Reinforcement Learning

Assignment 2

Efstathios Lempesis
s1433525

2015-2016

1)As we know the value of a state is defined as the value of the best action for this state based on the policy that is currently considered optimal. We can also have the discounted version for the value of a state and the discount parameter γ introduces the time horizon with length (using the equation from the slides of the third lecture at slide 10) $\frac{1}{1-\gamma}$. γ is a discount parameter and using this parameter we try to select actions so that the sum of the discounted rewards we receive over the future is maximized. If γ is closer to zero, we tend to consider only immediate rewards. If γ is closer to one, we consider future rewards with greater weight, willing to delay the reward. The time horizon depends on the problem. For our problem I think that a good choice for time horizon is 4 future rewards. I chose the number 4 because the agent navigates inside a 5x5 rectangular grid and it needs 4 steps to travel from the center of the 5x5 grid to a corner of this grid. For any other position in this grid it needs fewer steps to go to a corner. So, I think that 4 future rewards for time horizon is adequate and good choice for our problem, since the agent doing a step it receives reward considering the 4 future states inside a 5x5 rectangular grid. More future states would be redundancy.

Using the time horizon we can compute the γ which we'll use for the experiments of this assignment. The time horizon equals to $\frac{1}{1-\gamma}$. So, for 5x5 grids I chose 4 future rewards as the time horizon for our problem and our γ is:

$$\frac{1}{1-\gamma}=4 \Leftrightarrow 4-4\gamma=1 \Leftrightarrow 4\gamma=3 \Leftrightarrow \gamma=0.75$$

To model our problem, for 5x5 grids I use 25 states. I concluded to this number of states because the agent can be at any square on the grid and then it goes to the closest goal location. So, I should train our problem for any initial position(out of the 25 possible) of the agent on the grid.

To find the mean (immediate) reward for a good policy, I'll show what is the reward for every state in the 5x5 grid. As a good policy I consider a policy with which the agent directly goes to the closest goal location without doing redundant steps on the grid.

For the case where the passenger is not initially in the taxi we have:

	1	2	3	4	5
1	1	1	0.5	1	1
2	1	0.5	0.33	0.5	1
3	0.5	0.33	0.25	0.33	0.5
4	1	0.5	0.33	0.5	1
5	1	1	0.5	1	1

I compute the immediate rewards as $\frac{1}{steps}$ where steps are the steps that the agent does to go the nearest goal location.

If we summarize all the elements of this matrix and then we divide the sum by 25, which is the number of all the states of the problem, we get the mean (immediate) reward of the good policy equals to 0,7028.

We can compute the maximal value of a state-action pair using the equation (from the slides of the

forth lecture at slide 8) $\frac{r_{max}}{1-\gamma}$. In our problem the maximum immediate reward equals to 1. We have this reward in the case when the agent is next to or on the goal location.. This is our maximum reward because for any other case the agent has to do more than one step to reach the goal location and as a result the immediate reward is < $\frac{10}{1}$, because the denominator, which is the number of steps, is larger than 1. Then, we can compute our maximal value of a state-action pair:

$$\frac{r_{max}}{1-\gamma}=\frac{1}{1-0.75}=\frac{1}{0.25}=4$$

So, 4 is the upper bound for the values of the states of our problem for the 5x5 grid.

To define the number of trials that a standard algorithm needs in order to find a good solution depends on the exploration and it's general difficult to define to predict the number of trials. For example, if we do few steps at every trial we'll need more trials to find a good solution because we don't explore all the world. Nevertheless, if we do few steps we can do more trials and we starts a lot of times from different initial states and it is possible to find good states more often instead of starting from a state and doing a lot of steps near a single initial state. In dynamic programming to find an optimal policy we need polynomial time in the number of states and actions. In our case we have 25*5=125 number of states and actions and I'll use the simple case where we have polynomial of order 1 (constant*x) and I'll use 13*125=1625 trials for the case of the 5x5 grid.

For the cases of the 10x10 and 25x25 we can apply the same reasoning to compute the γ or the mean immediate reward but with values that are suitable for the grid 10x10 and 25x25. For example for the 10x10 grid I will use time horizon 7 because that is the number of steps that the agent has to do to go from the center of the 10x10 grid to the nearest goal location.

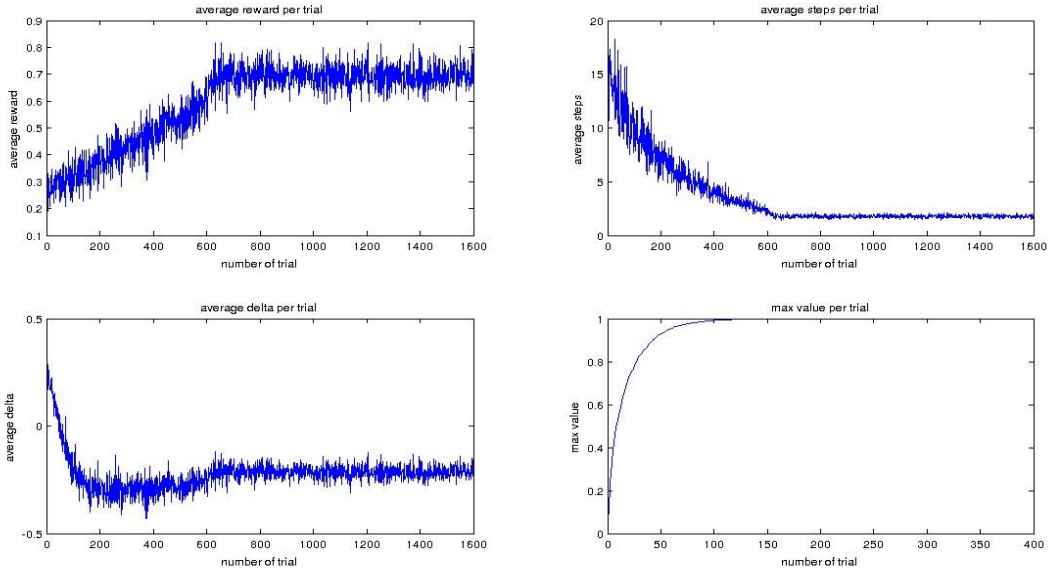
Our problem is deterministic because taking an action we are sure what is the next state and the reward. So, in deterministic worlds like our problem for every state-action pair there is a unique future state and unique reward. It is obvious that in our problem we have a deterministic policy that in every state we take the action with the highest value. So, there is a (deterministic) policy which maps states to actions. In contrast to the stochastic problem taking an action we are not sure what is the next state, there is uncertainty and in every state an action is chosen from a distribution. If our problem was non-deterministic we would have to redefine our estimations by taking expected values.

2) For this question I implemented a linear, gradient-descent version of the Q-learning algorithm with binary features φ(x) and ε-greedy policy[1]. Since, in this case we have a 5x5 grid and we want to have for each square one of the 25 indicator functions to be 1 and all the others to be 0, I created a 25x5 matrix, 25 is the number of states and 5 is the number of actions, where I store the values of the action-dependent parameters θ which carries the information about the estimation of the value function. So, for each action we have a weight-parameter for each one of the 25 possible states and $\theta^T\varphi(x)$ is constant over the individual regions, where in this case we have 25 regions.

I ran 50 experiments where each experiment had 1600 trials and each trial had 625 steps. I used the parameters γ=0.75, η=0.25 and initial ε=1.0. I kept the γ and η fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{625}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

The plot of the average reward, number of steps, delta and max values for the 50 experiments is the following. For the plot of the max value, I only present the max values of the first 400 trials so that

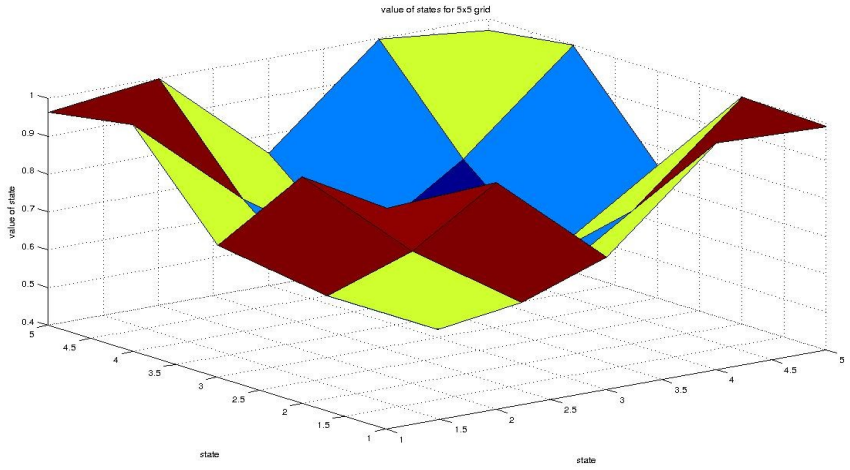
the values of the first steps to be more clear:



From these plots we can see that till the 650 trials the agent is learning how to go to the goal locations, because the average reward and the max value is increasing and the delta and the number of steps is decreasing. After the 650 trials we can notice that the algorithm has stopped to increase its performance and it's stable. Also, we take reasonable values for all the experiments since the average reward is always lower than the maximum reward which is 1 and the delta at the start is positive and it's decreasing and then it's always negative.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 5x5 grid. In order to find the value of each state, for each action I set the feature, which corresponds to each state, to the value 1 and then I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of feature indices. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 5x5 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 3,3

Best action is north

Next state of agent is 3,2 with value 0.421875

Best action is west

Next state of agent is 2,2 with value 0.562500

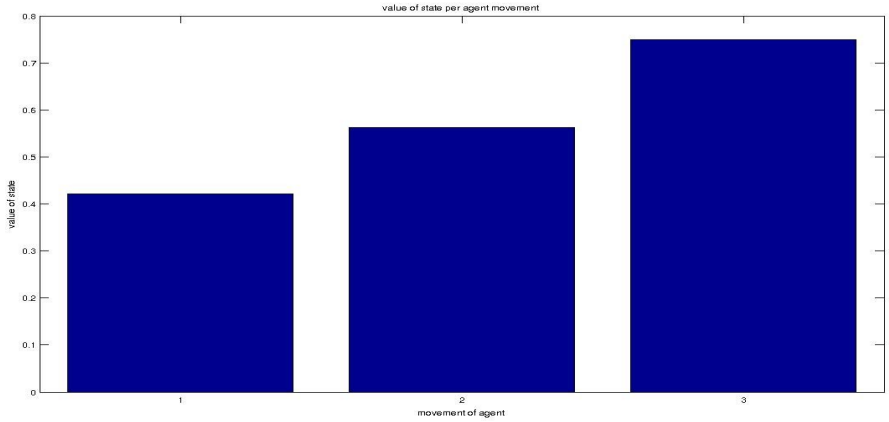
Best action is north

Next state of agent is 2,1 with value 0.750000

Best action is west

Goal

The plot of the values of this experiment is:

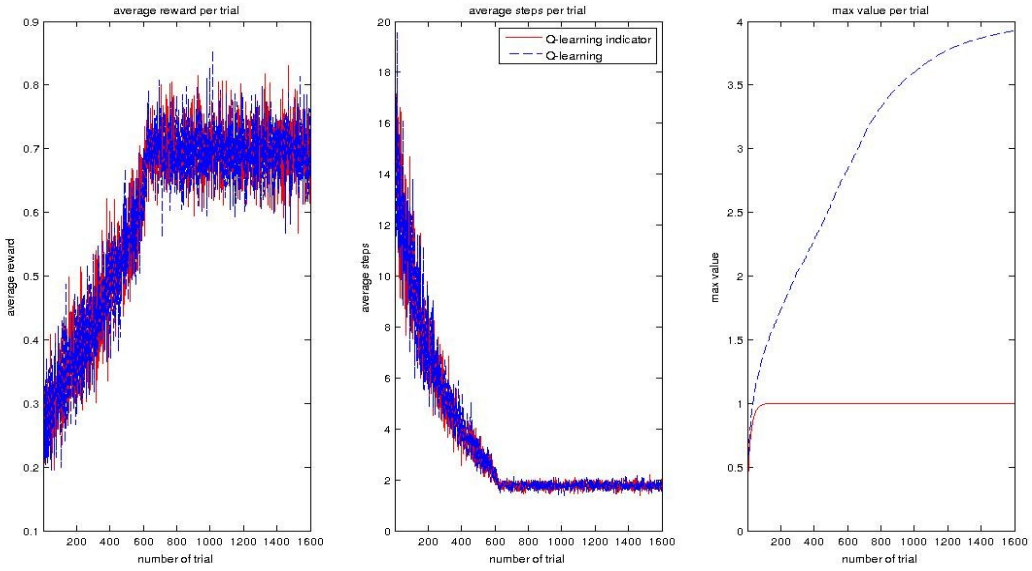


From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

From the way we implemented the above algorithm we can conclude that it's exactly the same with the default lookup table Q-learning algorithm. At the lookup table version we have the Q matrix where for each state we store the weights which correspond to the actions and these weights illustrate how good is an action for each state. The higher the weight of an action for a specific state, the better is the action for this state. So, if we have N states and M actions the Q matrix is NxM. In the case of the 25 indicator functions and the 5x5 grid we have 5 actions and each function has 25 features which each feature corresponds to one of the 25 states. So, for each action we have a vector θ which has 25 dimensions and each element of θ indicates how good is the action for the corresponding state. Therefore, for the lookup table we have the 25x5 Q matrix and for the indicator functions we have the 5x25 θ matrix. Both of these matrices are the outcome of the training procedure and contain the weights which show how good is an action for each state and every action without exception has a weight to every state of the 25 possible states. So, the 5x5 grid with the 25 indicator functions is an exact reproduction of the lookup table version, but in one case we have the 5x25 θ matrix and in the other case the 25x5 Q matrix.

