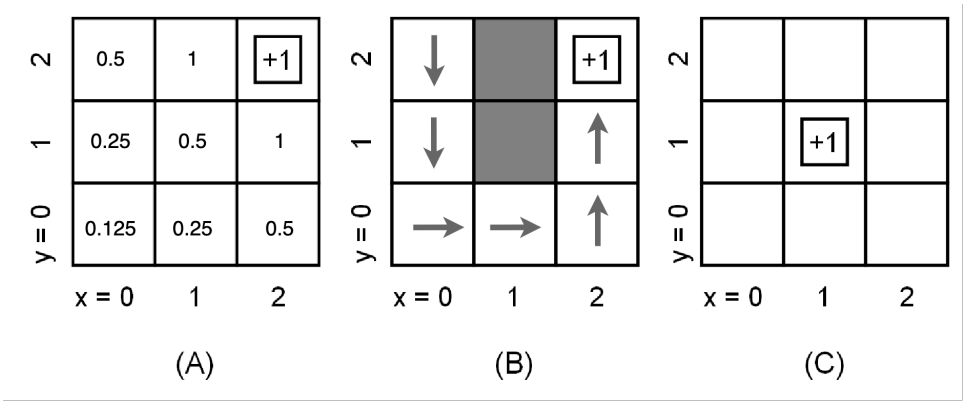
CS 6300 HW06: Functional Approximation Ryan Dalby Due March 22, 2022

Please use the LaTeX template to produce your writeups. See the Homework Assignments page on the class website for details. Hand in through gradescope.

1 Functional Approximation

For the following gridworld problems, the agent can take the actions N, S, E, W, which move the agent one square in the respective directions. There is no noise, so these actions always take the agent in the direction attempted, unless that direction would lead off the grid or into a blocked (grey) square, in which case the action does nothing. ~~The boxed +1 squares also permit the action X which causes the agent to exits the grid and enter the terminal state.~~ **Assumption: The boxed +1 squares are the terminal states.** The reward for all transitions are zero, except the exit transition, which has reward +1. Assume a discount of 0.5.



1. Fill in the optimal values for grid (A) (hint: this should require very little calculation).

   See grid (A) above.

2. Specify the optimal policy for grid (B) by placing an arrow in each empty square.

   See grid (B) above.

   Imagine we have a set of real-valued features $f_i(s)$ for each non-terminal state $s = (x, y)$, and we wish to approximate the optimal utility values $V^*(s)$ by $V(s) = \sum_i w_i \cdot f_i(s)$ (linear feature-based approximation).

3. If our features are $f_1(x, y) = x$ and $f_2(x, y) = y$, give values of w_1 and w_2 for which a one-step look-ahead policy extracted from V will be optimal in grid (A).

   This gives $V(s) = w_1 x + w_2 y$.

   Thus for each state $s = (x, y)$:

   - V((0,0)) = 0
   - V((1,0)) = w_1

- V((0,1)) = w_2
- V((2,0)) = $2w_1$
- V((1,1)) = $w_1 + w_2$
- V((0,2)) = $2w_2$
- V((2,1)) = $2w_1 + w_2$
- V((1,2)) = $w_1 + 2w_2$
- V((2,2)) = 0 (By definition of terminal state)

Weight values of $w_1 = w_2$ such that $0 < w_1 = w_2 < 1$ will give values that when extracted using one-step look-ahead policy extraction the policy is optimal in grid (A). (The upper bound enforces the constraint that in states (1,2) and (2,1) that $1 > 0.5(w_1 + w_2) = 0.5(2w_1) = 0.5(2w_2)$ so the action to go into the terminal state is actually chosen during policy extraction and the lower bound enforces that we actually proceed toward the goal when using policy extraction rather than away from it.)

(Note: the weight values given can give the optimal policy with a one-step look-ahead policy extraction, but do not necessarily provide and exhaustive bound of weight values that can give the optimal policy)

4. Can we represent the actual optimal values V^* for grid (A) using these two features? Why or why not?

   No. This can be seen by looking at the answer for the previous problem that enumerates the approximated values for each state $s = (x, y)$ for grid (A). Using the two features the approximated value for state (0,0) will always be 0 and cannot be the true optimal value of 0.125 regardless of values of w_1 and w_2.

5. For each of the feature sets listed below, state which (if any) of the grid MDPs above can be 'solved', in the sense that we can express some (possibly non-optimal) values which produce optimal one-step look-ahead policies.

   (Note: the weight values given can give the optimal policy with a one-step look-ahead policy extraction, but do not necessarily provide and exhaustive bound of weight values that can give the optimal policy)

   (a) $f_1(x, y) = x$ and $f_2(x, y) = y$.
   Grid (A) can be "solved" with $0 < w_1 = w_2 < 1$.
   Using information from 3 for the approximate value of each state.

   (b) For each (i, j), a feature $f_{i,j}(x, y) = 1$ if $(x, y) = (i, j)$, 0 otherwise.
   Grid (A), grid (B), grid (C) can be "solved" with $w_{i,j} = V^*((i, j))$.
   This is because each grid square value is approximated as the corresponding $w_{i,j}$ thus if the weight takes on the optimal value itself then the optimal policy will be produced using a one-step look-ahead policy. (It is likely other non-optimal values can also produce the optimal policy but this illustrates for all grids how this feature set can "solve" the MDP grids).
   For each state $s = (x, y)$ of solvable grids:
   - V((0,0)) = $w_{0,0}$
   - V((1,0)) = $w_{1,0}$
   - V((0,1)) = $w_{0,1}$
   - V((2,0)) = $w_{2,0}$
   - V((1,1)) = $w_{1,1}$ (Note: for grid (C) (1,1) = 0 by definition of terminal state)
   - V((0,2)) = $w_{0,2}$
   - V((2,1)) = $w_{2,1}$
   - V((1,2)) = $w_{1,2}$
   - V((2,2)) = $w_{2,2}$ (Note: for grid (A) and grid (B) (2,2) = 0 by definition of terminal state)

   (c) $f_1(x, y) = (x - 1)^2$, $f_2(x, y) = (y - 1)^2$, and $f_3(x, y) = 1$.
   Grid (C) can be "solved" with $0.5(w_1+w_2+w_3) < 1, 0.5(w_1+w_3) < 1, 0.5(w_2+w_3) < 1$ and either $w_1 < 0$ or $w_2 < 0$,.
   For each state $s = (x, y)$ of solvable grids:
   - V((0,0)) = $w_1 + w_2 + w_3$
   - V((1,0)) = $w_2 + w_3$
   - V((0,1)) = $w_1 + w_3$
   - V((2,0)) = $w_1 + w_2 + w_3$
   - V((1,1)) = 0
   - V((0,2)) = $w_1 + w_2 + w_3$
   - V((2,1)) = $w_1 + w_3$

- V((1,2)) = $w_2 + w_3$
- V((2,2)) = $w_1 + w_2 + w_3$