# Reinforcement Learning

## Lecture 2 : Dynamic Programming

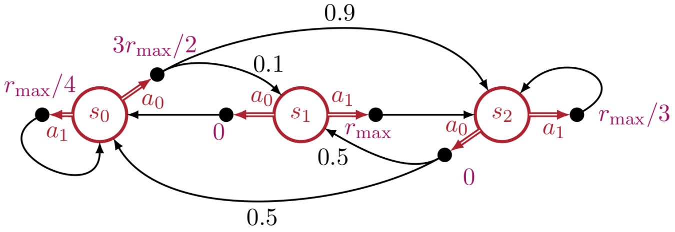
Emilie Kaufmann

Ecole Centrale de Lille, 2021/2022

# Reminder : Markov Decision Process

A MDP is parameterized by a tuple $(\mathcal{S}, \mathcal{A}, R, P)$ where

- $\mathcal{S}$ is the state space
- $\mathcal{A}$ is the action space (or $\mathcal{A}_s$ for each $s \in \mathcal{S}$)
- $R = (\nu_{(s,a)})_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ where $\nu_{(s,a)} \in \Delta(\mathbb{R})$ is the reward distribution for the state-action pair $(s, a)$ $\rightarrow r(s,a) = \mathbb{E}_{R\sim\nu_{(s,a)}}[R]$
- $P = (p(\cdot|s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ where $p(\cdot|s,a) \in \Delta(\mathcal{S})$ is the transition kernel associated to the state-action pair $(s, a)$



# Reminder : Policy

A policy $\pi = (\pi_0, \pi_1, \dots)$ is a sequence of mapping $\pi_t : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ that maps a state to a distribution over actions.

**Under policy $\boldsymbol{\pi}$**, at time $t$, the agent in state $s_t$ selects

$$a_t \sim \pi_t(s_t),$$

receives the instantaneous reward

$$r_t \sim \nu_{(s_t,a_t)} \text{ such that } \mathbb{E}[r_t|s_t, a_t] = r(s_t, a_t)$$

and transits to the new state $s_{t+1} \sim p(\cdot|s_t, a_t)$.

➜ a policy defines a probability model $\mathbb{P}^\pi, \mathbb{E}^\pi$ over sequences of observations :

$$s_0, a_0, r_0, s_1, a_1, r_1, \dots$$

# Reminder : Value Function

**Definition**

The value function of a policy $\pi = (\pi_0, \pi_1, \pi_2, \dots)$ is $V^\pi : \mathcal{S} \to \mathbb{R}$

① Finite-horizon criterion

$$V^\pi(s) = \mathbb{E}^\pi \left[ \sum_{t=1}^{H} r_t \middle| s_0 = s \right]$$

➜ We want to compute the optimal value $V^\star(s) = \max_\pi V^\pi(s)$ and an optimal policy $\pi_\star$ such that $V^\star = V^{\pi_\star}$.

➜ We will be able to do so when $\mathcal{S}$ and $\mathcal{A}$ are finite

$$S := |\mathcal{S}| < \infty \quad \text{and} \quad A := |\mathcal{A}| < \infty$$

*(some optimality equation may extend to continuous state spaces)*

# Reminder : Value Function

## Definition

The value function of a policy $\pi = (\pi_0, \pi_1, \pi_2, \dots)$ is $V^\pi : \mathcal{S} \to \mathbb{R}$

① Finite-horizon criterion

$$V^\pi(s) = \mathbb{E}^\pi \left[ \sum_{t=1}^{H} r(s_t, a_t) \middle| s_0 = s \right]$$

- ➜ We want to compute the optimal value $V^\star(s) = \max_\pi V^\pi(s)$ and an optimal policy $\pi_\star$ such that $V^\star = V^{\pi_\star}$.
- ➜ We will be able to do so when $\mathcal{S}$ and $\mathcal{A}$ are finite

$$S := |\mathcal{S}| < \infty \quad \text{and} \quad A := |\mathcal{A}| < \infty$$

