Tutorial on Value-Iteration Algorithm

Daniel Fišer

Department of Computer Science,
Faculty of Electrical Engineering,
danfis@danfis.cz

May 28, 2020

Definitions

MDP

A Infinite-Horizon Stationary Descrete-Time Fully Observable Markov Decision Process (MDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where:

- $\mathcal{S}$ is a finite set of states;
- $\mathcal{A}$ is a finite set of actions;
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is a transition function, a mapping specifying the probability $\mathcal{T}(s_1, a, s_2)$ of going to state s_2 if action a is executed when the agent is in state s_1;
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is a reward function that gives a finite numeric reward value $\mathcal{R}(s_1, a, s_2)$ obtained when the system goes from state s_1 to state s_2 as a result of executing action a.

Definitions

Policy

A Stationary Markovian Policy for an MDP is a function $\pi : \mathcal{S} \mapsto \mathcal{A}$ mapping states to actions.

Value Function

A stationary Markovian value function of a policy π is a mapping $V^\pi : \mathcal{S} \mapsto \mathbb{R}$ such that $V^\pi(s) = \sum_{s' \in \mathcal{S}} \mathcal{T}(s, \pi(s), s')[\mathcal{R}(s, \pi(s), s') + \gamma V^\pi(s')]$ for every state $s \in \mathcal{S}$, where γ is a discount factor.
The optimal value function $V^\star$ is

$$V^\star(s) = \max_{a \in \mathcal{A}} \left[\sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + \gamma V^\star(s')] \right] \tag{1}$$

and the optimal policy $\pi^\star$ is

$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} \left[\sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + \gamma V^\star(s')] \right] \tag{2}$$

Definitions

Value Function

A stationary Markovian value function of a policy π is a mapping $V^\pi : \mathcal{S} \mapsto \mathbb{R}$ such that $V^\pi(s) = \sum_{s' \in \mathcal{S}} \mathcal{T}(s, \pi(s), s')[\mathcal{R}(s, \pi(s), s') + \gamma V^\pi(s')]$ for every state $s \in \mathcal{S}$, where γ is a discount factor.

The optimal value function $V^\star$ is

$$V^\star(s) = \max_{a \in \mathcal{A}} \Big[\sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + \gamma V^\star(s')] \Big] \tag{1}$$

and the optimal policy $\pi^\star$ is

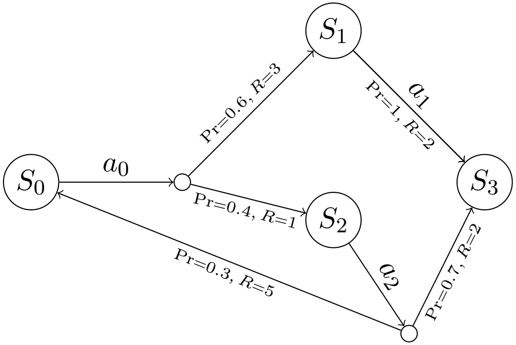
$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} \Big[\sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + \gamma V^\star(s')] \Big] \tag{2}$$

In this tutorial, we will always consider $\gamma = 1$, so we will simply disregard it from the equations.

Example 1

Consider the following example:

- Set of states $\mathcal{S} = \{S_0, \ldots, S_3\}$
- Set of actions $\mathcal{A} = \{a_0, a_1, a_2\}$
- Transition function $\mathcal{T}$:
    - $\mathcal{T}(S_0, a_0, S_1) = 0.6$
    - $\mathcal{T}(S_0, a_0, S_2) = 0.4$
    - $\mathcal{T}(S_1, a_1, S_3) = 1$
    - $\mathcal{T}(S_2, a_2, S_3) = 0.7$
    - $\mathcal{T}(S_2, a_2, S_0) = 0.3$
    - All other transition probabilities are 0
- Reward function $\mathcal{R}$:
    - $\mathcal{R}(S_0, a_0, S_1) = 3$
    - $\mathcal{R}(S_0, a_0, S_2) = 1$
    - $\mathcal{R}(S_1, a_1, S_3) = 2$
    - $\mathcal{R}(S_2, a_2, S_3) = 2$
    - $\mathcal{R}(S_2, a_2, S_0) = 5$



Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

- We need to find V^π for $S_0, \ldots, S_3$.

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

- We need to find V^π for $S_0, \ldots, S_3$.
- Intuitively, what would you say is the correct answer?

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

- We need to find V^π for $S_0, \ldots, S_3$.
- Intuitively, what would you say is the correct answer?
- First look at the states S_1, S_2, and S_3. What amount of reward can we gain by applying action a_0 there? (Go back to the definition of the value function.)

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

- We need to find V^π for $S_0, \ldots, S_3$.
- Intuitively, what would you say is the correct answer?
- First look at the states S_1, S_2, and S_3. What amount of reward can we gain by applying action a_0 there? (Go back to the definition of the value function.)
- Well, $\mathcal{T}(s, a_0, s') = 0$ for $s, s' \in \{S_1, S_2, S_3\}$. So, we wil always get $V^\pi(S_1) = V^\pi(S_2) = V^\pi(S_3) = 0$.

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

- We need to find V^π for $S_0, \ldots, S_3$.
- Intuitively, what would you say is the correct answer?
- First look at the states S_1, S_2, and S_3. What amount of reward can we gain by applying action a_0 there? (Go back to the definition of the value function.)
- Well, $\mathcal{T}(s, a_0, s') = 0$ for $s, s' \in \{S_1, S_2, S_3\}$. So, we wil always get $V^\pi(S_1) = V^\pi(S_2) = V^\pi(S_3) = 0$.
- Therefore, $V^\pi(S_0) = \mathcal{T}(S_0, a_0, S_1)[\mathcal{R}(S_0, a_0, S_1) + 0] + \mathcal{T}(S_0, a_0, S_2)[\mathcal{R}(S_0, a_0, S_2) + 0] = 0.6 \times 3 + 0.4 \times 1 = 2.2$

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = \pi(S_1) = \pi(S_2) = \pi(S_3) = a_0$?