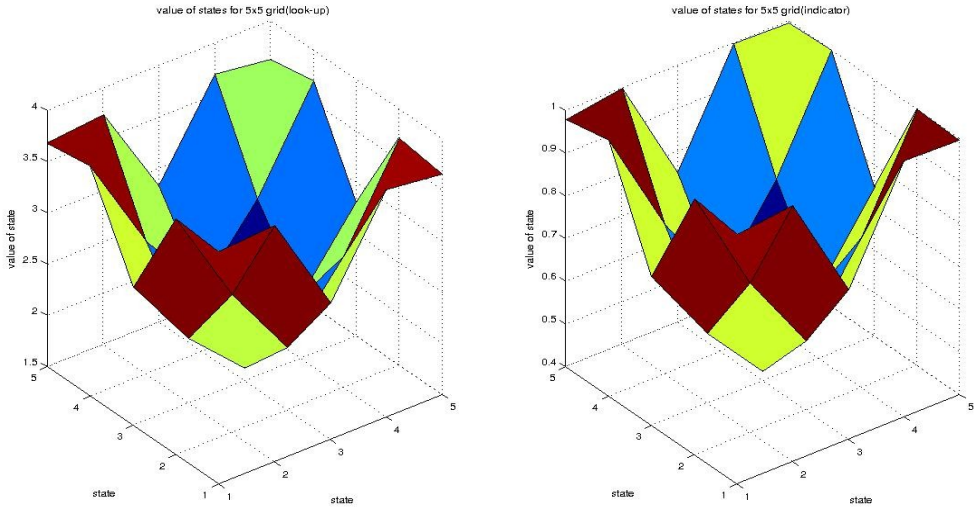
Now let's compare value of the states between the lookup table version and the indicator functions version. For this comparison I ran 50 experiments where each experiment had 1600 trial and each trial had 625 steps. I used the parameters γ=0.75, η=0.25 and initial ε=1.0. I kept the γ and η fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{625}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

The plot for the comparison between the 2 versions is:



From these plots we can observe that both the lookup table version and the indicator functions version have exactly the same behavior for the average reward and the average steps per trial. But, for the max value we can notice that there is difference between these 2 approaches. For the indicator functions the max value is increasing but near the 100 trial it reaches the value 1 and it's stable for all the remaining trials. On the other hand for the lookup table version the max value is increasing all the time and it reaches the value 4. We have this difference because at the indicator functions approach the θ matrix is updated by adding the value η*delta where delta can take negative values but at the lookup table approach the Q matrix is updated by value that are always positive. So, it's reasonable that the lookup table version can give larger max values.

Now let's compare value of the states between the lookup table and the indicator function.



From this plot we can conclude that both the 2 versions make the same evaluation of the states. They give large values to the same states and low values to the same states as well.

So. all the above simulations confirm that the 5x5 grid with the 25 indicator functions is an exact reproduction of the lookup table version, but in one case we have the 5x25 θ matrix and in the other case the 25x5 Q matrix.

As we know the lookup table version of the Q learning is limited to tasks with a small number of states and when we have a task with a large number of states, we have problem with the memory to store the large tables and with the time and data to fit them accurately. Also, we have problem with the exploration and convergence time. In this question where we have a small number of states and we have as many approximation functions as the number of states, we can observe the effect of the approximation function since we don't have generalization.

3) In this question we have a 10x10 grid and we still have 25 indicator functions. So, the indicator functions are distributed on the grid like this.

	1	2	3	4	5	6	7	8	9	10
1	φ_1	φ_1	φ_2	φ_2	φ_3	φ_3	φ_4	φ_4	φ_5	φ_5
2	φ_1	φ_1	φ_2	φ_2	φ_3	φ_3	φ_4	φ_4	φ_5	φ_5
3	φ_6	φ_6	φ_7	φ_7	φ_8	φ_8	φ_9	φ_9	φ_{10}	φ_{10}
4	φ_6	φ_6	φ_7	φ_7	φ_8	φ_8	φ_9	φ_9	φ_{10}	φ_{10}
5	φ_{11}	φ_{11}	φ_{12}	φ_{12}	φ_{13}	φ_{13}	φ_{14}	φ_{14}	φ_{15}	φ_{15}
6	φ_{11}	φ_{11}	φ_{12}	φ_{12}	φ_{13}	φ_{13}	φ_{14}	φ_{14}	φ_{15}	φ_{15}
7	φ_{16}	φ_{16}	φ_{17}	φ_{17}	φ_{18}	φ_{18}	φ_{19}	φ_{19}	φ_{20}	φ_{20}
8	φ_{16}	φ_{16}	φ_{17}	φ_{17}	φ_{18}	φ_{18}	φ_{19}	φ_{19}	φ_{20}	φ_{20}
9	φ_{21}	φ_{21}	φ_{22}	φ_{22}	φ_{23}	φ_{23}	φ_{24}	φ_{24}	φ_{25}	φ_{25}
10	φ_{21}	φ_{21}	φ_{22}	φ_{22}	φ_{23}	φ_{23}	φ_{24}	φ_{24}	φ_{25}	φ_{25}

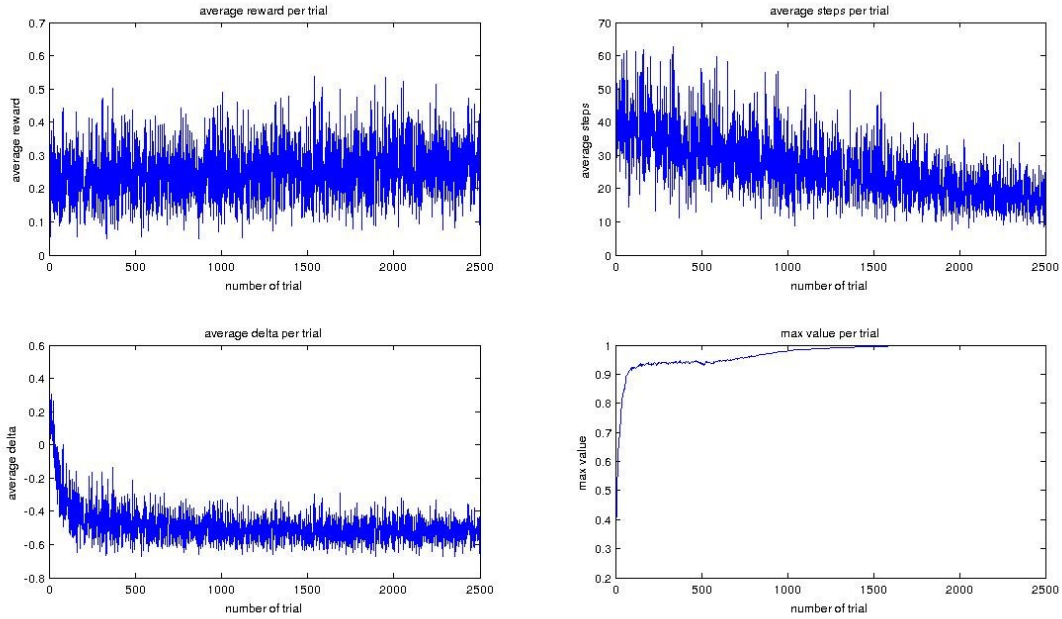
From the above grid we notice that every indicator function is responsible for 4 squares. For example, the indicator function 8 is responsible for the squares (3,5), (3,6), (4,5) and (4,6). In other words the states (3,5), (3,6), (4,5) and (4,6) have the same indicator function. Compared to question 2 where we had 1 indicator function per state, now we have 100 states but we only have 25 indicator functions. That means through the training procedure we'll try to fit weights θ which each one will correspond to 4 states. So, we'll have a 5x25 θ matrix, where the 5 is the number of actions and 25 is the number of indicator functions. Thus, the goal is that these 25 indicator functions should generalize to all the 100 states.

The 4 corners of the 10x10 grid are now the goal locations. Each goal location instead of a square is now 4 squares. So, we have the goal locations [(1,1)(1,2)(2,1)(2,2)], [(1,9)(1,10)(2,9)(2,10)], [(9,1)(9,2)(10,1)(10,2)] and [(9,9)(9,10)(10,9)(10,10)]. We have this new definition of the goal locations because the indicator functions of the corner-squares 1, 10, 91 and 100 belong to the indicator functions φ_1, φ_5, φ_{21} and φ_{25} respectively. When for example the agent is on the square (9,2) this square must be a goal location because all the squares [(9,1)(9,2)(10,1)(10,2)] have the same policy belonging to the indicator function φ_{21}. So, since the square (10,1) a corner of the grid and it must be a goal location, all the other squares of the φ_{21} [(9,1)(9,2)(10,2)] should also be goal locations.

Here changing the θ of one basis function changes the estimated value of 4 states. So, this change generalize to affect the value of many states and we expect to find an approximation that balances the errors in different states. In our case we have squares and this shape of the features determines the nature of generalization.

I ran 20 experiments where each experiment had 1600 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25 and initial ε=1.0. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

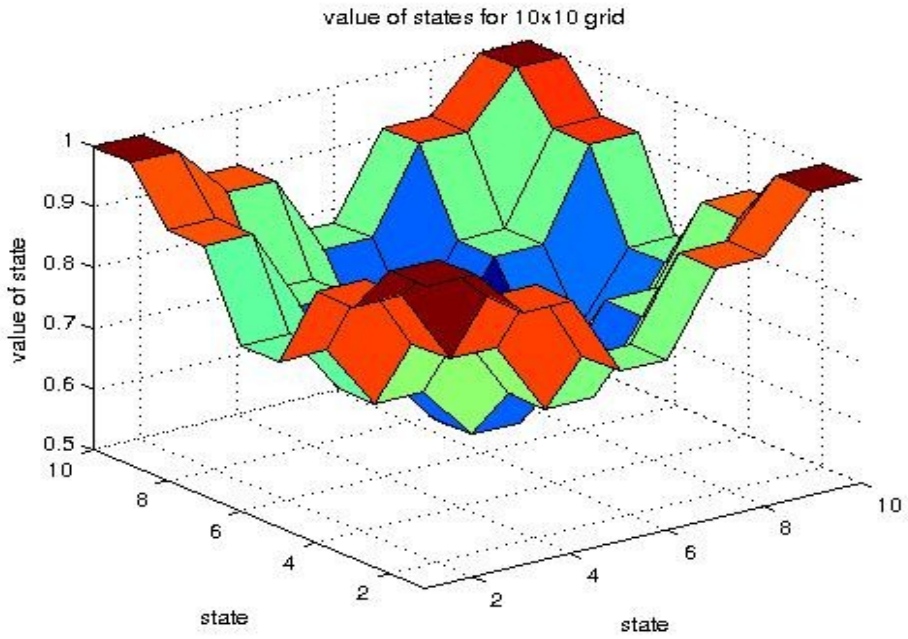
The plot of the average reward, number of steps, delta and max values for the 20 experiments is the following:



From these plots we can see that the average number of steps and the average delta decrease as the number of trials increases and the max value after the trial 1500 reach the value 1 and this agrees with the previous question where we had the 5x5 grid. So, the agent learns to go to the goal locations. Nevertheless, in the case of the 10x10 grid the increase of the average reward is not so clear and looks stable and 'noisy' compared to the 5x5 grid.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 10x10 grid. In order to find the value of each state, for each action I set the feature, which corresponds to each state, to the value 1 and then I compute θ*F, where θ is the 25x5 matrix and F, for every possible action a, is the set of feature indices. Now, we should notice that each element of θ correspond to 4 states out of the 100 possible states. For example, the θ_{17} of the φ_{17} corresponds to the states 63, 64, 73 and 74. So, all these states have the same value because they belong to the same indicator function and they have the same weight θ. As a value of a state I take the max value of an action for each state.

