Lab Report A7
CS4300
Derek Heldt-Werle 1,3,5
Matthew Lemon 2,4,6

12/1/16

1. Introduction

For this lab we will create a function which examines the value iteration algorithm. The agent will learn a policy for the following Wumpus world:

0	0	0	G
0	0	W	P
0	0	P	0
0	0	0	0

For this lab, the agent will be able to take the following actions:
A = {UP, LEFT, DOWN, RIGHT}

where these are movements with probabilistic outcomes as described in Peter Norvig and Stuart J. Russell's book Artificial Intelligence: A Modern Approach (i.e., 0.8 probability of going the direction selected, 0.1 of going to either side). We will thus develop a transition probability table for the 16 cells for the 4 actions. We will implement the algorithm based around the value iteration algorithm found on page 653 of Peter Norvig and Stuart J. Russell's book Artificial Intelligence: A Modern Approach. We will then also implement a policy generator based around the value iteration algorithm. We will look to answer the following questions in this lab:

Does increasing the value of gamma closer to 1 have any relation to the actual results when dealing with the 3x4 board?

At what value of R(s) does the agent begin to consistently make decisions that will lead to a score of -1000?

2. Method

CS4300_A7_runner:
We created a runner function in which we hard coded the probabilities of the two different boards, the Wumpus World board and the 3x4 board. We would then call the

CS4300_MDP_value_iteration function to produce the utilities and then used those utilities when calling CS4300_MDP_policy to produce an optimal policy.

CS4300_MDP_value_iteration:
We followed the algorithm outlined in the book and presented in lecture to create the MDP value iteration algorithm. We would loop over all of the states updating their utilities using the function:

$$U'(s') = R(s') + \gamma \; max \; a \in A(s) \; \sum P(s' \mid s, \; a) \; U(s')$$

Until there was sufficiently small change in the utilities produced.

CS4300_MDP_policy:
Using the utilities produced by the value iteration algorithm and the probabilities we calculated the optimal MDP policy. For each state the optimal policy was produced using the equation:

$$Policy(s') = max \; a \in A(s) \; \sum P(s' \mid s, \; a) \; U(s')$$

3. Verification of Program

0.8116	0.8678	0.9178	1.0000
0.7616	0	0.6603	-1.0000
0.7053	0.6553	0.6114	0.3879

Table 1: Results of Running Value Iteration on 4x3 World with ɣ=.99999

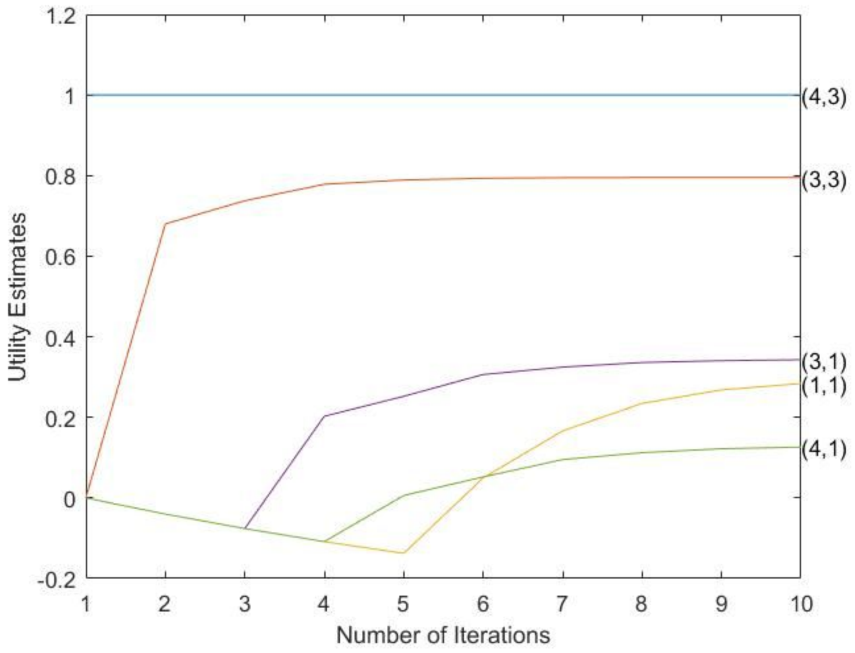

Figure 1: Results of Running Value Iteration on 4x3 World with ɣ=.9

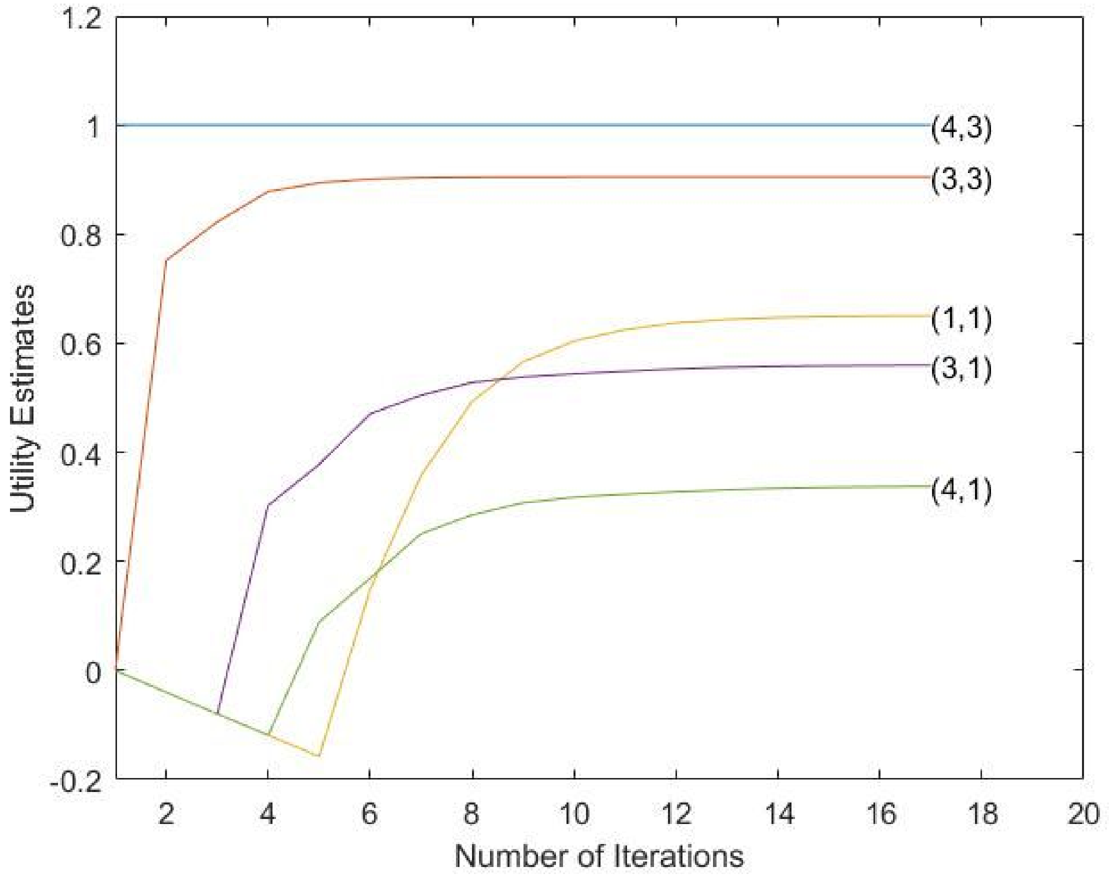

Figure 2: Results of Running Value Iteration on 4x3 World with ɣ=.99

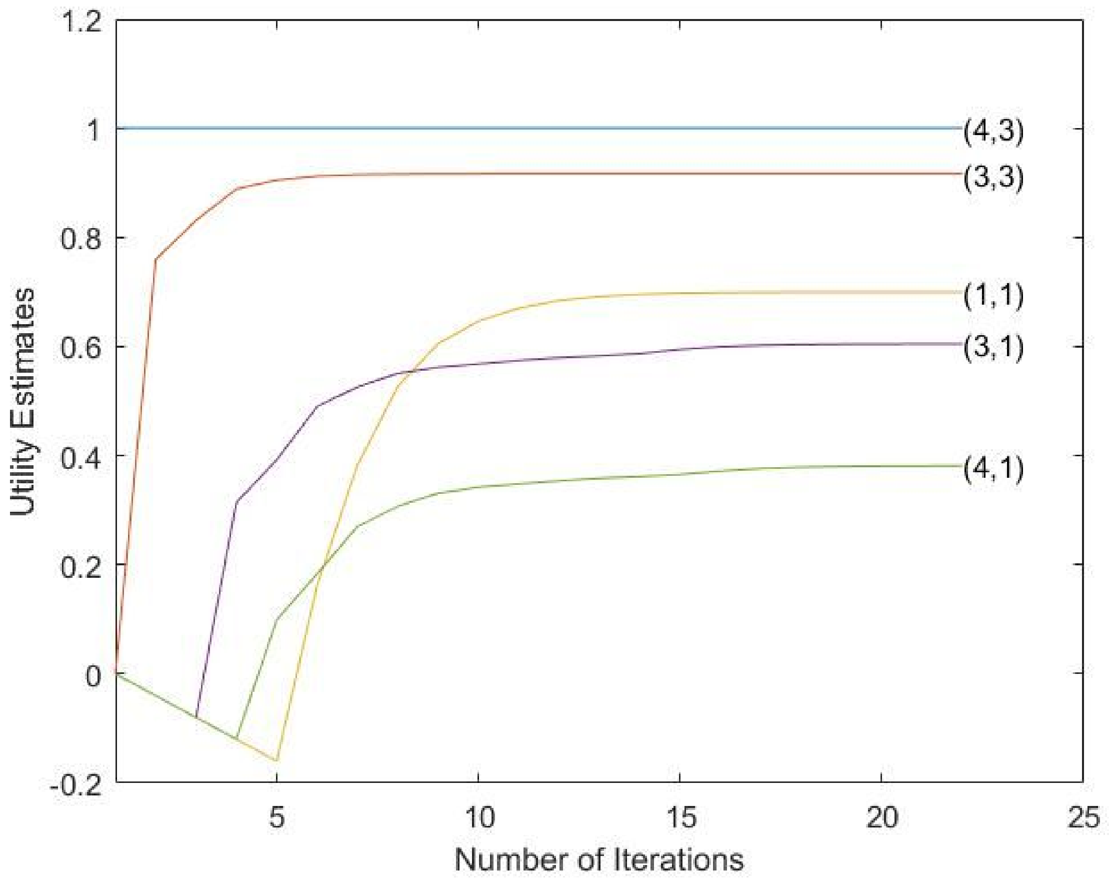

Figure 3: Results of Running Value Iteration on 4x3 World with ɣ=.999

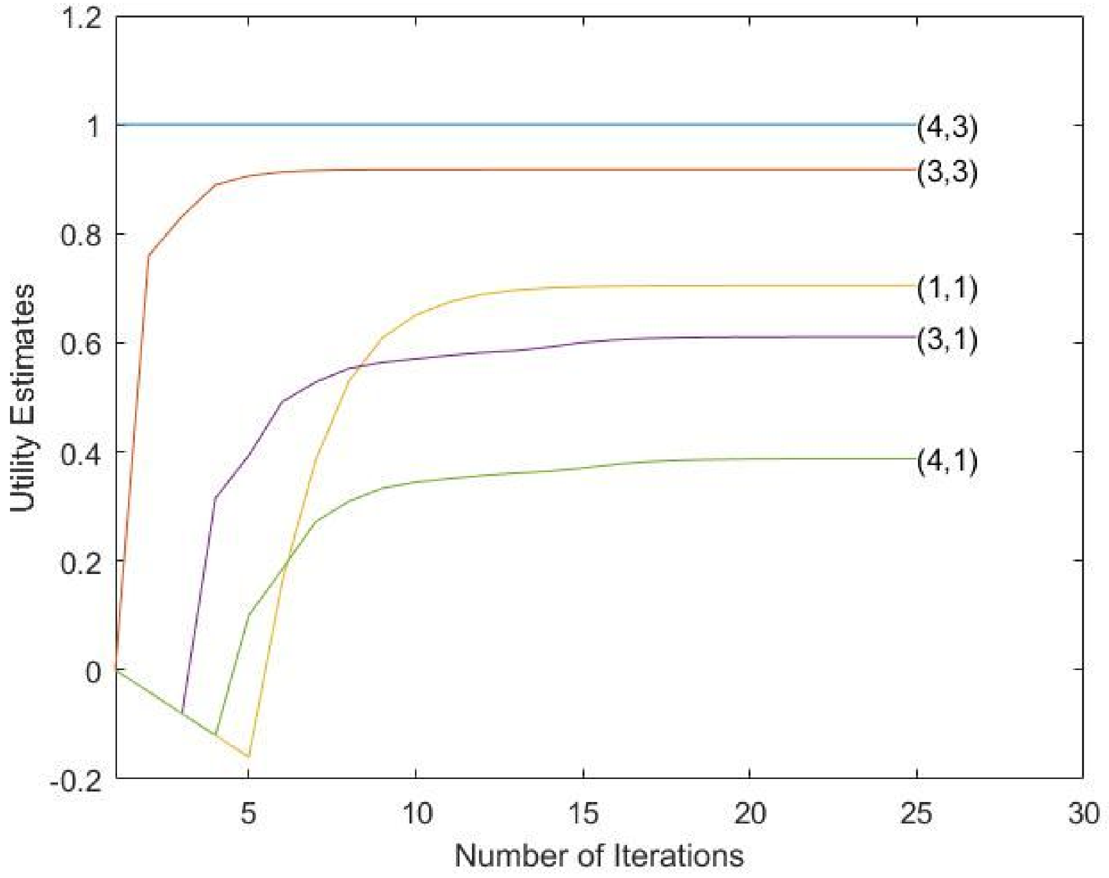

Figure 4: Results of Running Value Iteration on 4x3 World with ɤ=.9999