*(some optimality equation may extend to continuous state spaces)*

# Reminder : Value Function

## Definition

The value function of a policy $\pi = (\pi_0, \pi_1, \pi_2, \dots)$ is $V^\pi : \mathcal{S} \to \mathbb{R}$

② Infinite horizon with a discount $\gamma$

$$V^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t) \middle| s_0 = s\right]$$

- We want to compute the optimal value $V^\star(s) = \max_\pi V^\pi(s)$ and an optimal policy $\pi_\star$ such that $V^\star = V^{\pi_\star}$.
- We will be able to do so when $\mathcal{S}$ and $\mathcal{A}$ are finite

$$S := |\mathcal{S}| < \infty \quad \text{and} \quad A := |\mathcal{A}| < \infty$$

*(some optimality equation may extend to continuous state spaces)*

# Outline

# Value functions

Let $H$ be the known time horizon.

## Value functions at step $h$

For a policy $\pi = (\pi_1, \ldots, \pi_H)$,

$$V_h^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=h}^{H} r_t \middle| s_h = s\right]$$

and

$$V_h^\star(s) = \max_{\pi_h,\ldots,\pi_H} \mathbb{E}^\pi\left[\sum_{t=h}^{H} r_t \middle| s_h = s\right]$$

**Goal :** compute

$$V^\pi(s) = V_1^\pi(s), V^\star(s) = V_1^\star(s) \quad \text{and} \quad \pi^\star = (\pi_1^\star, \ldots, \pi_H^\star).$$

➜ we will actually compute $V_h^\pi(s)$ and $V_h^\star(s)$ for all $h \leq H$.

# Value functions

Let $H$ be the known time horizon.

### Value functions at step $h$

For a policy $\pi = (\pi_1, \dots, \pi_H)$,

$$V_h^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=h}^{H} r(s_t, a_t) \middle| s_h = s\right]$$

and

$$V_h^\star(s) = \max_{\pi_h, \dots, \pi_H} \mathbb{E}^\pi\left[\sum_{t=h}^{H} r(s_t, a_t) \middle| s_h = s\right]$$

**Goal :** compute

$$V^\pi(s) = V_1^\pi(s), V^\star(s) = V_1^\star(s) \quad \text{and} \quad \pi^\star = (\pi_1^\star, \dots, \pi_H^\star).$$

➜ we will actually compute $V_h^\pi(s)$ and $V_h^\star(s)$ for all $h \leq H$.

# Value functions

Let $H$ be the known time horizon.

## Value functions at step $h$

For a deterministic policy $\pi = (\pi_1, \dots, \pi_H)$,

$$V_h^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=h}^{H} r(s_t, \pi_t(s_t)) \middle| s_h = s\right]$$

and

$$V_h^\star(s) = \max_{\pi_h, \dots, \pi_H} \mathbb{E}^\pi\left[\sum_{t=h}^{H} r(s_t, \pi_t(s_t)) \middle| s_h = s\right]$$

**Goal :** compute

$$V^\pi(s) = V_1^\pi(s), V^\star(s) = V_1^\star(s) \quad \text{and} \quad \pi^\star = (\pi_1^\star, \dots, \pi_H^\star).$$

➜ we will actually compute $V_h^\pi(s)$ and $V_h^\star(s)$ for all $h \leq H$.

# Value functions

Let $H$ be the known time horizon.

## Value functions at step $h$

For a policy $\pi = (\pi_1, \dots, \pi_H)$,

$$V_h^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=h}^{H} r_t \middle| s_h = s\right]$$

and

$$V_h^\star(s) = \max_{\pi_h,\dots,\pi_H} \mathbb{E}^\pi\left[\sum_{t=h}^{H} r_t \middle| s_h = s\right]$$

**How ?**

- Monte-Carlo estimation ? only approximate
- Develop the tree of all possible realizations ? too complex

# Bellman equations

**Proposition**

The value functions of a deterministic policy $\pi$ satisfies the following equations : for all $h \in \{1, \ldots, H\}$,

$$V_h^\pi(s) \quad = \quad r(s, \pi_h(s)) + \sum_{s' \in \mathcal{S}} p(s'|s, \pi_h(s)) V_{h+1}^\pi(s'),$$

with the convention that $V_{H+1}^\pi(s) = 0$ for all $s \in \mathcal{S}$.