- We need to find V^π for $S_0, \ldots, S_3$.
- Intuitively, what would you say is the correct answer?
- First look at the states S_1, S_2, and S_3. What amount of reward can we gain by applying action a_0 there? (Go back to the definition of the value function.)
- Well, $\mathcal{T}(s, a_0, s') = 0$ for $s, s' \in \{S_1, S_2, S_3\}$. So, we wil always get $V^\pi(S_1) = V^\pi(S_2) = V^\pi(S_3) = 0$.
- Therefore, $V^\pi(S_0) = \mathcal{T}(S_0, a_0, S_1)[\mathcal{R}(S_0, a_0, S_1) + 0] + \mathcal{T}(S_0, a_0, S_2)[\mathcal{R}(S_0, a_0, S_2) + 0] = 0.6 \times 3 + 0.4 \times 1 = 2.2$
- This wasn't a very good policy, right? Actually, the example is so simple that the only reasonable policy is $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = X$, where X can be any action.

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.
- What can we gain from using the policy π in state S_3?

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.
- What can we gain from using the policy π in state S_3?
- $\mathcal{T}(S_3, a_0, X) = 0$ for every X, so the value function for S_3 must be zero.

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.
- What can we gain from using the policy π in state S_3?
- $\mathcal{T}(S_3, a_0, X) = 0$ for every X, so the value function for S_3 must be zero.
- Therefore, $V^\pi(S_1) = \mathcal{T}(S_1, a_1, S_3)[\mathcal{R}(S_1, a_1, S_3) + V^\pi(S_3)] = 1 \times (2 + 0) = 2$.

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.
- What can we gain from using the policy π in state S_3?
- $\mathcal{T}(S_3, a_0, X) = 0$ for every X, so the value function for S_3 must be zero.
- Therefore, $V^\pi(S_1) = \mathcal{T}(S_1, a_1, S_3)[\mathcal{R}(S_1, a_1, S_3) + V^\pi(S_3)] = 1 \times (2 + 0) = 2$.
- So far, it is easy, but what about $V^\pi(S_2)$?
- Let's write down the equation:
  $V^\pi(S_2) = \mathcal{T}(S_2, a_2, S_3)[\mathcal{R}(S_2, a_2, S_3) + V^\pi(S_3)] + \mathcal{T}(S_2, a_2, S_0)[\mathcal{R}(S_2, a_2, S_0) + V^\pi(S_0)] =$
  $\mathcal{T}(S_2, a_2, S_3)[\mathcal{R}(S_2, a_2, S_3) + 0] + \mathcal{T}(S_2, a_2, S_0)[\mathcal{R}(S_2, a_2, S_0) + V^\pi(S_0)] =$
  $0.7 \times [2 + 0] + 0.3 \times [5 + V^\pi(S_0)] = 1.4 + 1.5 + 0.3 \times V^\pi(S_0) = 2.9 + 0.3 \times V^\pi(S_0)$

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.
- What can we gain from using the policy π in state S_3?
- $\mathcal{T}(S_3, a_0, X) = 0$ for every X, so the value function for S_3 must be zero.
- Therefore, $V^\pi(S_1) = \mathcal{T}(S_1, a_1, S_3)[\mathcal{R}(S_1, a_1, S_3) + V^\pi(S_3)] = 1 \times (2 + 0) = 2$.
- So far, it is easy, but what about $V^\pi(S_2)$?
- Let's write down the equation:
  $V^\pi(S_2) = 2.9 + 0.3 \times V^\pi(S_0)$
- But we don't know the value of $V^\pi(S_0)$, right?
- So, let's write down the equation:
  $V^\pi(S_0) = \mathcal{T}(S_0, a_0, S_1)[\mathcal{R}(S_0, a_0, S_1) + V^\pi(S_1)] + \mathcal{T}(S_0, a_0, S_2)[\mathcal{R}(S_0, a_0, S_2) + V^\pi(S_2)] = 0.6 \times [3 + 2] + 0.5 \times [1 + V^\pi(S_2)] = 3.5 + 0.5 \times V^\pi(S_2)$

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

- Again, try to reason about this intuitively.
- What can we gain from using the policy π in state S_3?
- $\mathcal{T}(S_3, a_0, X) = 0$ for every X, so the value function for S_3 must be zero.
- Therefore, $V^\pi(S_1) = \mathcal{T}(S_1, a_1, S_3)[\mathcal{R}(S_1, a_1, S_3) + V^\pi(S_3)] = 1 \times (2 + 0) = 2$.
- So far, it is easy, but what about $V^\pi(S_2)$?
- Let's write down the equation:
  $V^\pi(S_2) = 2.9 + 0.3 \times V^\pi(S_0)$
- But we don't know the value of $V^\pi(S_0)$, right?
- So, let's write down the equation:
  $V^\pi(S_0) = 3.5 + 0.5 \times V^\pi(S_2)$
- And again, we don't know $V^\pi(S_2)$. How do you solve this problem?

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?

Let's recapitulate:

- $V^\pi(S_3) = 0$
- $V^\pi(S_1) = 2$
- $V^\pi(S_2) = 2.9 + 0.3 \times V^\pi(S_0)$
- $V^\pi(S_0) = 3.5 + 0.5 \times V^\pi(S_2)$

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?
Let's recapitulate:

- $V^\pi(S_3) = 0$
- $V^\pi(S_1) = 2$
- $V^\pi(S_2) = 2.9 + 0.3 \times V^\pi(S_0)$
- $V^\pi(S_0) = 3.5 + 0.5 \times V^\pi(S_2)$
- But it is obvious, how to solve it:
  We have two equations and two variables $V^\pi(S_2)$ and $V^\pi(S_0)$!