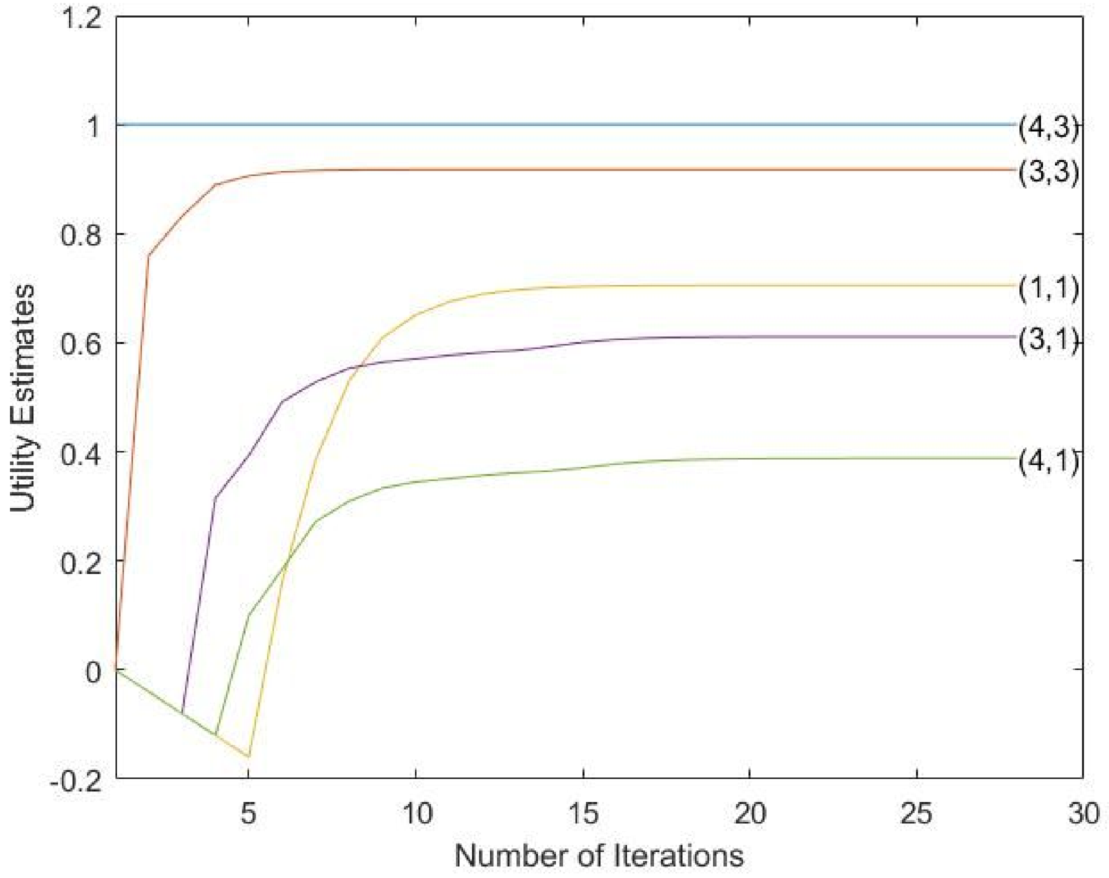

Figure 5: Results of Running Value Iteration on 4x3 World with ɣ=.99999

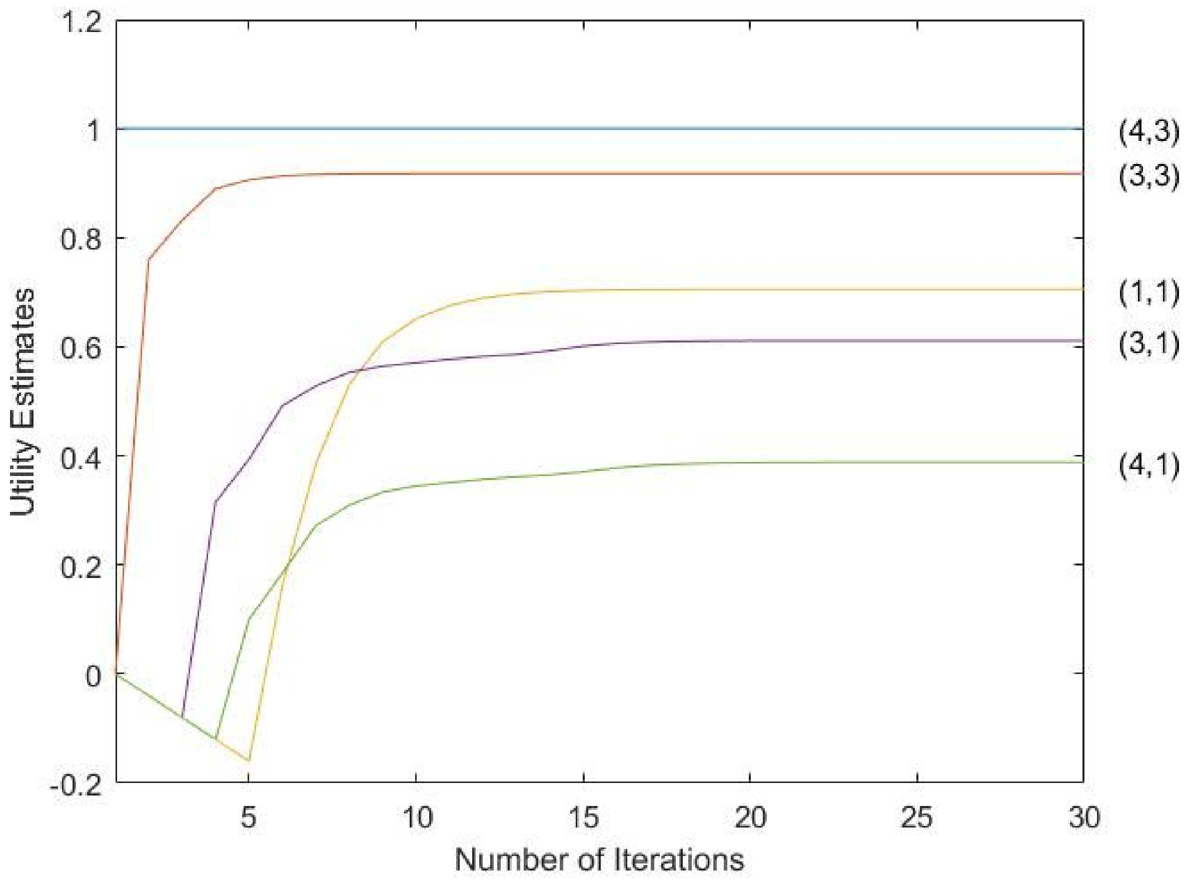

Figure 6: Results of Running Value Iteration on 4x3 World with ɣ=.999999

4. Data and Analysis

→	→	↑	1000
↑	←	-1000	-1000
↑	←	-1000	↓
↑	←	↓	↓

Table 2: Optimal policy for stochastic environment with R(s) = -1

→	→	→	1000
↑	↑	-1000	-1000
↑	↑	-1000	↓
↑	↑	←	←

Table 3: Optimal policy for stochastic environment with R(s) = -100

→	→	→	1000
↑	↑	-1000	-1000
→	→	-1000	↑
↑	↑	↑	↑

Table 4: Optimal policy for stochastic environment with R(s) = -500

↑	←	↑	1000
↑	←	-1000	-1000
↑	←	-1000	↓
↑	←	↓	↓

Table 5: Optimal policy for stochastic environment with R(s) = 1

5. Interpretation

As seen in the above tables and graphs, as gamma became increasingly close to 1, there existed a strong relation in having the proper values returned when dealing with the 3x4 board. This makes logical sense as increasing this value leads the algorithm to become increasingly cautious. When cautious, the algorithm will take any necessary steps in order to avoid any chances of failure. On the flip side, as gamma is decreased there is invariably a higher chance at which the agent will make a poor decision as it doesn't iterate enough times in order to evaluate the decisions it makes before potentially making a fatal one. Through testing we found, and as seen in table 4, it wasn't until the rewards reached a value of about -500 that the algorithm would begin to consistently fall to one's death, or run into the wumpus.

6. Critique

Through this assignment we were able to get an introduction to MDP value iteration and policy generation. We were able to learn about utilities and how they work with probabilities to develop a policy to successfully navigate a given world with an agent who might not always do what we want. It was interesting to be able to see how the differences in rewards also affect the policies we generate and that in some situations it’s better to just run for the closest pit or wumpus.

7. Log

Author Matthew Lemon & Derek Heldt-Werle
Coding Portion (Worked together): 12
Report (Derek): 2
Report (Matt): 2