The plot of the values of the 10x10 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 5,7

Best action is south

Next state of agent is 5,8 with value 0.681357

Best action is south

Next state of agent is 5,9 with value 0.681357

Best action is east

Next state of agent is 6,9 with value 0.757357

Best action is east

Next state of agent is 7,9 with value 0.757357

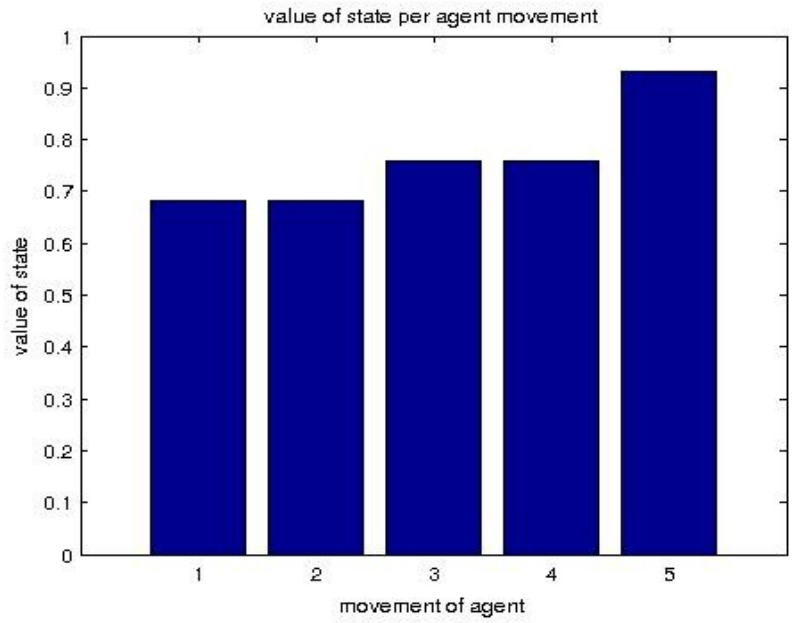
Best action is east

Next state of agent is 8,9 with value 0.930426

Best action is east

Goal

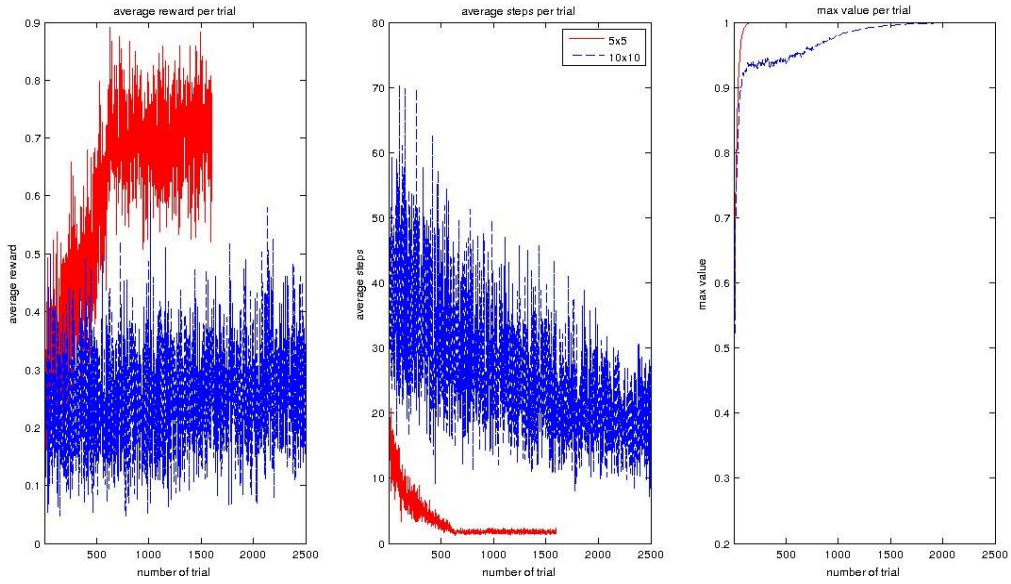
The plot of the values of this experiment is:



From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

Now let's notice the difference between the 5x5 grid and the 10x10 grid concerning the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments. I ran 20 experiments and for the 5x5 grid I ran 1600 trials and 625 steps per trial and for the 10x10 grid I ran 2500 trials and 10000 steps per trial. Also, I used the parameters γ=0.75 for the 5x5 grid and γ=0.9 for the 10x10 grid, η=0.25 and initial ε=1.0 for both the 5x5 and 10x10 grid. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{625})^{\frac{1}{2}}$, where t is the number of trial.

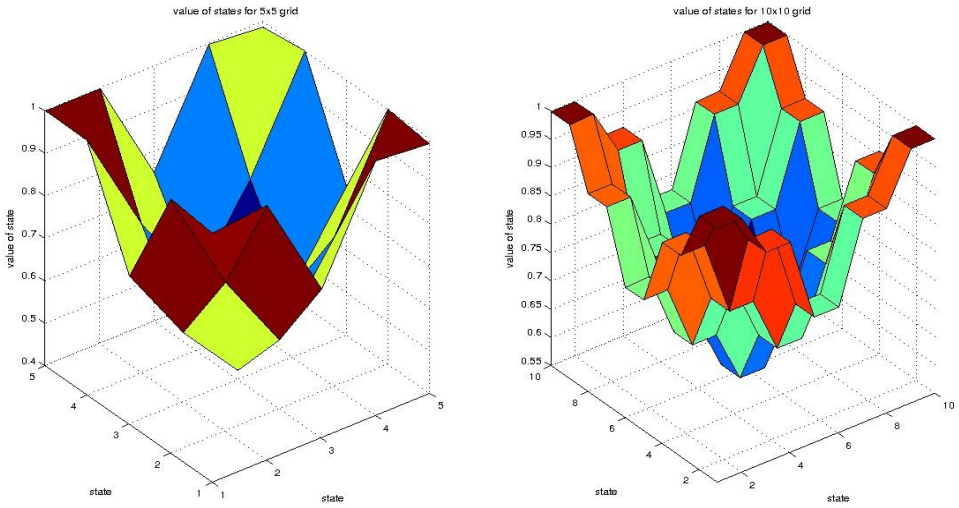
The plot for the comparison between the 5x5 and the 10x10 is:



From the above plots we can notice the 5x5 grid gives larger average reward compared to the 10x10 grid because the 5x5 grid is smaller than the 10x10 grid and the agent does fewer steps to go the

goal locations and as a consequence it takes larger rewards. We define the reward as $\frac{1}{number\ of\ steps}$. Also, the 5x5 grid gives smaller average steps compared to the 10x10 grid because the grid is smaller. An other difference between these sizes of grids is at the max value. We can notice that the max value of the 10x10 grid till the 2000 trial is lower than the max value of the 5x5 grid. As we know we have binary features so in order to compute the max value we only take the θ values of the states. In the case of the 5x5 grid we have a θ value per state but in the case of the 10x10 grid we have a θ value for 4 states and as a consequence we have generalization compared to the 5x5 grid. Because of this generalization is more difficult for the algorithm to converge and be trained and this makes the algorithm to have smaller θ-weights compared to the 5x5 grid. So, the 5x5 grid can be trained faster and have larger weights and max values in earlier trials than the 10x10 grid.

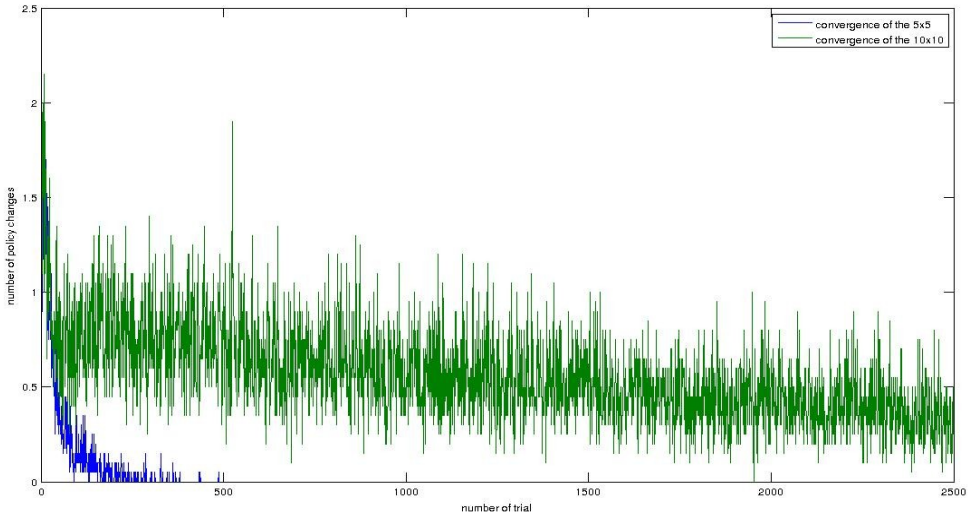
We should also observe the values of the states for the 5x5 and 10x10 grids.



We notice that both the surfaces have same shape and that means we have the same areas on the grids that good locations or bad locations. The difference on these surfaces are that in the case of the 10x10 grid because we have generalization we have little squares(2x2 areas on the grid) on the surface that have the same values. But, in the case of the 5x5 grid every square of the grid has its own value.

In order to observe how the converge time changes from a 5x5 grid to a 10x10 grid, we'll check when the actions don't change any more. To find this convergence I keep in a 25-dimensional vector the index of the action that has the maximum theta per state. In other words, for every state(out of the 25 possible states) I keep the index of the best action. So, per trial I count in this 25-dimensional vector how many actions have changed compared to the previous trial. As a result, when the counter is zero, we can conclude that the policy has converged and the agent know what is the best action to do at every state. I ran 20 experiments and I calculated the average of the policy changes of these 20 experiments.

I plot the convergence time for the 5x5 and the 10x10 grid.



From these plots we can notice that in the case of the 5x5 grid at around the 500 trials the policy has converged because there isn't any change at the best action of the states after the 500 trial. But, in the case of the 10x10 grid we can notice that till the 2500 trial the policy hasn't converged. From these observations we can conclude that because in the case of the 10x10 grid we use 25 states for all the 100 squares, it's difficult for our problem and the learning procedure to converge. The necessary generalization leads our problem of the 10x10 grid to be more difficult to converge.

4) Before we start this question we should point out that the binary features are advantageous from a computational point of view because per action we don't summarize all the features but only the feature that correspond to the specific states that the agent is. Here we'll use basis functions that and for each action we'll summarize all the features.

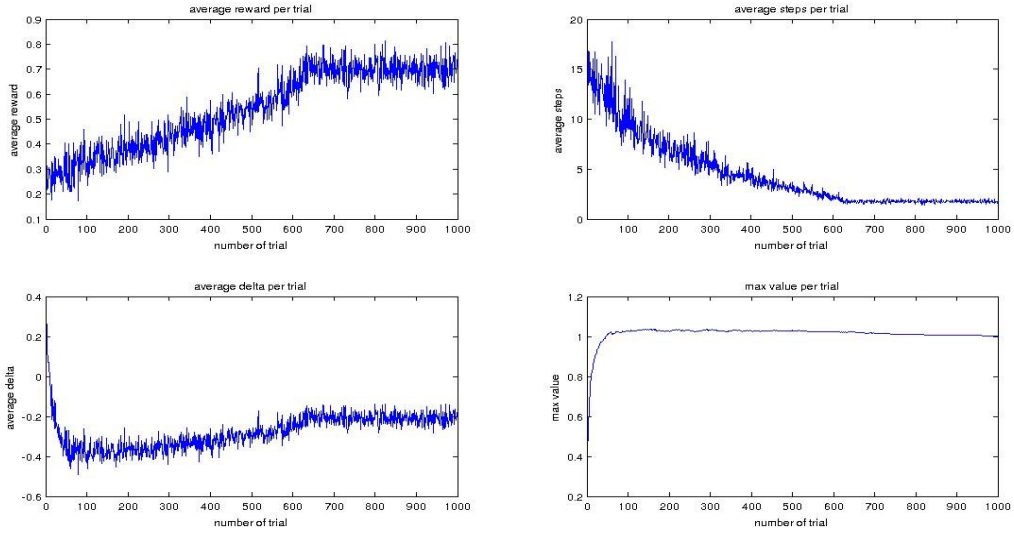
For this question we'll use 25 two-dimensional Gaussian bell-shaped functions as basis functions for the representation of the value function with regularly spaced centers. At the following grid we can see the centers of the radial basis functions for the case of the 5x5 grid and 25 basis functions. We'll also use these centers for the agent to identify where is our agent in each step.

	1	2	3	4	5
1	(0.5,0.5) φ_1	(1.5,0.5) φ_2	(2.5,0.5) φ_3	(3.5,0.5) φ_4	(4.5,0.5) φ_5
2	(0.5,1.5) φ_6	(1.5,1.5) φ_7	(2.5,1.5) φ_8	(3.5,1.5) φ_9	(4.5,1.5) φ_{10}
3	(0.5,2.5) φ_{11}	(1.5,2.5) φ_{12}	(2.5,2.5) φ_{13}	(3.5,2.5) φ_{14}	(4.5,2.5) φ_{15}
4	(0.5,3.5) φ_{16}	(1.5,3.5) φ_{17}	(2.5,3.5) φ_{18}	(3.5,3.5) φ_{19}	(4.5,3.5) φ_{20}
5	(0.5,4.5) φ_{21}	(1.5,4.5) φ_{22}	(2.5,4.5) φ_{23}	(3.5,4.5) φ_{24}	(4.5,4.5) φ_{25}

For example, when the agent is at the state (2,4), so the agent's position is (1.5,3.5), the value of the basis function φ_{23} is $\exp\left(\frac{-\|[1.5,3.5]-[2.5,4.5]\|^2}{2\sigma^2}\right)=0.2$ and the value of the basis function φ_{10} is $\exp\left(\frac{-\|[1.5,3.5]-[4.5,1.5]\|^2}{2\sigma^2}\right)=0.0004$, where $\sigma=0.8$. So, from this example we can see that the basis functions, which are close to the position of the agent, take larger values and the closer the agent is to a basis function the larger is the amount of the update of the θ that belongs to this basis function.