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?
Let's recapitulate:

- $V^\pi(S_3) = 0$
- $V^\pi(S_1) = 2$
- $V^\pi(S_2) = 2.9 + 0.3 \times V^\pi(S_0)$
- $V^\pi(S_0) = 3.5 + 0.5 \times V^\pi(S_2)$
- But it is obvious, how to solve it:
  We have two equations and two variables $V^\pi(S_2)$ and $V^\pi(S_0)$!
- Now, you can do the calculation yourself. But what does this exercise tells you about computing value functions in general?

Example 1

What is the value function for the policy π s.t. $\pi(S_0) = a_0$, $\pi(S_1) = a_1$, $\pi(S_2) = a_2$, and $\pi(S_3) = a_0$?
Let's recapitulate:

- $V^\pi(S_3) = 0$
- $V^\pi(S_1) = 2$
- $V^\pi(S_2) = 2.9 + 0.3 \times V^\pi(S_0)$
- $V^\pi(S_0) = 3.5 + 0.5 \times V^\pi(S_2)$
- But it is obvious, how to solve it:
  We have two equations and two variables $V^\pi(S_2)$ and $V^\pi(S_0)$!
- Now, you can do the calculation yourself. But what does this exercise tells you about computing value functions in general?
- You can always write down all equations for all states and all transitions and solve it as a system of linear equations with variables $V^\pi(s)$.

Example 1

Now, let's move to optimal value function.

Example 1

Now, let's move to optimal value function.
Previously, we have shown that we can compute the value function for the given policy π as a system of linear equations.

Example 1

Now, let's move to optimal value function.
Previously, we have shown that we can compute the value function for the given policy π as a system of linear equations.
Can we use a similar approach with the optimal value function?

Example 1

Now, let's move to optimal value function.
Previously, we have shown that we can compute the value function for the given policy π as a system of linear equations.
Can we use a similar approach with the optimal value function? **Yes!**

Example 1

Now, let's move to optimal value function.
Previously, we have shown that we can compute the value function for the given policy π as a system of linear equations.
Can we use a similar approach with the optimal value function? **Yes!**
Before continuing, try to figure it out yourself. (There is no catch, really...)

Example 1

Now, let's move to optimal value function.
Previously, we have shown that we can compute the value function for the given policy π as a system of linear equations.
Can we use a similar approach with the optimal value function? **Yes!**

$$\begin{array}{rl}
\text{Variables} & V^\star(s) \in \mathbb{R}\ \forall s \in \mathcal{S} \\
\text{Minimize} & \sum_{s \in \mathcal{S}} V^\star(s) \\
\text{Constraints} & V^\star(s) \geq \sum_{s' \in \mathcal{S}} [\mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + V^\star(s')]]\ \forall s \in \mathcal{S}, \forall a \in \mathcal{A}
\end{array}$$

Example 1

Now, let's move to optimal value function.
Previously, we have shown that we can compute the value function for the given policy π as a system of linear equations.
Can we use a similar approach with the optimal value function? **Yes!**

Variables	$V^\star(s) \in \mathbb{R}\ \forall s \in \mathcal{S}$
Minimize	$\sum_{s\in\mathcal{S}} V^\star(s)$
Constraints	$V^\star(s) \geq \sum_{s'\in\mathcal{S}}[\mathcal{T}(s,a,s')[\mathcal{R}(s,a,s') + V^\star(s')]]\ \forall s \in \mathcal{S}, \forall a \in \mathcal{A}$

However, note how many constraints and variables we need. For big problems, it is often better (and much faster) to use some approximation approach like the *value-iteration algorithm*.

Value-Iteration

The idea of the value-iteration algorithm is very simple:

- We use dynamic programming, where we compute values of the value function from the values from the previous cycle.
- We initialize $V_0(s)$ to any number (e.g., zero could be the most convenient).
- In cycle n, we compute $V_n(s)$ as $\max_{a \in \mathcal{A}} \left[\sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s')[\mathcal{R}(s,a,s') + V_{n-1}(s')] \right]$, i.e., we treat the values from the previous cycle as the basis for the computation in the current cycle.
- We iterate until the biggest change in the current cycle is smaller than some small constant ϵ.
- It was shown that this process will eventually terminate for any ϵ (and any initialization) and it converges to the optimal value function.
- Pseudo-code is on the next slide.

Value-Iteration

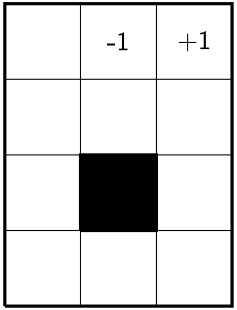
Algorithm 1: Value-Iteration

```
Input: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, a constant $\epsilon$
Output: Value function $V$ and policy $\pi$
1  $n \leftarrow 0$;
2  Initialize $V_0(s)$ arbitrarily for every $s \in \mathcal{S}$;
3  do
4      $n \leftarrow n + 1$;
5      foreach $s \in \mathcal{S}$ do
6          $V_n(s) \leftarrow \max_{a \in \mathcal{A}} \left[ \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + V_{n-1}(s')] \right]$;
7          $\text{residual}_n(s) \leftarrow |V_n(s) - V_{n-1}(s)|$;
8      end
9  while $\max_{s \in \mathcal{S}} \text{residual}_n(s) > \epsilon$;
10 $V \leftarrow V_n$;
11 foreach $s \in \mathcal{S}$ do
12     $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} \left[ \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{R}(s, a, s') + V_n(s')] \right]$;
13 end
```

Example 2

Consider the following maze-like example problem.