**Consequence :** for a finite state space $\mathcal{S}$ such that $|\mathcal{S}| = S$

- $V_1^\pi(s)$ can be computed using backwards induction
- space complexity : $S \times H$
- time complexity : $S \times (S+1) \times H$

# Bellman equations

**Proposition**

The value functions of a deterministic policy $\pi$ satisfies the following equations : for all $h \in \{1, \dots, H\}$,

$$V_h^\pi(s) \quad = \quad r(s, \pi_h(s)) + \sum_{s' \in \mathcal{S}} p(s'|s, \pi_h(s)) V_{h+1}^\pi(s'),$$

with the convention that $V_{H+1}^\pi(s) = 0$ for all $s \in \mathcal{S}$.

**Proof.**

$$V_h^\pi(s) = \mathbb{E}^\pi \left[ r(s_h, \pi_h(s_h)) + \sum_{t=h+1}^{H} r(s_t, \pi_t(s_t)) \middle| s_h = s \right]$$

$$= r(s, \pi_h(s)) + \mathbb{E}^\pi \left[ \sum_{t=h+1}^{H} r(s_t, \pi_t(s_t)) \middle| s_h = s, a_h = \pi_h(s) \right]$$

$$= r(s, \pi_h(s)) + \sum_{s' \in \mathcal{S}} \mathbb{P}(s_{h+1} = s' | s_h = s, a_h = \pi_h(s)) \mathbb{E}^\pi \left[ \sum_{t=h+1}^{H} r(s_t, \pi_t(s_t)) \middle| s_{h+1} = s' \right]$$

$$= r(s, \pi_h(s)) + \sum_{s' \in \mathcal{S}} p(s'|s, \pi_h(s)) V_{h+1}^\pi(s')$$

# Bellman equations

**Proposition**

The value functions of a deterministic policy $\pi$ satisfies the following equations : for all $h \in \{1, \dots, H\}$,

$$V_h^\pi(s) \quad = \quad r(s, \pi_h(s)) + \mathbb{E}_{s' \sim p(\cdot|s,\pi_h(s))}\left[V_{h+1}^\pi(s')\right],$$

with the convention that $V_{H+1}^\pi(s) = 0$ for all $s \in \mathcal{S}$.

These equations may be generalized :

- to a possibly infinite state space

# Bellman equations

**Proposition**

The value functions of a policy $\pi$ satisfies the following equations : for all $h \in \{1, \ldots, H\}$,

$$V_h^\pi(s) \quad = \quad \mathbb{E}_{a \sim \pi_h(s)}\left[r(s,a) + \mathbb{E}_{s' \sim p(\cdot|s,a)}\left[V_{h+1}^\pi(s')\right]\right],$$

with the convention that $V_{H+1}^\pi(s) = 0$ for all $s \in \mathcal{S}$.

These equations may be generalized :

- to a possibly infinite state space
- to randomized policies

# Bellman equations for the optimal values

## Proposition

The optimal values $V_h^\star$ satisfy the **Bellman equations** :

$$V_h^\star(s) = \max_{a \in \mathcal{A}} \left[ r(s,a) + \sum_{s' \in \mathcal{S}} p(s'|s,a) V_{h+1}^\star(s') \right] \quad \text{for all } h \leq H,$$

with the convention that $V_{H+1}^\star(s) = 0$ for all $s \in \mathcal{S}$.
Moreover, an optimal policy is given by

$$\pi_h^\star(s) \in \operatorname*{argmax}_{a \in \mathcal{A}} \left[ r(s,a) + \sum_{s'=1}^{S} p(s'|s,a) V_{h+1}^\star(s') \right].$$