For the case of the 5x5 grid I ran 50 experiments where each experiment had 1000 trials and each trial had 625 steps. I used the parameters γ=0.75, η=0.25, initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{625})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

The plot of the average reward, number of steps, delta and max values for the 50 experiments is the following:

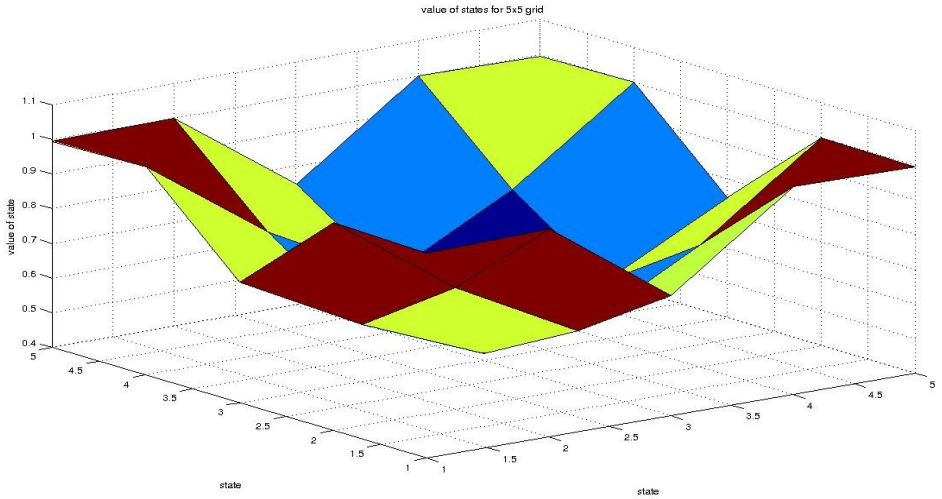


From these plots we can see that till the 650 trials the agent is learning how to go to the goal locations, because the average reward and the max value is increasing and the delta and the number of steps is decreasing. After the 650 trials we can notice that the algorithm has stopped to increase its performance and it's stable. Also, we take reasonable values for all the experiments since the average reward is always lower than the maximum reward which is 1 and the delta at the start is positive and it's decreasing and then it's always negative.

Compared to the 5x5 grid with the indicator functions we can notice that the behavior of the indicator functions and the basis functions are the same concerning the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments. We have this similarity because in both case we use 25 functions for 25 states. So, the effect of using radial basis function is not obvious in the case of the 5x5 grid.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 5x5 grid. In order to find the value of each state, for each action I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of basis functions. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 5x5 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Compared to the 5x5 grid with the indicator functions we can notice that the shape of the surface and the values of the states are the same with the indicator functions version. We have this similarity because in both case we use 25 functions for 25 states. So, the effect of using radial basis function is not obvious in the case of the 5x5 grid.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 3,5

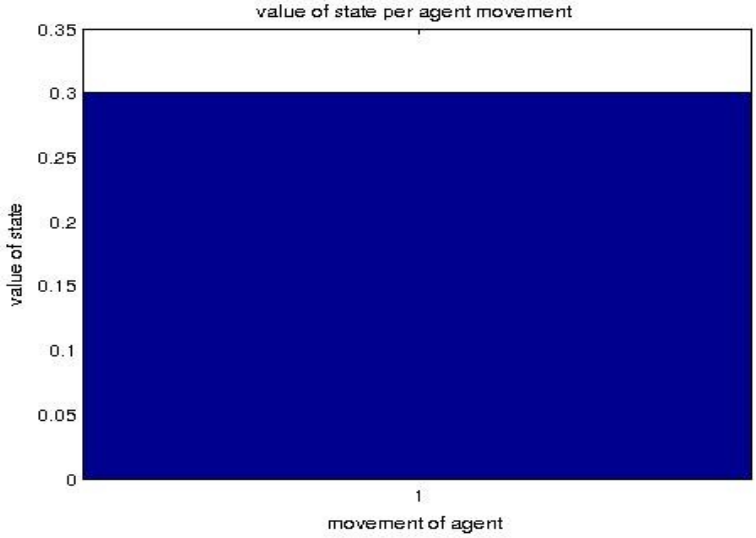
Best action is east

Next state of agent is 4,5 with value 0.299988

Best action is east

Goal

The plot of the values of this experiment is:



From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

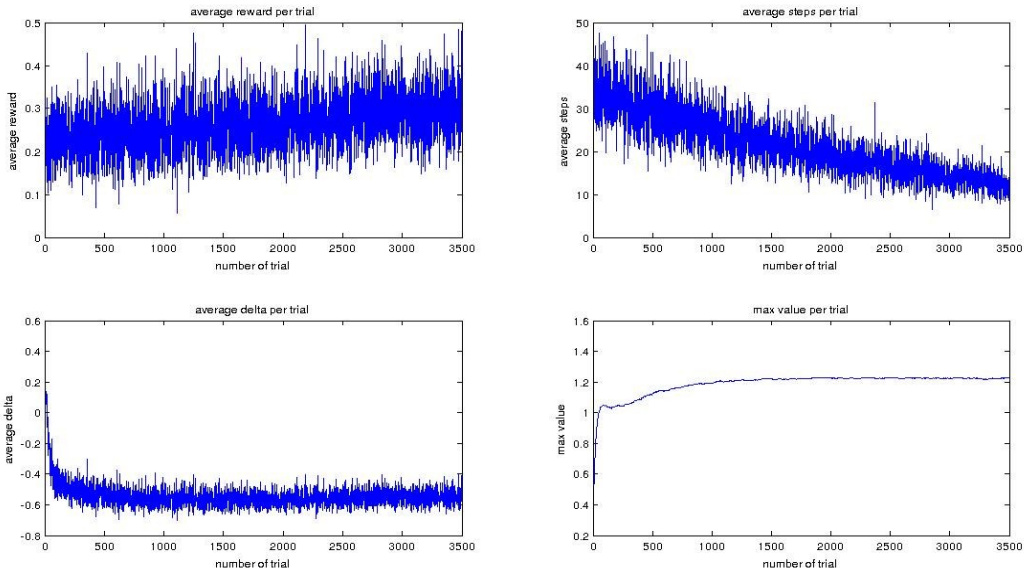
For the case of the 10x10 grid we use the same idea that we used in the case of the 5x5 grid, but now because the grid 10x10 is larger than the 5x5 grid and we want to place 25 radial basis functions in 100 squares, we'll place the center of the basis functions sparser than in the case of the 5x5 grid. The way we place the centers of the basis functions is as following:

	1	2	3	4	5
1	(1.5,1.5) φ_1	(3.5,1.5) φ_2	(5.5,1.5) φ_3	(7.5,1.5) φ_4	(9.5,1.5) φ_5
2	(1.5,3.5) φ_6	(3.5,3.5) φ_7	(5.5,3.5) φ_8	(7.5,3.5) φ_9	(9.5,3.5) φ_{10}
3	(1.5,5.5) φ_{11}	(3.5,5.5) φ_{12}	(5.5,5.5) φ_{13}	(7.5,5.5) φ_{14}	(9.5,5.5) φ_{15}
4	(1.5,7.5) φ_{16}	(3.5,7.5) φ_{17}	(5.5,7.5) φ_{18}	(7.5,7.5) φ_{19}	(9.5,7.5) φ_{20}
5	(1.5,9.5) φ_{21}	(3.5,9.5) φ_{22}	(5.5,9.5) φ_{23}	(7.5,9.5) φ_{24}	(9.5,9.5) φ_{25}

I identify the position of the agent as the center of the square where the agent is at each step. So, for the agent position we have 100 centers(equals to the number of the 100 squares).

I ran 40 experiments where each experiment had 3500 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25, initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ and η fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{10000}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

The plot of the average reward, number of steps, delta and max values for the 40 experiments is the following:



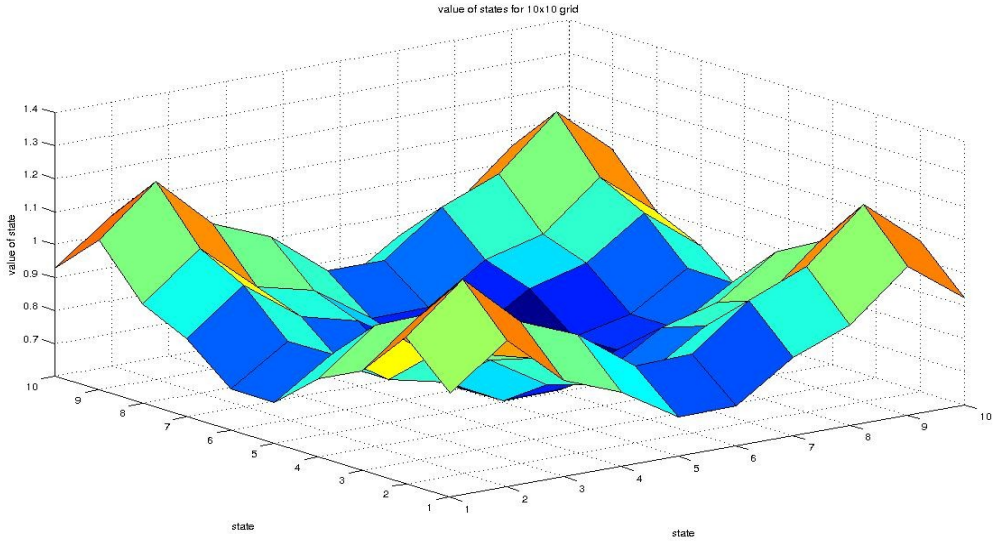
We can see that the average number of steps decreases as the number of trials increases. The average delta decreases quickly at the first trials, but after around the 100 trials it stops to decreases and it looks stable but 'noisy'. The average max value increases quickly at the first trials and then it

converges to the value 1.2. The increase of the average reward is not so clear and looks stable and 'noisy' compared to the 5x5 grid and in general the average reward is lower.

Compared to the 10x10 grid with the indicator functions we can notice that in the case of the radial basis function the agent does fewer steps and the average delta decreases faster. Also, the radial basis functions version reaches higher max value. So, the radial basis functions seems to give better performance compared to the indicator functions version.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 10x10 grid. In order to find the value of each state, for each action I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of basis functions. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 10x10 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better. Moreover, compared to the 10x10 grid of the indicator functions we can see here the surface is much more smooth. We have this smoothness because using indicator functions we have binary features only one feature is updated at every step and this is very strict and local and creates squares on the surface which have the same value(the squares that belong to a specific indicator function), but using radial basis function all the features take weight and all the feature take part in the updating of the weights at every step. So, it is reasonable that the radial basis function gives smoother surface.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 5,5

Best action is north

Next state of agent is 5,4 with value 0.367139

Best action is west

Next state of agent is 4,4 with value 0.469946

Best action is west

Next state of agent is 3,4 with value 0.592197

Best action is west

Next state of agent is 2,4 with value 0.592197

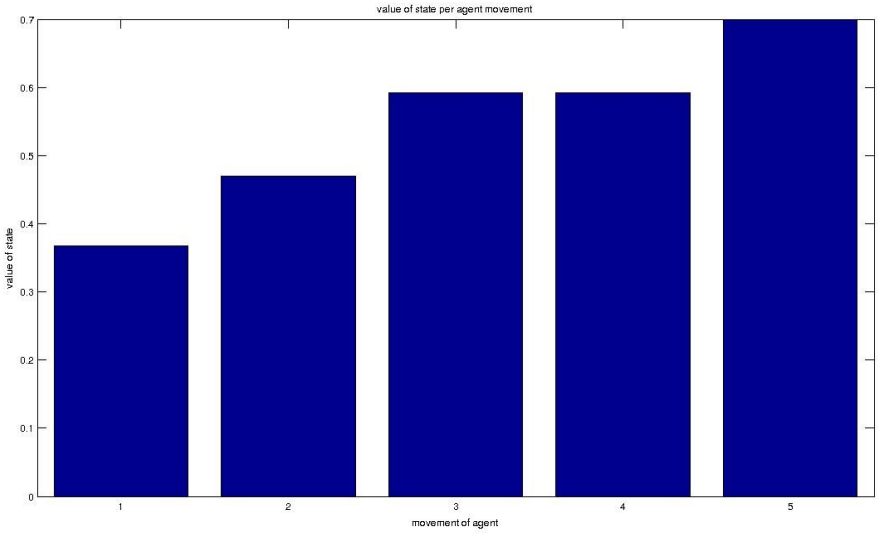
Best action is north

Next state of agent is 2,3 with value 0.699289

Best action is north

Goal

The plot of the values of this experiment is:



From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

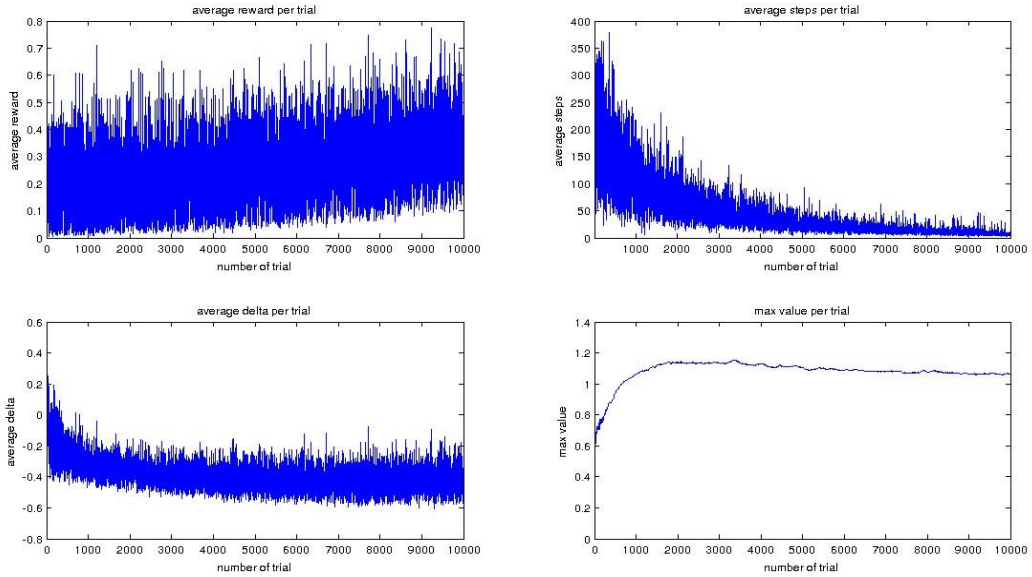
For the case of the 25x25 grid we use the same idea that we used in the case of the 5x5 grid, but now because the grid 25x25 is larger than the 5x5 grid and we want to place 25 radial basis functions in 625 squares, we'll place the center of the basis functions sparser than in the case of the 5x5 grid. The way we place the centers of the basis functions is as following:

	1	2	3	4	5
1	(2.5,2.5) φ_1	(7.5,2.5) φ_2	(12.5,2.5) φ_3	(17.5,2.5) φ_4	(22.5,2.5) φ_5
2	(2.5,7.5) φ_6	(7.5,7.5) φ_7	(12.5,7.5) φ_8	(17.5,7.5) φ_9	(22.5,7.5) φ_{10}
3	(2.5,12.5) φ_{11}	(7.5,12.5) φ_{12}	(12.5,12.5) φ_{13}	(17.5,12.5) φ_{14}	(22.5,12.5) φ_{15}
4	(2.5,17.5) φ_{16}	(7.5,17.5) φ_{17}	(12.5,17.5) φ_{18}	(17.5,17.5) φ_{19}	(22.5,17.5) φ_{20}
5	(2.5,22.5) φ_{21}	(7.5,22.5) φ_{22}	(12.5,22.5) φ_{23}	(17.5,22.5) φ_{24}	(22.5,22.5) φ_{25}

I identify the position of the agent as the center of the square where the agent is at each step. So, for the agent position we have 625 centers(equals to the number of the 625 squares).