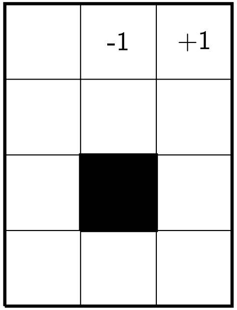
- The agent lives in the grid
- Walls block agent's movement



Example 2

Consider the following maze-like example problem.

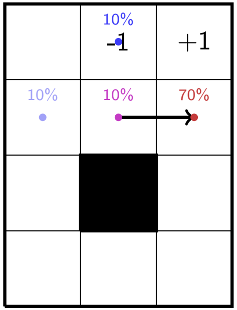
- The agent lives in the grid
- Walls block agent's movement
- The agent can move up, down, left, and right, but the action succeeds only with 70% probability and with 10% probability the agent ends up in one of the other three locations (and if blocked by a wall, the agent stays put).



Example 2

Consider the following maze-like example problem.

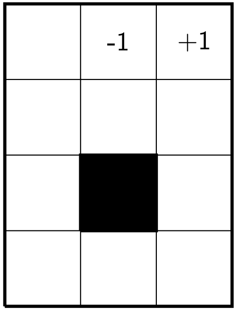
- The agent lives in the grid
- Walls block agent's movement
- The agent can move up, down, left, and right, but the action succeeds only with 70% probability and with 10% probability the agent ends up in one of the other three locations (and if blocked by a wall, the agent stays put).
- For example, if the agent move right, he ends up here with 70% probability, here with 10% probability, here with 10% probability (the movement down is blocked by the wall), and here with 10% probability,



Example 2

Consider the following maze-like example problem.

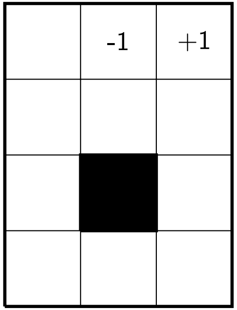
- The agent lives in the grid
- Walls block agent's movement
- The agent can move up, down, left, and right, but the action succeeds only with 70% probability and with 10% probability the agent ends up in one of the other three locations (and if blocked by a wall, the agent stays put).
- Reward for every action is 0, except for stepping on the tile "+1", where the reward is 1, and for stepping on the tile "-1", where the reward is -1.



Example 2

Consider the following maze-like example problem.

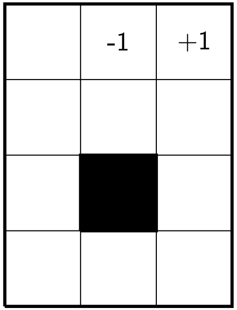
- The agent lives in the grid
- Walls block agent's movement
- The agent can move up, down, left, and right, but the action succeeds only with 70% probability and with 10% probability the agent ends up in one of the other three locations (and if blocked by a wall, the agent stays put).
- Reward for every action is 0, except for stepping on the tile "+1", where the reward is 1, and for stepping on the tile "-1", where the reward is -1.
- The "+1" and "-1" are terminal positions, meaning that once the agent steps there, it cannot move away.



Example 2

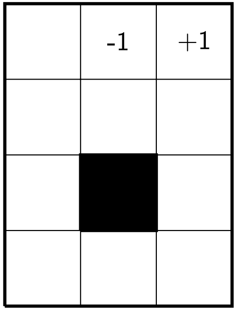
Consider the following maze-like example problem.

- The agent lives in the grid
- Walls block agent's movement
- The agent can move up, down, left, and right, but the action succeeds only with 70% probability and with 10% probability the agent ends up in one of the other three locations (and if blocked by a wall, the agent stays put).
- Reward for every action is 0, except for stepping on the tile "+1", where the reward is 1, and for stepping on the tile "-1", where the reward is -1.
- The "+1" and "-1" are terminal positions, meaning that once the agent steps there, it cannot move away.
- The question is what policy should we use to move the agent from any position to the most rewarding position ("+1").



Example 2

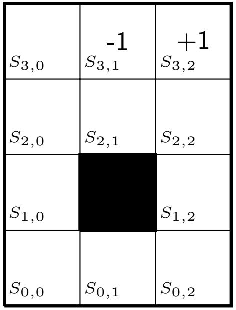
Before we proceed with the value-iteration algorithm, let's make it clear how this problem maps to MDP:



Example 2

Before we proceed with the value-iteration algorithm, let's make it clear how this problem maps to MDP:

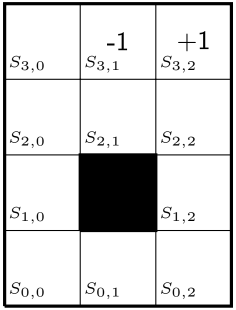
- A state is a position of the agent.



Example 2

Before we proceed with the value-iteration algorithm, let's make it clear how this problem maps to MDP:

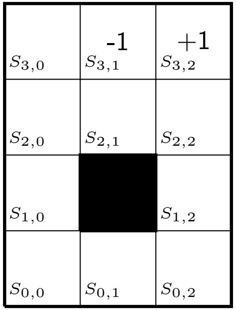
- A state is a position of the agent.
- We have actions a_{up}, a_{down}, a_{left}, and a_{right} with
  - $\mathcal{T}(S_{0,0}, a_{\text{up}}, S_{1,0}) = 0.7$
  - $\mathcal{T}(S_{0,0}, a_{\text{up}}, S_{0,1}) = 0.1$
  - $\mathcal{T}(S_{0,0}, a_{\text{up}}, S_{0,0}) = 0.2$ (movement down and left is blocked)
  - ...
  - $\mathcal{T}(S_{2,0}, a_{\text{right}}, S_{2,1}) = 0.7$
  - $\mathcal{T}(S_{2,0}, a_{\text{up}}, S_{3,0}) = 0.1$
  - $\mathcal{T}(S_{2,0}, a_{\text{up}}, S_{1,0}) = 0.1$
  - $\mathcal{T}(S_{2,0}, a_{\text{up}}, S_{2,0}) = 0.1$ (movement left is blocked)
  - ...