# Bellman equations for the optimal values

**Proposition**

The optimal values $V_h^\star$ satisfy the **Bellman equations** :

$$V_h^\star(s) = \max_{a \in \mathcal{A}} \left[ r(s,a) + \sum_{s' \in \mathcal{S}} p(s'|s,a) V_{h+1}^\star(s') \right] \quad \text{for all } h \leq H,$$

with the convention that $V_{H+1}^\star(s) = 0$ for all $s \in \mathcal{S}$.
Moreover, an optimal policy is given by

$$\pi_h^\star(s) \in \operatorname*{argmax}_{a \in \mathcal{A}} \left[ r(s,a) + \sum_{s'=1}^{S} p(s'|s,a) V_{h+1}^\star(s') \right].$$

**Consequence :** for finite $\mathcal{S}, \mathcal{A}$ such that $|\mathcal{S}| = S$, $|\mathcal{A}| = A$

- ➜ $\pi^\star = (\pi_1^\star, \dots, \pi_H^\star)$ can be computed using backwards induction
- ➜ space complexity : $S \times H$
- ➜ time complexity : $O(S^2 A H)$

# Bellman equations for the optimal values

**Proposition**

The optimal values $V_h^\star$ satisfy the **Bellman equations** :

$$V_h^\star(s) = \max_{a\in\mathcal{A}} \left[ r(s,a) + \sum_{s'\in\mathcal{S}} p(s'|s,a) V_{h+1}^\star(s') \right] \quad \text{for all } h \leq H,$$

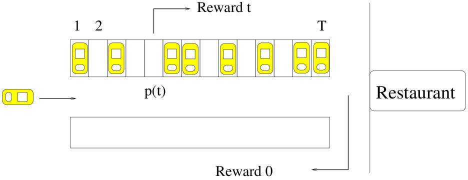
with the convention that $V_{H+1}^\star(s) = 0$ for all $s \in \mathcal{S}$.
Moreover, an optimal policy is given by

$$\pi_h^\star(s) \in \mathop{\mathrm{argmax}}_{a\in\mathcal{A}} \left[ r(s,a) + \sum_{s'=1}^{S} p(s'|s,a) V_{h+1}^\star(s') \right].$$

This technique is known as **Dynamic Programming**

- term invented in the 50s by Bellman : an algorithmic principle for optimization in which solving an optimization problem of a given size reduces to solving (several) of the same optimization problem but of smaller size

# Example : Optimal Parking




Exercise :

- ➜ model optimal parking as solving a MDP with a finite horizon
- ➜ write the optimal policy

# Outline

# Values functions

Let $\gamma \in (0,1)$ be a known discount factor

## Value functions

For a policy $\pi = (\pi_1, \pi_2, \dots)$,

$$V^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \,\middle|\, s_1 = s\right]$$

and

$$V^\star(s) = \max_\pi \mathbb{E}^\pi\left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \,\middle|\, s_1 = s\right]$$

**How to compute them ?**

➜ We need to generalize Dynamic Programming to infinite horizon...

# Values functions

Let $\gamma \in (0,1)$ be a known discount factor

## Value functions

For a deterministic policy $\pi = (\pi_1, \pi_2, \dots)$,

$$V^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, \pi_t(s_t)) \middle| s_1 = s\right]$$

and

$$V^\star(s) = \max_\pi \mathbb{E}^\pi\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, \pi_t(s_t)) \middle| s_1 = s\right]$$

**How to compute them ?**

➜ We need to generalize Dynamic Programming to infinite horizon...