I ran 10 experiments where each experiment had 10000 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25, initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5 . I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

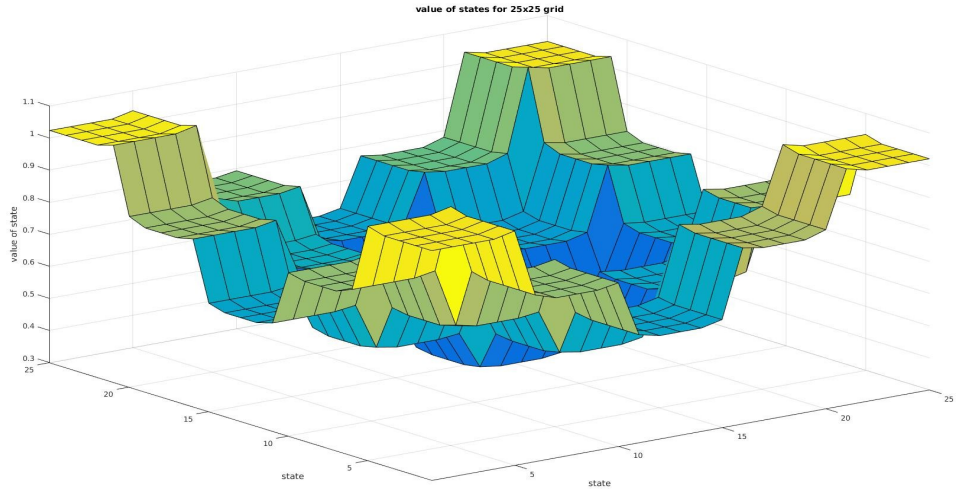
The plot of the average reward, number of steps, delta and max values for the 10 experiments is the following:



We can observe that compared to the 10x10 grid of the radial basis function version the average reward is more 'noisy' and it takes lower values, the average delta has larger values and the agents does more steps. All these observations are reasonable because the 25x25 grid is larger than the 10x10 and for the case of the 25x25 grid we use 25 basis functions to generalize 625 states. So, except that the generalization is more difficult because of the big difference between the number of the 25 basis function and the 625 states, the agent has also to do more steps to reach the goal locations.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 25x25 grid. In order to find the value of each state, for each action I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of basis functions. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 25x25 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 14,13

Best action is east

Next state of agent is 15,13 with value 0.382767

Best action is east

Next state of agent is 16,13 with value 0.382767

Best action is north

Next state of agent is 16,12 with value 0.553371

Best action is north

Next state of agent is 16,11 with value 0.553371

Best action is north

Next state of agent is 16,10 with value 0.553371

Best action is east

Next state of agent is 17,10 with value 0.654751

Best action is east

Next state of agent is 18,10 with value 0.654751

Best action is east

Next state of agent is 19,10 with value 0.654751

Best action is east

Next state of agent is 20,10 with value 0.654751

Best action is east

Next state of agent is 21,10 with value 0.654751

Best action is north

Next state of agent is 21,9 with value 0.778406

Best action is north

Next state of agent is 21,8 with value 0.778406

Best action is north

Next state of agent is 21,7 with value 0.778406

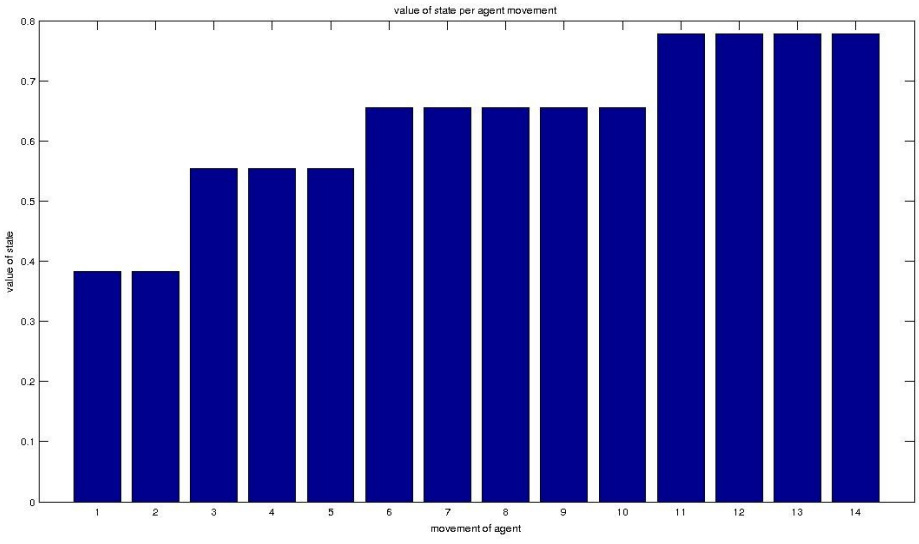
Best action is north

Next state of agent is 21,6 with value 0.778406

Best action is north

Goal

The plot of the values of this experiment is:



From the above plot we can see that the agent has learned to move to better states and then it reaches the goal. Compared to the case of the 5x5 and 10x10 grid we observe that the agent has to do more steps to reach a goal location because the grid is larger.

The 25x25 grid has broader generalization compared to the 10x10 grid because we use the same number of basis functions for a larger number of states.

The primary advantage of RBFs over binary features is that they produce approximate functions that vary smoothly, but RBFs have a higher computational cost. Also, instead of using the linear sum $Q(s,a)=\theta^T * \varphi$ to compute the action-state value we could use a non-linear sum to approximate the target function much more precisely.

Moreover, instead of using regularly spaced centers for our radial basis functions, we could place more basis functions near the goals where correct actions are more critical. Using regularly spaced basis functions we assume that all the places of the grid have the same importance.

The downside of RBF is the greater computational complexity and it needs more manual tuning before learning is robust and efficient.

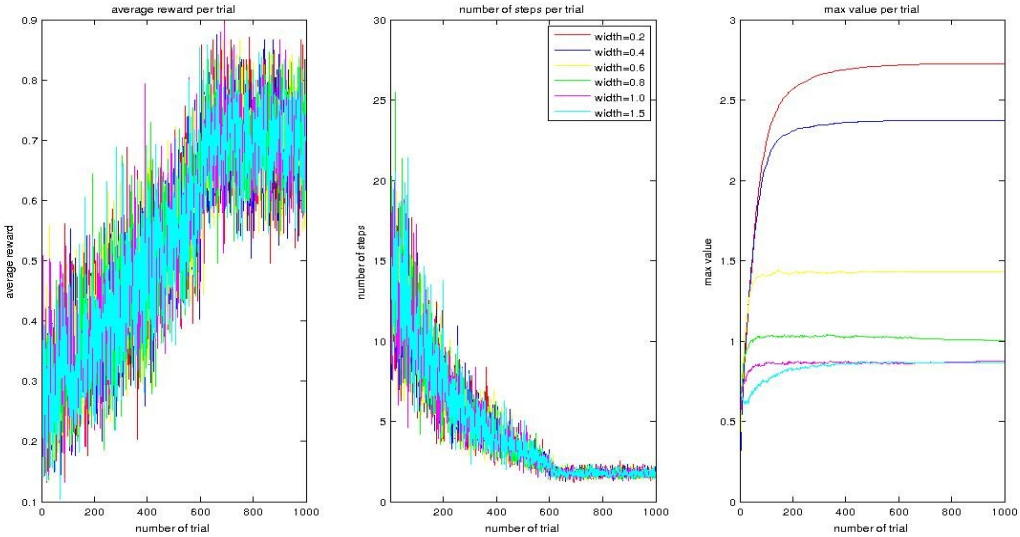
Computing the width of the basis functions we fix the degree of overlapping between the basis functions. A small value of σ creates a narrow gaussian bell and a large value creates a broad gaussian bell. This allows finding a compromise between locality and smoothness. If the distances between the centers are not equal, it is better to assign a specific width to each radial basis function. For example, we could assign a larger width to basis functions are widely separated from each other and a smaller width to closer ones.

In general the value of the width plays a very significant role. For example at a classification problem the effectiveness of RBF networks is extremely sensitive to the values used for hidden node widths. A width that's too small tends to over-fit the training data, leading to poor classification accuracy. A width that's too large tends to under-fit the data, which also leads to poor classification.

We should select the widths in such away to guarantee a natural overlap between the radial basis functions, preserving the local properties and at the same time to maximize the generalization ability.

Now let's observe the effect of different values of σ. For the 5x5 grid I ran 20 experiments where each experiment had 1000 trials and each trial had 625 steps. I used the parameters γ=0.75, η=0.25 and initial ε=1.0. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps> and the average max value <max value> between the experiments.

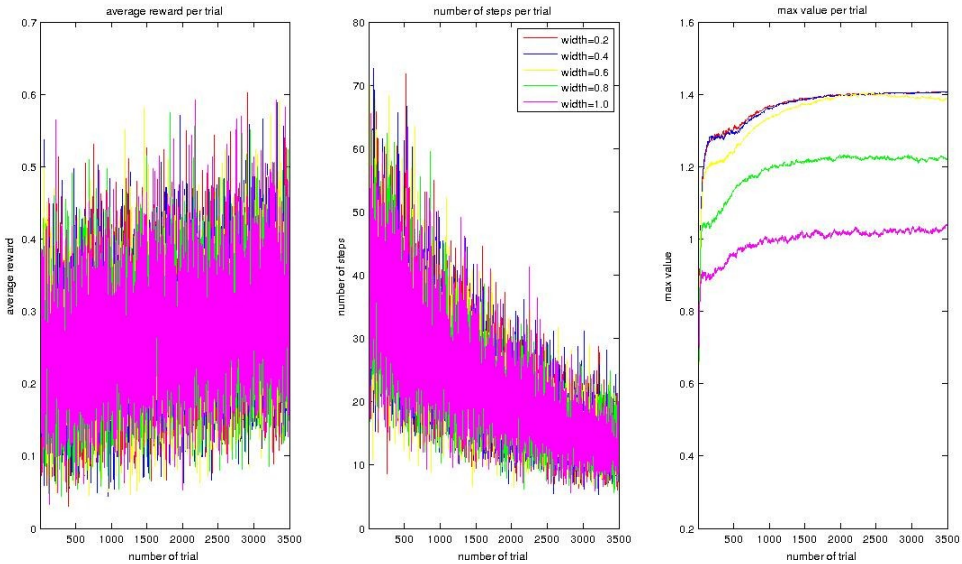
The plot of the average reward, number of steps, delta and max values for the 20 experiments is the following:



From these plots we can see that as far as the average reward and the average number of steps are concerned our problem seems to have the same behavior regardless of the value of σ. Concerning the max value we notice that the lower the value of σ is, the higher the value of max value is. We have this behavior because smaller values of σ create more localized radial basis function because these small values create narrow distributions. As a result our network of the regularly spaced radial basis functions is not smooth and at the updates of θ only some θs are updated each time taking all the value of the update and this leads to larger values compared to a smooth network where a lot of θs are updated at every step participating by a specific amount at the update of θ.

For the 10x10 grid I ran 20 experiments where each experiment had 3500 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25 and initial ε=1.0. I kept the γ and η fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{10000}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps> and the average max value <max value> between the experiments.