Example 2

Before we proceed with the value-iteration algorithm, let's make it clear how this problem maps to MDP:

- A state is a position of the agent.
- We have actions a_{up}, a_{down}, a_{left}, and a_{right}
- The reward function assigns 1 when stepping on "+1" and -1 when stepping on "-1".
  - $\mathcal{R}(S_{3,0}, a_{\text{right}}, S_{3,1}) = -1$
  - $\mathcal{R}(S_{2,1}, a_{\text{up}}, S_{3,1}) = -1$
  - $\mathcal{R}(S_{2,2}, a_{\text{up}}, S_{3,2}) = 1$
  - and zero for everywhere else.

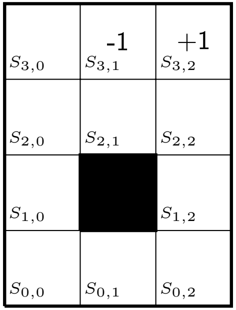


Example 2

Before we proceed with the value-iteration algorithm, let's make it clear how this problem maps to MDP:

- A state is a position of the agent.
- We have actions a_{up}, a_{down}, a_{left}, and a_{right}
- The reward function assigns 1 when stepping on "+1" and -1 when stepping on "-1".

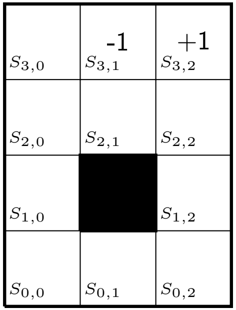
How do we compute the *optimal* value for the state $S_{2,1}$?



Example 2

How do we compute the *optimal* value for the state $S_{2,1}$?

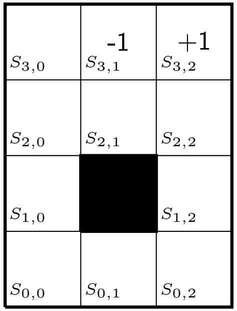
$$\begin{aligned}
V^\star(S_{2,1}) = \max \big[& \mathcal{T}(S_{2,1}, a_{\text{up}}, S_{3,1})[\mathcal{R}(S_{2,1}, a_{\text{up}}, S_{3,1}) + V^\star(S_{3,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{up}}, S_{2,0})[\mathcal{R}(S_{2,1}, a_{\text{up}}, S_{2,0}) + V^\star(S_{2,0})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{up}}, S_{2,1})[\mathcal{R}(S_{2,1}, a_{\text{up}}, S_{2,1}) + V^\star(S_{2,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{up}}, S_{2,2})[\mathcal{R}(S_{2,1}, a_{\text{up}}, S_{2,2}) + V^\star(S_{2,2})], \\
& \mathcal{T}(S_{2,1}, a_{\text{down}}, S_{3,1})[\mathcal{R}(S_{2,1}, a_{\text{down}}, S_{3,1}) + V^\star(S_{3,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{down}}, S_{2,0})[\mathcal{R}(S_{2,1}, a_{\text{down}}, S_{2,0}) + V^\star(S_{2,0})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{down}}, S_{2,1})[\mathcal{R}(S_{2,1}, a_{\text{down}}, S_{2,1}) + V^\star(S_{2,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{down}}, S_{2,2})[\mathcal{R}(S_{2,1}, a_{\text{down}}, S_{2,2}) + V^\star(S_{2,2})], \\
& \mathcal{T}(S_{2,1}, a_{\text{left}}, S_{3,1})[\mathcal{R}(S_{2,1}, a_{\text{left}}, S_{3,1}) + V^\star(S_{3,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{left}}, S_{2,0})[\mathcal{R}(S_{2,1}, a_{\text{left}}, S_{2,0}) + V^\star(S_{2,0})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{left}}, S_{2,1})[\mathcal{R}(S_{2,1}, a_{\text{left}}, S_{2,1}) + V^\star(S_{2,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{left}}, S_{2,2})[\mathcal{R}(S_{2,1}, a_{\text{left}}, S_{2,2}) + V^\star(S_{2,2})], \\
& \mathcal{T}(S_{2,1}, a_{\text{right}}, S_{3,1})[\mathcal{R}(S_{2,1}, a_{\text{right}}, S_{3,1}) + V^\star(S_{3,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{right}}, S_{2,0})[\mathcal{R}(S_{2,1}, a_{\text{right}}, S_{2,0}) + V^\star(S_{2,0})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{right}}, S_{2,1})[\mathcal{R}(S_{2,1}, a_{\text{right}}, S_{2,1}) + V^\star(S_{2,1})] \\
& + \mathcal{T}(S_{2,1}, a_{\text{right}}, S_{2,2})[\mathcal{R}(S_{2,1}, a_{\text{right}}, S_{2,2}) + V^\star(S_{2,2})],
\end{aligned}$$



Example 2

How do we compute the *optimal* value for the state $S_{2,1}$?

