

September 26, 2014

Question 1

Imagine the following real-life situation. You are driving down an 11-lane super-highway on the Gold Coast with your trashy girlfriend, who is completely deaf, in the passenger seat. The 9 lanes in the middle are all underground, and separated by concrete barriers. At 10km intervals, the road forks and you are forced to move to the lane on either the right or left. Your girlfriend decides which direction to move, and she's fairly unpredictable - half the time she chooses left and half the time right. However, it's fairly dark in the tunnels and honey bunny can only communicate with sign language, so you only actually see her choice half of the time. If you miss her signal, no worries, you just move outwards from the center since the speed limit is higher the closer you are to the edge, and you just widened the hole in your muffler - this makes the car sound cooler when you drive faster. Hence, moving to an outer lane increases street cred, which is what we would like to maximise. However, you also just robbed a 7-11 to get a Slurpee for Bonnie, so the police helicopter is after you. Driving in either of the outermost lanes is extremely costly, since they're exposed to the skies, and cages are for birds. Indeed, if you go to jail your girlfriend will dump you for some guy with a bigger southern cross tattoo, and someone will steal your custom muffler for sure, which costs -15 for reasons which are arbitrary. On the bright side, if you're not in the tunnel then sign language is suddenly a much more viable form of communication! Your gf chooses to stay in the outer lane or move inwards with equal probability, and you obey, because the fear of being cheated on is ever-present... Also, the highway is circular and has no exits, because the GCCC is too terrified of the bogans who wanted a sick racetrack, so you have to drive forever. Also, λ describes the laziness of the police on duty that day - as time wears on, they are more likely to land on the helipad at McDonalds in Nerang, so the cost of moving to the outer lanes decreases with time.

Question 2

Certainly we would expect optimal policies to be symmetric about 0, up to multiplication by -1, simply because the reward structure is symmetric. When λ is close to one, the impact of rewards in the future is negligible, so the optimal policy will be one of instant gratification - always move outwards. On the other hand, when $\lambda$is close to 1, the discounting effect is lessened, so choosing a path that ends up in an outer lane even in the distant future will be very costly. Hence an optimal policy should strike some balance between always moving outwards, and staying away from the endpoints, e.g.

$$(1, 1, -1, -1, 1, 1, 1, -1, -1)$$

might be an optimal policy. Here the nine entries correspond to decisions made in states

$$(-4, -3, -2, -1, 0, 1, 2, 3, 4).$$

Question 3

We determined the optimal policy for λ values ranging from 0.01 to 0.99 at intervals of 0.01, using: policy enumeration, value iteration and policy iteration. The code used to implement these three algorithms is included in the appendix. Each algorithm gave the same results - these are presented in the table below, and mesh with our predictions in Question 2:

λ range	Optimal policy
0.01 - 0.63	(-1,-1,-1,-1,1,1,1,1,1)
0.64 - 0.85	(1,-1,-1,-1,1,1,1,1,-1)
0.86 - 0.92	(1,1,-1,-1,1,1,1,-1,-1)
0.93 - 0.95	(1,1,1,-1,1,1,-1,-1,-1)
0.96 - 0.99	(1,1,1,1,1,-1,-1,-1,-1)

Notably, whenever $\lambda \leq 0.6$, the optimal policy is to just be greedy and always move outwards. Above this threshold, we have to be more careful.

Appendix

GENERATING POLICIES:

```
P = dec2bin([0:511].');
policies = zeros(512,9);
for i = [1:512]
    for j = [1:9]
        policies(i,j) = str2num(P(i,j));
    end
end
policies = 2 * policies - 1;
```

POLICY ENUMERATION CODE:

```
V_max = -inf*ones(1,11);

for policy = policies.'

    %%CALCULATE REWARD
    reward = [-15 [-4:4].*policy.' -15];

    %%CALCULATE TRANSITION MATRIX
    P = zeros(11,11);
    P(1,:) = [[1/2 1/2] zeros(1,9)];
    P(11,:) = [zeros(1,9) [1/2 1/2]];
    for i = [2:10]
        if policy(i-1) == -1
            P(i, i-1) = 3/4;
            P(i, i+1) = 1/4;
        else
            P(i, i-1) = 1/4;
            P(i, i+1) = 3/4;
        end
    end

    %%EVALUTATE POLICY
    V = ((eye(11) - lambda * P)\reward.').';
    if sum(V) > sum(V_max)
        V_max = V;
        opt_policy = policy.';
    end
end
```

VALUE ITERATION:

```
lambda = 0.98;
epsilon = 0.01;

V_last = epsilon * (1 - lambda) / (lambda)*ones(1,11);
V = zeros(1,11);
```

```
opt_policy = zeros(1,9);

while norm(V - V_last) > epsilon * (1 - lambda) / (2 * lambda)
    V_last = V;
    V(1) = -15 + lambda*(V_last(1)*1/2 + V_last(2)*1/2);
    V(11) = -15 + lambda*(V_last(11)*1/2 + V_last(10)*1/2);
    for state = [2:10]
        V_minus = -(state-6) + lambda*(3/4*V_last(state-1) + 1/4*V_last(state+1));
        V_plus = (state-6) + lambda*(1/4*V_last(state-1) + 3/4*V_last(state+1));
        if V_minus >= V_plus
            V(state) = V_minus;
        else
            V(state) = V_plus;
        end
    end
end

for state = [2:10]
        V_minus = -(state-6) + lambda*(3/4*V(state-1) + 1/4*V(state+1));
        V_plus = (state-6) + lambda*(1/4*V(state-1) + 3/4*V(state+1));
        if V_minus >= V_plus
            opt_policy(state-1) = -1;
        else
            opt_policy(state-1) = 1;
        end
end
```

POLICY ITERATION CODE:

```
opt_policy = ones(1,9);
policy_prev = zeros(1,9);

reward = [-15 [-4:4] -15];

while sum(opt_policy ~= policy_prev)>0
    policy_prev = opt_policy;
    reward = [-15 policy_prev.*[-4:4] -15];

    P = zeros(11,11);
    P(1,:) = [[1/2 1/2] zeros(1,9)];
    P(11,:) = [zeros(1,9) [1/2 1/2]];
    for i = [2:10]
        if policy_prev(i-1) == -1
            P(i, i-1) = 3/4;
            P(i, i+1) = 1/4;
        else
            P(i, i-1) = 1/4;
            P(i, i+1) = 3/4;
        end
    end

    V = (eye(11) - lambda*P)\(reward.');
    V_p = -inf;

    for policy = policies.'
        reward = [-15 (policy.').*[-4:4] -15];
        P = zeros(11,11);
        P(1,:) = [[1/2 1/2] zeros(1,9)];
        P(11,:) = [zeros(1,9) [1/2 1/2]];
        for i = [2:10]
            if policy(i-1) == -1
                P(i, i-1) = 3/4;
                P(i, i+1) = 1/4;
            else
                P(i, i-1) = 1/4;
                P(i, i+1) = 3/4;
            end
        end

        if sum(reward + lambda*(P*V).') > V_p
            new_policy = policy.';
            V_p = sum(reward + lambda*(P*V).');
        end
    end
    if sum(V_p) > sum(V)
        opt_policy = new_policy;
    end
end
```