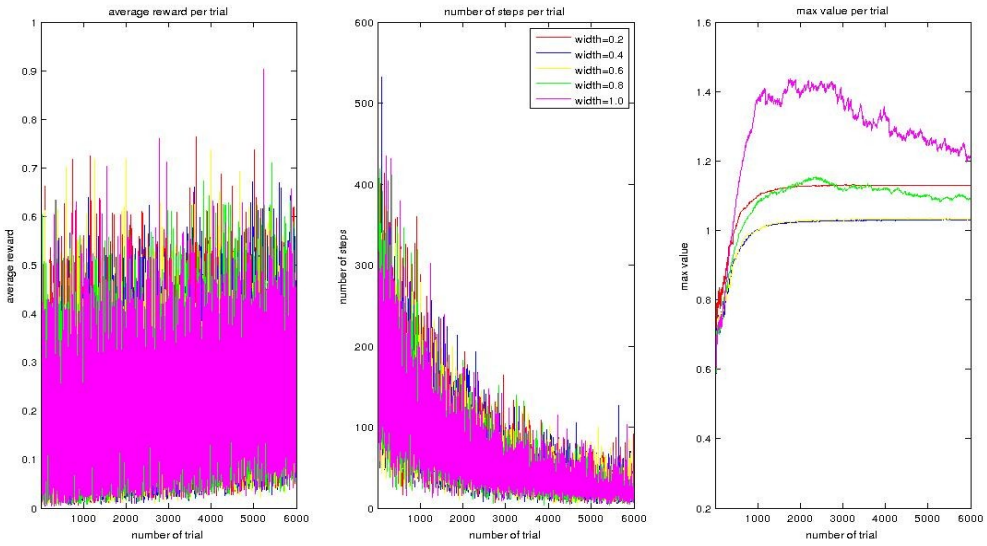
The plot of the average reward, number of steps, delta and max values for the 20 experiments is the following:



Also, at these plots we can make the same conclusions with the 5x5 grid about the values of the widths and the max values because again we observe that as the value of the width decreases, the max value increases. Concerning the average reward and the average number of steps we can see that the behavior look similar for all the values of the width.

For the 25x25 grid I ran 10 experiments where each experiment had 6000 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25 and initial ε=1.0. I kept the γ and η fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{10000}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps> and the average max value <max value> between the experiments.

The plot of the average reward, number of steps, delta and max values for the 10 experiments is the following:



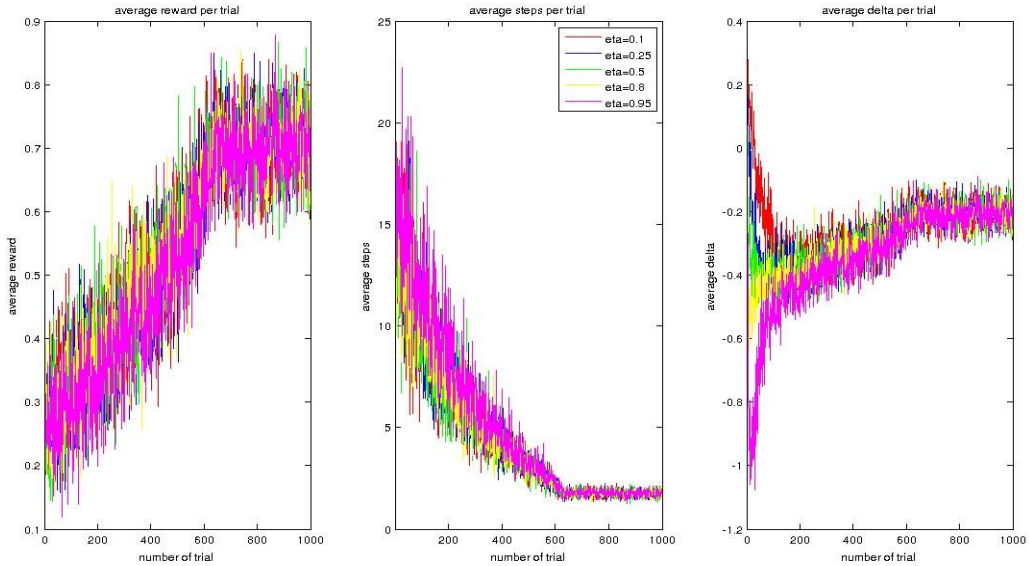
From these plots we can't make safe conclusions because the effect of the values of the widths is not so clear.

Now, let's investigate the effect of the parameters γ and η. As we know learning rate determines to what extent the newly acquired information will override the old information and as the η increases we consider more and more the most recent information. As I said in question 1, γ determines the importance of future rewards.

First I'll investigate the effect of the learning rate η.

For the case of the 5x5 grid I ran 30 experiments where each experiment had 1000 trials and each trial had 625 steps. I used the parameters γ=0.75 and initial ε=1.0 and radial basis function width $\sigma=\frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{625}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

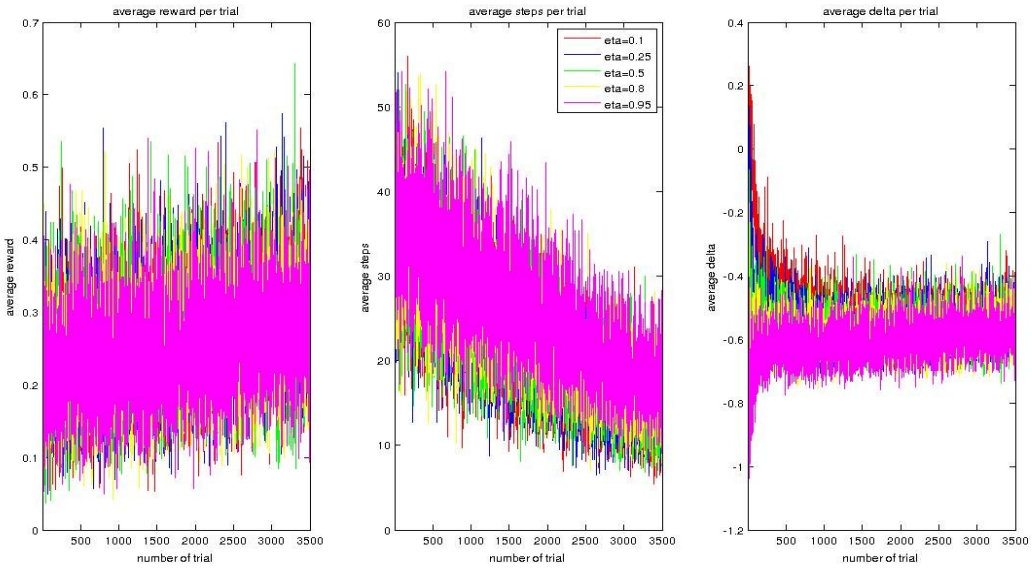
The plot of the effect of η is:



For the average reward and number of steps we can't make safe conclusions about the difference on the performance based on the value of η. Concerning the average delta we can observe that at the first trials we have the largest error for lower values of η. Especially we have the largest error for η=0.1 which is the lowest value for which I did the experiments. So, we can conclude that when we have low η the learning procedure is slower, but as the number of trials increases it seems that for all the values of η, we have the same behavior.

For the case of the 10x10 grid I ran 25 experiments where each experiment had 3500 trials and each trial had 10000 steps. I used the parameters γ=0.9 and initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{10000}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

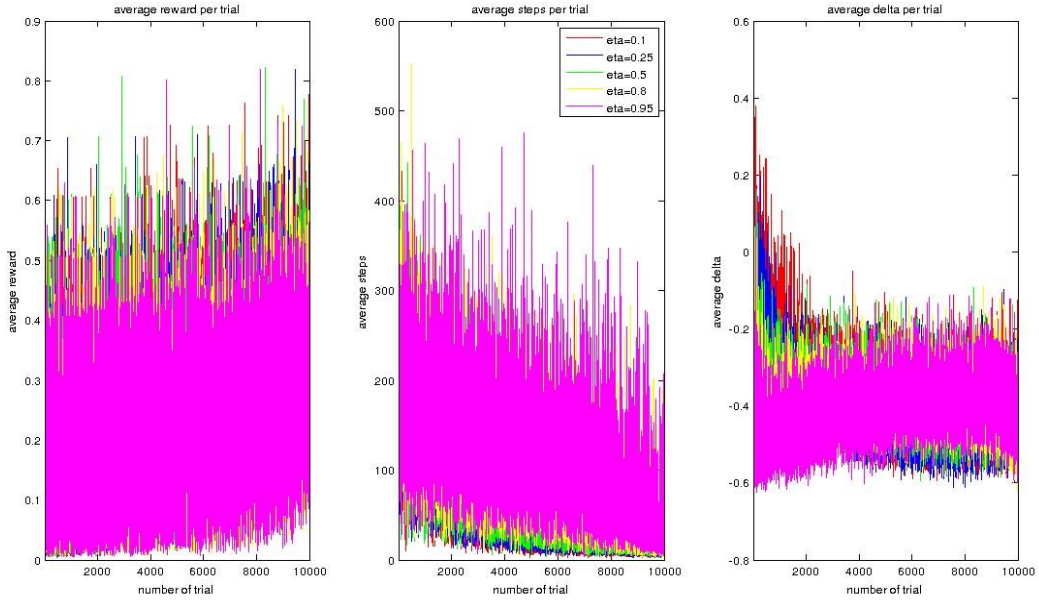
The plot of the effect of η is:



For these plots we can make the same conclusions with the case of the 5x5 grid. At the first trials low η gives larger error. But the plot is more 'noisy' compared to the 5x5 grid.

For the case of the 25x25 grid I ran 10 experiments where each experiment had 10000 trials and each trial had 10000 steps. I used the parameters γ=0.9 and initial ε=1.0 and radial basis function width $\sigma=\frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

The plot of the effect of η is:



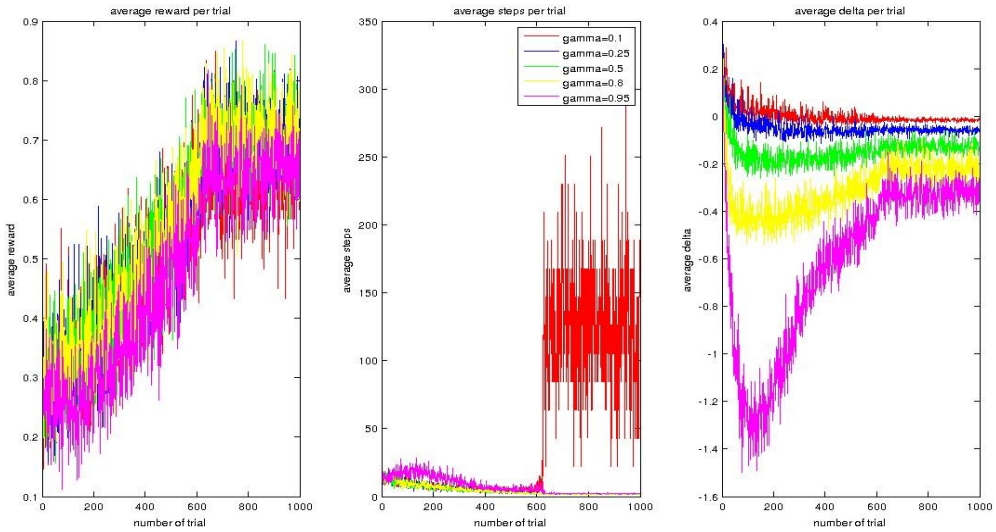
Compared to the 5x5 and 10x10 grids we can observe the plots are much more 'noisy' and we can make safe conclusions. But, at the average delta it seems that when we have η=0.1 at the first trials we get larger error than when we have η=0.95. And this agrees with the observations for the grid 5x5 and 10x10.

So, these observations confirm that our problem is deterministic as we know that in fully deterministic problem a learning rate of value=1 is optimal(we can see that at the first trials larger η gives lower delta). In comparison to stochastic problems where we should use η close to 0. But, as the number of trials increases our problem seems to behave the same for any value of η.

Now I'll investigate the effect of the learning rate γ.

For the case of the 5x5 grid I ran 30 experiments where each experiment had 1000 trials and each trial had 625 steps. I used the parameters η=0.25 and initial ε=1.0 and radial basis function width $\sigma=\frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{625})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

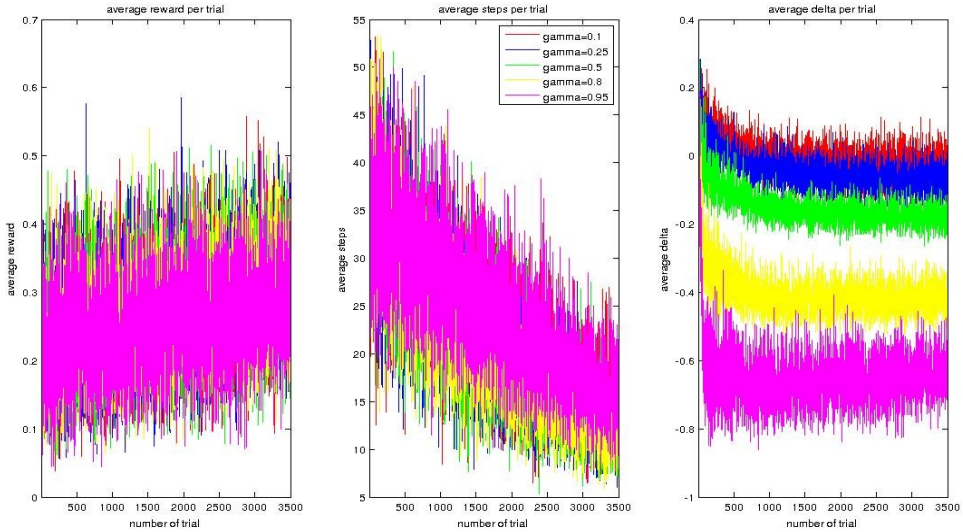
The plot of the effect of γ is:



For the average reward and number of steps we can't make safe conclusions about the difference on the performance based on the value of γ. Concerning the average delta we can observe that we have the largest error for lower values of γ. Especially we have the largest error for γ=0.1 which is the lowest value for which I did the experiments. So, we can conclude that when we have low γ the learning procedure is slower because the agent only considers the current reward and it doesn't care for the future reward.