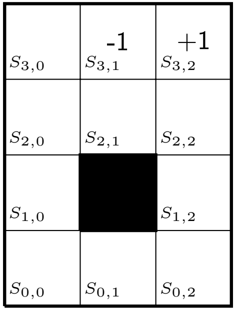
$$
\begin{aligned}
V^\star(S_{2,1}) = \max\,[&0.7 \times [-1 + V^\star(S_{3,1})] \\
&+ 0.1 \times [0 + V^\star(S_{2,0})] \\
&+ 0.1 \times [0 + V^\star(S_{2,1})] \\
&+ 0.1 \times [0 + V^\star(S_{2,2})] \\
&0.1 \times [-1 + V^\star(S_{3,1})] \\
&+ 0.1 \times [0 + V^\star(S_{2,0})] \\
&+ 0.7 \times [0 + V^\star(S_{2,1})] \\
&+ 0.1 \times [0 + V^\star(S_{2,2})], \\
&0.1 \times [-1 + V^\star(S_{3,1})] \\
&+ 0.7 \times [0 + V^\star(S_{2,0})] \\
&+ 0.1 \times [0 + V^\star(S_{2,1})] \\
&+ 0.1 \times [0 + V^\star(S_{2,2})], \\
&0.1 \times [-1 + V^\star(S_{3,1})] \\
&+ 0.1 \times [0 + V^\star(S_{2,0})] \\
&+ 0.1 \times [0 + V^\star(S_{2,1})] \\
&+ 0.7 \times [0 + V^\star(S_{2,2})], \\
&]
\end{aligned}
$$



Example 2

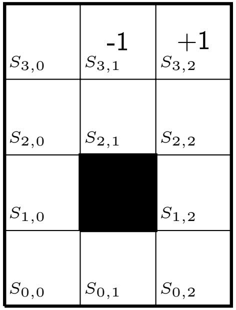
How do we compute the *optimal* value for the state $S_{2,1}$?

$$\begin{aligned}V^\star(S_{2,1}) = \max\big[&0.7\times[-1+V^\star(S_{3,1})]+0.1\times[V^\star(S_{2,0})+V^\star(S_{2,1})+V^\star(S_{2,2})],\\ &0.7\times V^\star(S_{2,1})+0.1\times[-1+V^\star(S_{3,1})+V^\star(S_{2,0})+V^\star(S_{2,2})],\\ &0.7\times V^\star(S_{2,0})+0.1\times[-1+V^\star(S_{3,1})+V^\star(S_{2,1})+V^\star(S_{2,2})],\\ &0.7\times V^\star(S_{2,2})+0.1\times[-1+V^\star(S_{3,1})+V^\star(S_{2,0})+V^\star(S_{2,1})]\\ &\big]\end{aligned}$$



Example 2

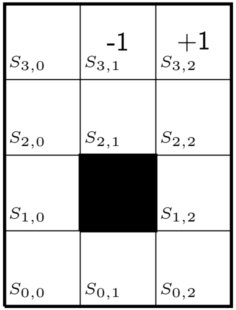
And how do we compute the V_n values in the n'th step of the value-iteration algorithm?



Example 2

And how do we compute the V_n values in the n'th step of the value-iteration algorithm?

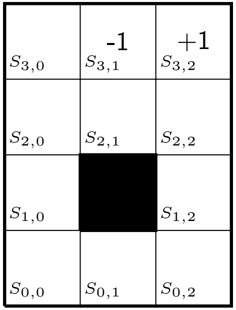
$$\begin{aligned}
V_n(S_{2,1}) = \max\, [&0.7 \times [-1 + V_{n-1}(S_{3,1})] + 0.1 \times [V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})],\\
&0.7 \times V_{n-1}(S_{2,1}) + 0.1 \times [-1 + V_{n-1}(S_{3,1}) + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,2})],\\
&0.7 \times V_{n-1}(S_{2,0}) + 0.1 \times [-1 + V_{n-1}(S_{3,1}) + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})],\\
&0.7 \times V_{n-1}(S_{2,2}) + 0.1 \times [-1 + V_{n-1}(S_{3,1}) + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1})]\\
&]
\end{aligned}$$



Example 2

And how do we compute the V_n values in the n'th step of the value-iteration algorithm?

$$\begin{aligned} V_n(S_{2,1}) = \max\ [&0.7 \times [-1 + V_{n-1}(S_{3,1})] + 0.1 \times [V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})], \\ &0.7 \times V_{n-1}(S_{2,1}) + 0.1 \times [-1 + V_{n-1}(S_{3,1}) + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,2})], \\ &0.7 \times V_{n-1}(S_{2,0}) + 0.1 \times [-1 + V_{n-1}(S_{3,1}) + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})], \\ &0.7 \times V_{n-1}(S_{2,2}) + 0.1 \times [-1 + V_{n-1}(S_{3,1}) + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1})] \\ &] \end{aligned}$$

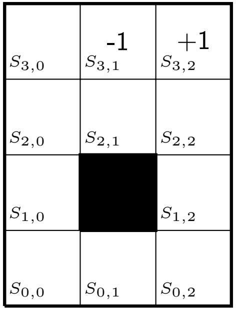


Note also that the value for the terminal states $S_{3,1}$ and $S_{3.2}$ must be zero.

Example 2

And how do we compute the V_n values in the n'th step of the value-iteration algorithm?

$$\begin{aligned}V_n(S_{2,1}) = \max\,[&0.7 \times [-1+0] + 0.1 \times [V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})],\\ &0.7 \times V_{n-1}(S_{2,1}) + 0.1 \times [-1 + 0 + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,2})],\\ &0.7 \times V_{n-1}(S_{2,0}) + 0.1 \times [-1 + 0 + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})],\\ &0.7 \times V_{n-1}(S_{2,2}) + 0.1 \times [-1 + 0 + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1})]\\ &]\end{aligned}$$

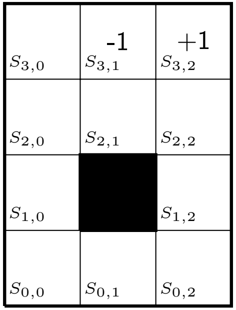


Note also that the value for the terminal states $S_{3,1}$ and $S_{3.2}$ must be zero.

Example 2

And how do we compute the V_n values in the n'th step of the value-iteration algorithm?

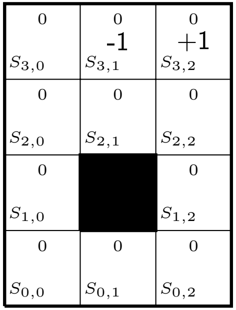
$$\begin{aligned} V_n(S_{2,1}) = \max [&0.7 \times [-1 + 0] + 0.1 \times [V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})], \\ &0.7 \times V_{n-1}(S_{2,1}) + 0.1 \times [-1 + 0 + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,2})], \\ &0.7 \times V_{n-1}(S_{2,0}) + 0.1 \times [-1 + 0 + V_{n-1}(S_{2,1}) + V_{n-1}(S_{2,2})], \\ &0.7 \times V_{n-1}(S_{2,2}) + 0.1 \times [-1 + 0 + V_{n-1}(S_{2,0}) + V_{n-1}(S_{2,1})] \\ &] \end{aligned}$$



Note also that the value for the terminal states $S_{3,1}$ and $S_{3.2}$ must be zero.
Now we know everything we need to execute the value-iteration algorithm.

Example 2

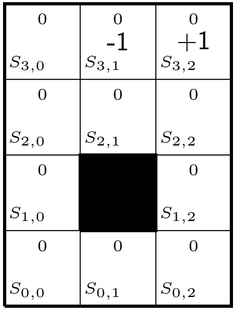
We start by initializating $V_0(s)$ to zero for every state.



Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

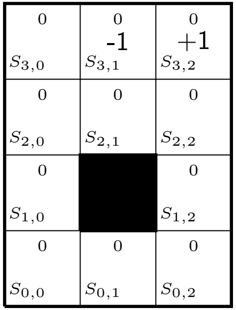
$$
\begin{aligned}
V_1(S_{2,1}) = \max \big[& 0.7 \times [-1+0] + 0.1 \times [V_0(S_{2,0}) + V_0(S_{2,1}) + V_0(S_{2,2})], \\
& 0.7 \times V_0(S_{2,1}) + 0.1 \times [-1 + 0 + V_0(S_{2,0}) + V_0(S_{2,2})], \\
& 0.7 \times V_0(S_{2,0}) + 0.1 \times [-1 + 0 + V_0(S_{2,1}) + V_0(S_{2,2})], \\
& 0.7 \times V_0(S_{2,2}) + 0.1 \times [-1 + 0 + V_0(S_{2,0}) + V_0(S_{2,1})] \\
& \big]
\end{aligned}
$$



Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

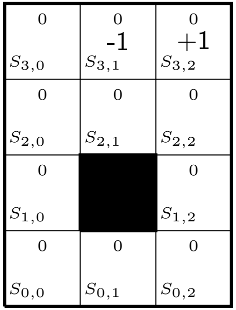
$$\begin{aligned}V_1(S_{2,1}) = \max \big[&0.7 \times [-1+0] + 0.1 \times 0,\\ &0.7 \times 0 + 0.1 \times -1,\\ &0.7 \times 0 + 0.1 \times -1,\\ &0.7 \times 0 + 0.1 \times -1\\ &\big]\end{aligned}$$



Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

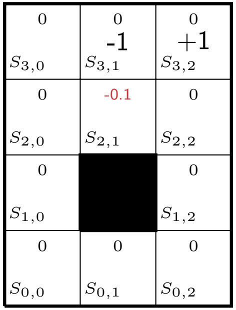
$$
\begin{aligned}
V_1(S_{2,1}) = \max \Big[& 0.7 \times [-1 + 0] + 0.1 \times 0, \\
& 0.7 \times 0 + 0.1 \times -1, \\
& 0.7 \times 0 + 0.1 \times -1, \\
& 0.7 \times 0 + 0.1 \times -1 \\
& \Big]
\end{aligned}
$$



Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

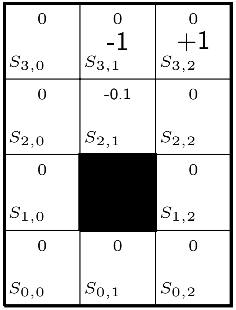


Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\big[-0.7, -0.1, -0.1, -0.1\big] = -0.1$$

Now, repeat the same step for every state by yourself...

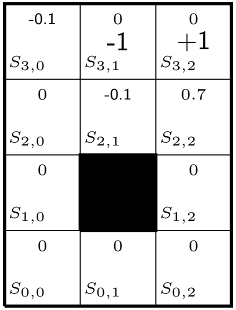


Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\big[-0.7, -0.1, -0.1, -0.1\big] = -0.1$$

Now, repeat the same step for every state by yourself...

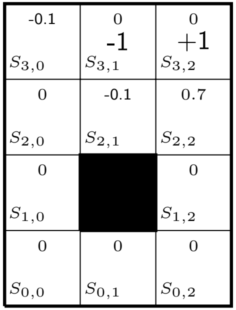


Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s\in\mathcal{S}} \text{residual}_1(s) = .7$.

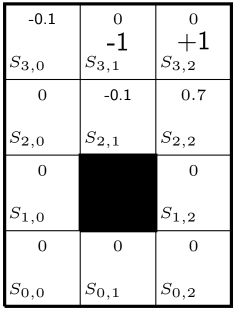


Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s\in\mathcal{S}} \text{residual}_1(s) = .7$.
Next, we use the values V_1 for computing V_2...

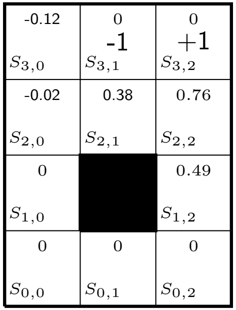


Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s\in\mathcal{S}} \text{residual}_1(s) = .7$.
Next, we use the values V_1 for computing V_2...