For the case of the 10x10 grid I ran 25 experiments where each experiment had 3500 trials and each trial had 10000 steps. I used the parameters η=0.25 and initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the η fixed and I decreased the ε in every trial as follows $\left(1-\frac{t}{10000}\right)^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

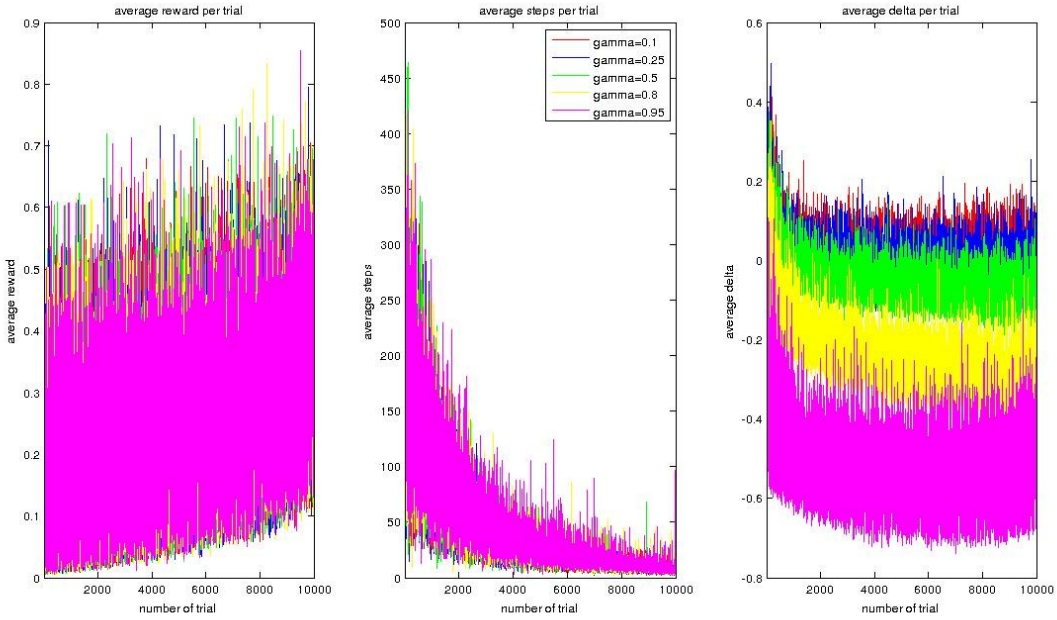
The plot of the effect of γ is:



For these plots we can make the same conclusions with the case of the 5x5 grid. Lower values of γ give error because the agent only considers the current reward and it doesn't care for the future reward.

For the case of the 25x25 grid I ran 10 experiments where each experiment had 10000 trials and each trial had 10000 steps. I used the parameters η=0.25 and initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

The plot of the effect of γ is:

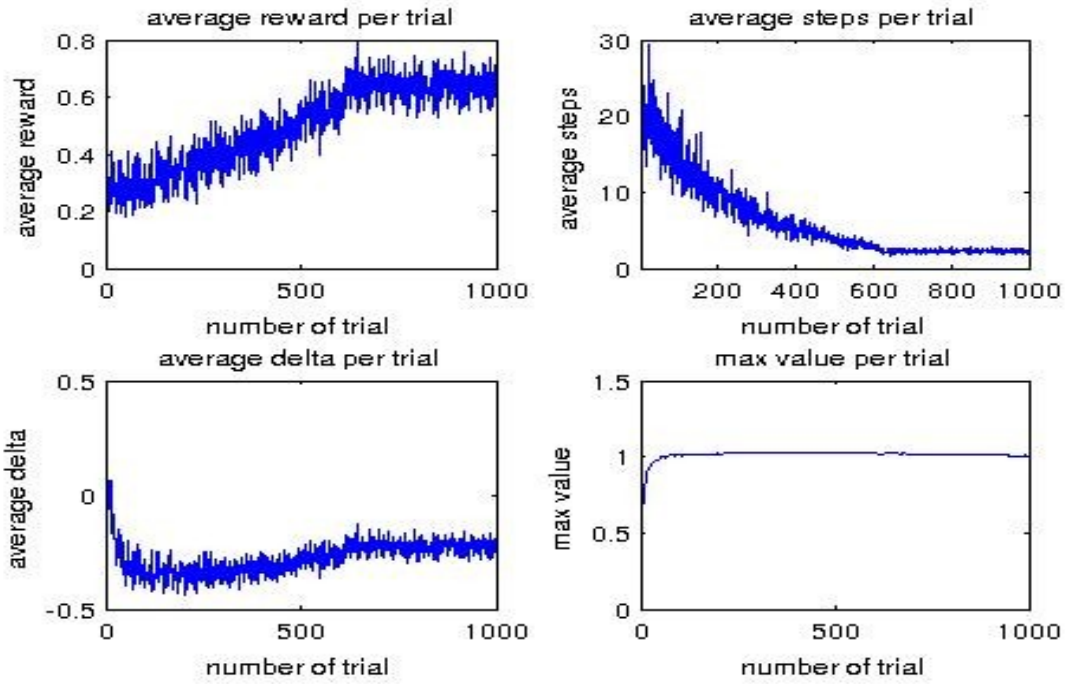


Again we can see that when we use larger values for γ we get lower error because the agent takes into consideration the future reward when it selects an action and as a consequence it takes better actions. Compared to the 5x5 and 10x10 here the plot are more 'noisy' because the grid is larger and we use more generalization.

5) For this question I'll use the obstacles like the taxi problem of the assignment 1 and all the other details about the basis functions, for example how I place the center of the basis functions on the grid are exactly the same with the question 4.

For the case of the 5x5 grid with obstacles I ran 50 experiments where each experiment had 1000 trials and each trial had 625 steps. I used the parameters γ=0.75, η=0.25, initial ε=1.0 and radial basis function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{625})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

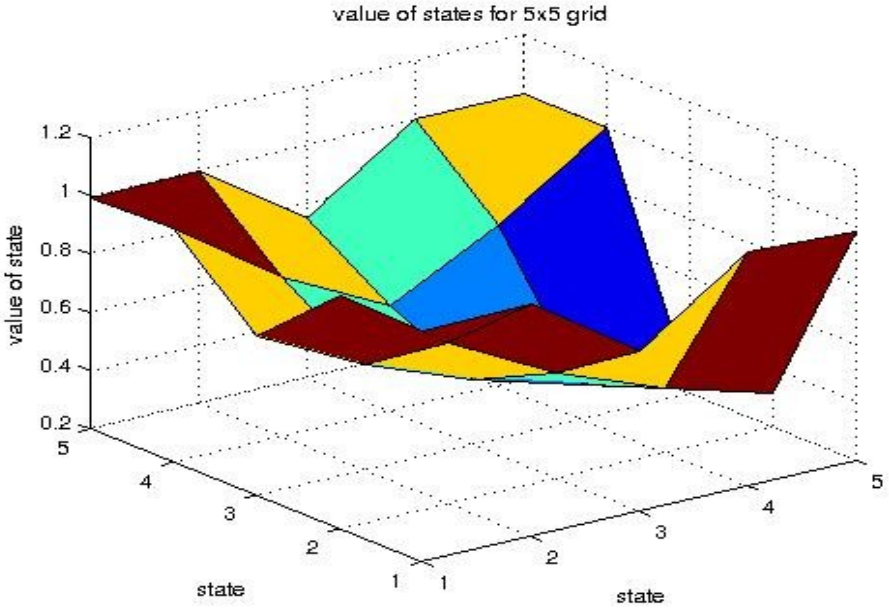
The plot of the average reward, number of steps, delta and max values for the 50 experiments is the following:



Compared to the 5x5 grid without obstacles we can see that in this case where we have obstacles the average number of steps especially at the first trials is larger than the case without obstacles because the agent has to avoid the obstacle and as a consequence it does more steps. After some trials where the agent has learned to navigate avoiding the obstacles, it does approximately the same steps with the version without obstacles. Also, the average reward, average delta and max value seem to have the same behavior with the version without obstacles.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 5x5 grid. In order to find the value of each state, for each action I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of basis functions. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 5x5 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Compared to the 5x5 grid without obstacles we can see that the states near the places where we have the obstacles have lower value than the states far away from the obstacles and especially the goal locations. That means the agent has learned that the states near the obstacles are not good states and it avoids them.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 1,3

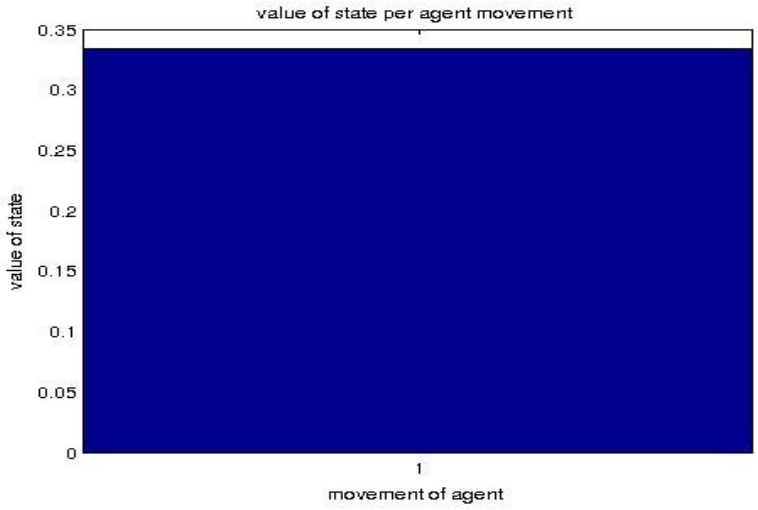
Best action is south

Next state of agent is 1,4 with value 0.334182

Best action is south

Goal

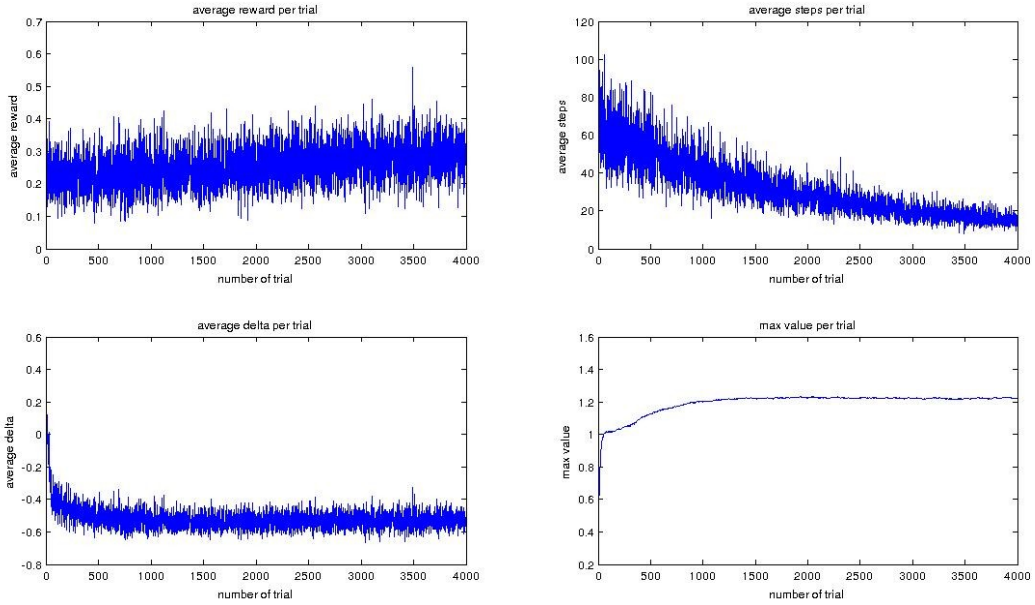
The plot of the values of this experiment is:



From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

For the case of the 10x10 grid I ran 40 experiments where each experiment had 4000 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25, initial ε=1.0 and radial basis function width $\sigma=\frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