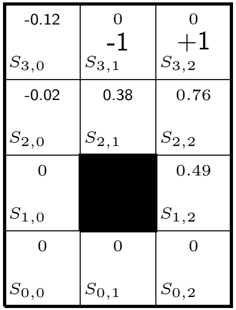


Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s \in \mathcal{S}} \mathrm{residual}_1(s) = .7$.
Next, we use the values V_1 for computing V_2...
Next, we compute V_3 using V_2...



Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s \in \mathcal{S}} \text{residual}_1(s) = .7$.
Next, we use the values V_1 for computing V_2...
Next, we compute V_3 using V_2...

-0.138 $S_{3,0}$	0 -1 $S_{3,1}$	0 +1 $S_{3,2}$
0.252 $S_{2,0}$	0.468 $S_{2,1}$	0.863 $S_{2,2}$
-0.002 $S_{1,0}$	⬛	0.581 $S_{1,2}$
0 $S_{0,0}$	0 $S_{0,1}$	0.343 $S_{0,2}$

Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s \in \mathcal{S}} \text{residual}_1(s) = .7$.
Next, we use the values V_1 for computing V_2...
Next, we compute V_3 using V_2...
What is the resulting policy π given V_3? Recall line 12 of the algorithm: $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} \left[\sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s')[\mathcal{R}(s,a,s') + V_n(s')]\right]$

-0.138 $S_{3,0}$	0 -1 $S_{3,1}$	0 +1 $S_{3,2}$
0.252 $S_{2,0}$	0.468 $S_{2,1}$	0.863 $S_{2,2}$
-0.002 $S_{1,0}$	■	0.581 $S_{1,2}$
0 $S_{0,0}$	0 $S_{0,1}$	0.343 $S_{0,2}$

Example 2

We start by initializating $V_0(s)$ to zero for every state.
Now we can compute V_1.
For $S_{2,1}$ we get:

$$V_1(S_{2,1}) = \max\left[-0.7, -0.1, -0.1, -0.1\right] = -0.1$$

Now, repeat the same step for every state by yourself...
The maximum of residuals is $\max_{s \in \mathcal{S}} \text{residual}_1(s) = .7$.
Next, we use the values V_1 for computing V_2...
Next, we compute V_3 using V_2...
What is the resulting policy π given V_3? Recall line 12 of the algorithm: $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} \left[\sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s')[\mathcal{R}(s,a,s') + V_n(s')]\right]$
For example the following...

-0.138 a_{down} $S_{3,0}$	0 -1 $S_{3,1}$	0 +1 $S_{3,2}$
0.252 a_{right} $S_{2,0}$	0.468 a_{right} $S_{2,1}$	0.863 a_{up} $S_{2,2}$
-0.002 a_{down} $S_{1,0}$	⬛	0.581 a_{up} $S_{1,2}$
0 a_{up} $S_{0,0}$	0 a_{up} $S_{0,1}$	0.343 a_{up} $S_{0,2}$

Example 2

Now, implement this algorithm (for this particular example). You will see that only after 8 cycles, the residual gets below 0.1 and the value function and resulting policy is as follows.

0.3598 a_{down} $S_{3,0}$	0 -1 $S_{3,1}$	0 +1 $S_{3,2}$
0.5971 a_{right} $S_{2,0}$	0.678 a_{right} $S_{2,1}$	0.9395 a_{up} $S_{2,2}$
0.4871 a_{up} $S_{1,0}$	■	0.8072 a_{up} $S_{1,2}$
0.5643 a_{right} $S_{0,0}$	0.6805 a_{right} $S_{0,1}$	0.7635 a_{up} $S_{0,2}$

Example 2

Now, implement this algorithm (for this particular example). You will see that only after 8 cycles, the residual gets below 0.1 and the value function and resulting policy is as follows.

Is this the best policy we can get?

0.3598 a_{down} $S_{3,0}$	0 -1 $S_{3,1}$	0 $+1$ $S_{3,2}$
0.5971 a_{right} $S_{2,0}$	0.678 a_{right} $S_{2,1}$	0.9395 a_{up} $S_{2,2}$
0.4871 a_{up} $S_{1,0}$	■	0.8072 a_{up} $S_{1,2}$
0.5643 a_{right} $S_{0,0}$	0.6805 a_{right} $S_{0,1}$	0.7635 a_{up} $S_{0,2}$

Example 2

Now, implement this algorithm (for this particular example). You will see that only after 8 cycles, the residual gets below 0.1 and the value function and resulting policy is as follows.

Is this the best policy we can get?
What about these actions?

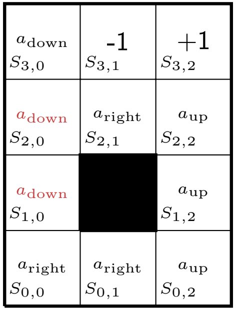
0.3598 a_{down} $S_{3,0}$	0 -1 $S_{3,1}$	0 +1 $S_{3,2}$
0.5971 a_{right} $S_{2,0}$	0.678 a_{right} $S_{2,1}$	0.9395 a_{up} $S_{2,2}$
0.4871 a_{up} $S_{1,0}$	■	0.8072 a_{up} $S_{1,2}$
0.5643 a_{right} $S_{0,0}$	0.6805 a_{right} $S_{0,1}$	0.7635 a_{up} $S_{0,2}$

Example 2

Now, implement this algorithm (for this particular example). You will see that only after 8 cycles, the residual gets below 0.1 and the value function and resulting policy is as follows.

Is this the best policy we can get?
What about these actions?

If you run your implementation longer, you will find that eventually we arrive at this policy.



a_{down} $S_{3,0}$
-1 $S_{3,1}$
+1 $S_{3,2}$
a_{down} $S_{2,0}$
a_{right} $S_{2,1}$
a_{up} $S_{2,2}$
a_{down} $S_{1,0}$
a_{up} $S_{1,2}$
a_{right} $S_{0,0}$
a_{right} $S_{0,1}$
a_{up} $S_{0,2}$