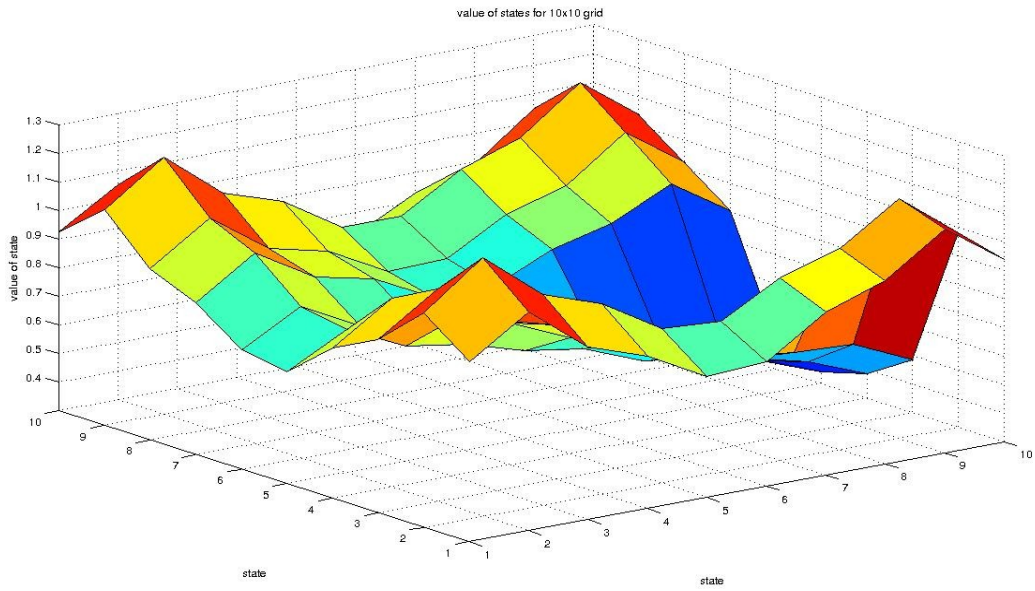
The plot of the average reward, number of steps, delta and max values for the 40 experiments is the following:



Compared to the 10x10 grid without obstacles we can see that in this case where we have obstacles the average number of steps is larger than the case without obstacles because the agent has to avoid the obstacles and as a consequence it does more steps. This agrees with the observation we made at the case of the 5x5 grid with obstacles. Also, the average reward, average delta and max value seem to have the same behavior with the version without obstacles.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 10x10 grid. In order to find the value of each state, for each action I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of basis functions. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 10x10 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Again compared to the 10x10 grid without obstacles we can see that the states near the places where we have the obstacles have lower value than the states far away from the obstacles and especially the goal locations. That means the agent has learned that the states near the obstacles are not good states and it avoids them.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 8,5

Best action is south

Next state of agent is 8,6 with value 0.478192

Best action is south

Next state of agent is 8,7 with value 0.478192

Best action is east

Next state of agent is 9,7 with value 0.598564

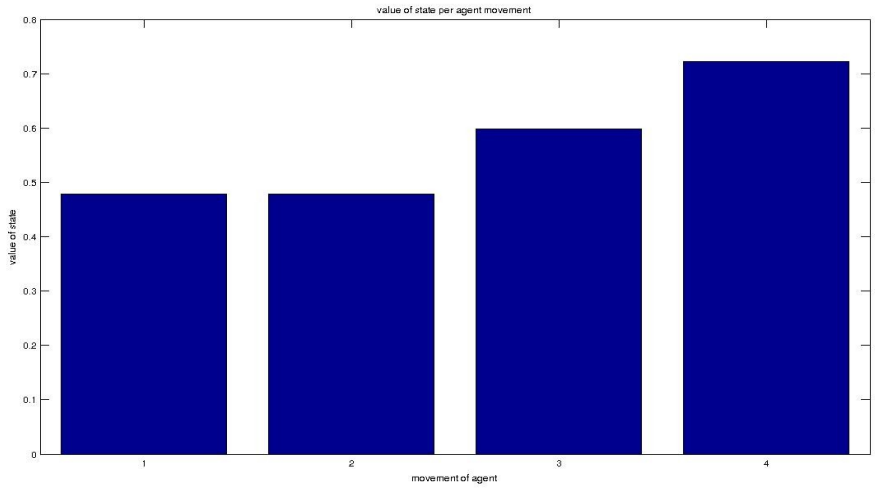
Best action is south

Next state of agent is 9,8 with value 0.722456

Best action is south

Goal

The plot of the values of this experiment is:

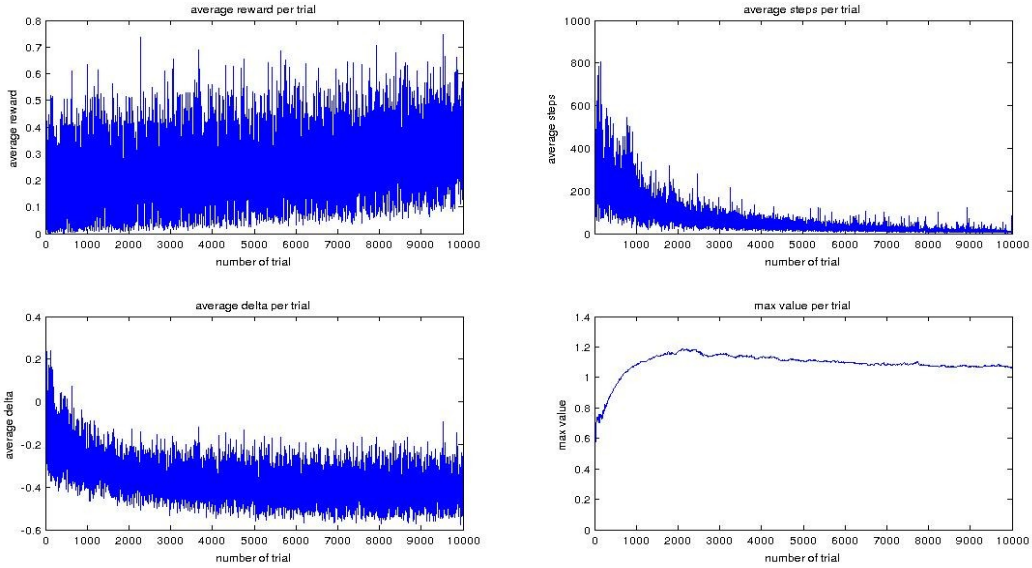


From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

For the case of the 25x25 grid I ran 10 experiments where each experiment had 10000 trials and each trial had 10000 steps. I used the parameters γ=0.9, η=0.25, initial ε=1.0 and radial basis

function width $\sigma = \frac{\sqrt{2*(N-1)^2}}{\sqrt{2}*N}$ where N equals to 5. I kept the γ and η fixed and I decreased the ε in every trial as follows $(1-\frac{t}{10000})^{\frac{1}{2}}$, where t is the number of trial. Then, for each trial I took the average reward <r>, the average number of steps <steps>, the average delta <delta> and the average max value <max value> between the experiments.

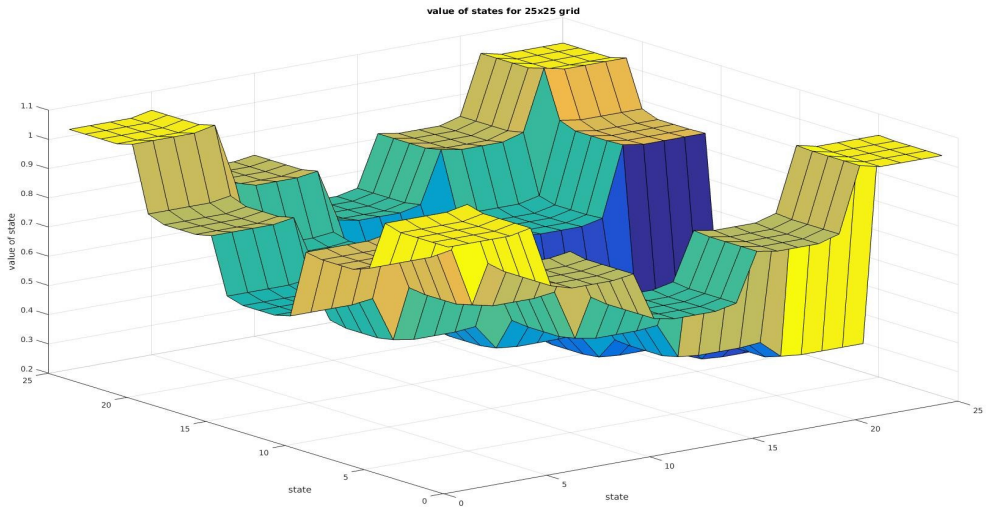
The plot of the average reward, number of steps, delta and max values for the 40 experiments is the following:



Compared to the 25x25 grid without obstacles we can see that in this case where we have obstacles the average number of steps, especially at the first trials, is larger than the case without obstacles because the agent has to avoid the obstacles and as a consequence it does more steps. This agrees with the observation we made at the case of the 5x5 and 10x10 grid with obstacles. Also, the average reward, average delta and max value seem to have the same behavior with the version without obstacles.

In order to check that our problem has been trained and that the values of the 4 corners of the grid have the highest values, I plot the values of the 25x25 grid. In order to find the value of each state, for each action I compute θ*F, where θ is the 25x5 matrix where I store the values of the action-dependent parameters and F, for every possible action a, is the set of basis functions. Then, as a value of a state I take the max value of an action for each state.

The plot of the values of the 25x25 grid is the following:



From the plot we can observe that the states near the 4 corners of the grid have greater values than the states at the middle of the grid and the values of the 4 corners have the highest values. So, the agent has learned to go to the corners, because these states are better.

Again compared to the 25x25 grid without obstacles we can see that the states near the places where we have the obstacles have lower value than the states far away from the obstacles and especially the goal locations. That means the agent has learned that the states near the obstacles are not good states and it avoids them.

Now, we'll run an experiment to see how the agent behaves after it has learned to go to the goal locations.

The initial state of the agent is 23,15

Best action is south

Next state of agent is 23,16 with value 0.660988

Best action is south

Next state of agent is 23,17 with value 0.826798

Best action is south

Next state of agent is 23,18 with value 0.826798

Best action is south

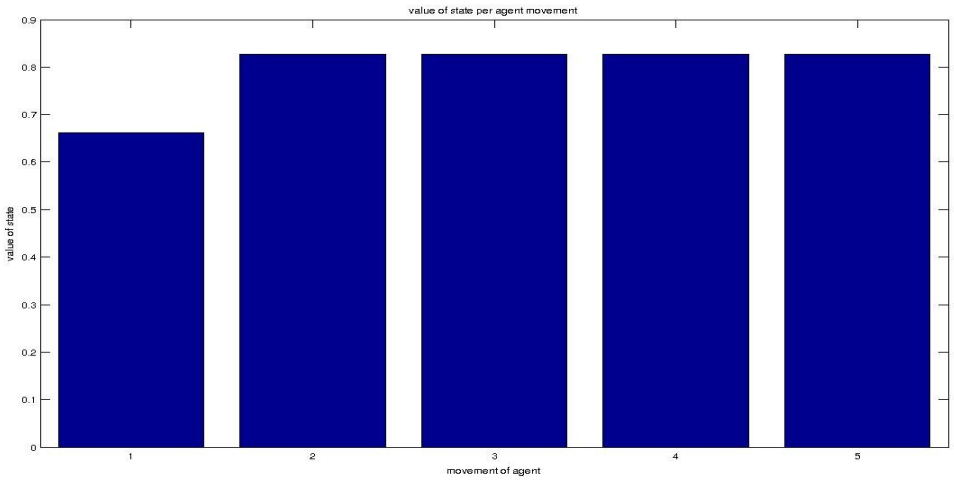
Next state of agent is 23,19 with value 0.826798

Best action is south

Next state of agent is 23,20 with value 0.826798

Best action is south
Goal

The plot of the values of this experiment is:



From the above plot we can see that the agent has learned to move to better states and then it reaches the goal.

From all the above observation we conclude that we can use the function approximation for non-trivial problem and it works well.

7) From all the above discussion and experiments we can draw the following conclusions:

- The 5x5 grid with the 25 indicator functions is an exact reproduction of the lookup table version, but in one case we have the 5x25 θ matrix and in the other case the 25x5 Q matrix
- In the case of the 5x5 grid both the lookup table version and the indicator functions version have exactly the same behavior for the average reward and the average steps per trial. But, the lookup table version gives higher max value
- When we have more states than the basis functions we have generalization, since a basis function corresponds to more than one state and changing the θ of one basis function changes the estimated value of more than one state
- For the 5x5 and 10x10 we had to redefine the goal locations because the states next to the corners of the grids had the same policy
- As the grid is getting bigger, the plots are getting more 'noisy' because of the bigger generalization(we use a specific number of basis functions for a bigger grid)
- The 5x5 grid compared to the 10x10 grid gives larger average reward and fewer steps because the agent does fewer steps to go the goal locations. And the 5x5 grid can be trained faster and have larger weights and max values in earlier trials than the 10x10 grid
- The convergence of the 10x10 is slower because the necessary generalization leads our problem of the 10x10 grid to be more difficult to converge
- In the case of the 5x5 grid the performance of the radial basis functions and the indicator functions is the same because we have the same number of basis functions and states, but for 10x10 and 25x25 grids the radial basis functions version gives better performance compared to the indicator functions version
- The radial basis functions give smoother surfaces
- As the grid is getting bigger the agent has to do more steps and as a consequence we get lower average reward
- Computing the width of the basis functions we fix the degree of overlapping between the basis functions and we should select the widths in such away to guarantee a natural overlap between the radial basis functions, preserving the local properties and at the same time to maximize the generalization ability
- With smaller width we get larger max value because smaller values of σ create more localized radial basis function
- For smaller learning rate we get lower average error at the first trials
- For smaller γ we get lower average error
- Using obstacles we can see at the surfaces that the states close to the obstacles have smaller values, because the agent has learned to avoid these states
- In the case of the obstacles at the first trials the agent does more steps because it hasn't learned to avoid them and as a consequence it does more steps compared to the grids without obstacles
- The function approximation can effectively be applied to non-trivial problems

References

[1] Richard S. Sutton and Andrew G. Barto, 2014-2015 second edition, page 212