# Outline

# Bellman equation for a stationary policy

**Proposition**

Any stationary deterministic policy $\pi$ satisfies, for all $s \in \mathcal{S}$,

$$V^{\pi}(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) V^{\pi}(s')$$

**Proof.**

$$\begin{aligned}
V^{\pi}(s) &= \mathbb{E}^{\pi}\left[ r(s,\pi(s)) + \sum_{t=2}^{\infty} \gamma^{t-1} r(s_t, \pi(s_t)) \middle| s_1 = s \right] \\
&= r(s,\pi(s)) + \gamma \mathbb{E}^{\pi}\left[ \sum_{t=1}^{\infty} \gamma^{t-2} r(s_t, \pi(s_t)) \middle| s_1 = s, a_1 = \pi(s) \right] \\
&= r(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s_2 = s' | s_1 = s, a_1 = \pi(s)) \mathbb{E}^{\pi}\left[ \sum_{t=2}^{\infty} \gamma^{t-2} r(s_t, \pi(s_t)) \middle| s_2 = s' \right] \\
&= r(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) \underbrace{\mathbb{E}^{\pi}\left[ \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, \pi(s_t)) \middle| s_1 = s' \right]}_{V^{\pi}(s')}
\end{aligned}$$

# Bellman equation for a stationary policy

**Proposition**

Any stationary deterministic policy $\pi$ satisfies, for all $s \in \mathcal{S}$,

$$V^\pi(s) = r(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,\pi(s)) V^\pi(s')$$

More general statement :

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(s)} \left[ r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \left[ V^\pi(s') \right] \right]$$

(also applies to infinite state space and randomized policies)

# Solving the Bellman equations

Fix a stationary, deterministic policy $\pi$.

**Proposition**

$$\forall s \in \mathcal{S}, \quad V^{\pi}(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) V^{\pi}(s')$$

# Solving the Bellman equations

Fix a stationary, deterministic policy $\pi$.

**Proposition**

$$\forall s \in \mathcal{S}, \quad V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, \pi(s)) V^\pi(s')$$

Introducing the vectors

$$\begin{aligned} V^\pi &= (V^\pi(s))_{s=1}^S \in \mathbb{R}^S \\ r^\pi &= (r(s, \pi(s))_{s=1}^S \in \mathbb{R}^S \end{aligned}$$

and the matrix

$$P^\pi = \Big(p(s'|s, \pi(s))\Big)_{\substack{1 \le s \le S \\ 1 \le s' \le S}} \in \mathbb{R}^{S \times S},$$

the Bellman equations rewrite

$$\boldsymbol{V^\pi = r^\pi + \gamma P^\pi V^\pi}$$

# Solving the Bellman equations

$$V^\pi = r^\pi + \gamma P^\pi V^\pi$$

The vector $V^\pi \in \mathbb{R}^S$ satisfies

$$\begin{aligned} (I - \gamma P^\pi)\, V^\pi &= r^\pi \\ V^\pi &= (I - \gamma P^\pi)^{-1}\, r^\pi \end{aligned}$$

provided that the matrix $I - \gamma P^\pi$ is invertible.

## Proposition

The eigenvalues of the *stochastic*[a] matrix $P^\pi$ all belong to $[0, 1]$. As a consequence, $-\gamma \notin \mathrm{sp}(P^\pi)$ thus $I - \gamma P^\pi$ is invertible.

*a.* the entries in its rows sum to 1

➜ $V^\pi$ can be computed by inverting a $S \times S$ matrix!

# An alternative :
# Exploiting the Bellman operator

## Definition

The **Bellman operator** associated to a policy $\pi$ is defined by

$$\begin{array}{rcl} T^\pi : \mathbb{R}^S & \longrightarrow & \mathbb{R}^S \\ V & \mapsto & T^\pi(V) \end{array}$$

where

$$T^\pi(V)(s) = r(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,\pi(s)) V(s')$$

The Bellman equation for policy $\pi$ rewrites

$$\boldsymbol{V^\pi = T^\pi V^\pi}$$

➜ the vector $V^\pi$ is a fixed point of the Bellman operator $T^\pi$

# Intermezzo : Fixed Point Theorem

## Definition

Let $(\mathcal{X}, |\cdot|)$ be a normed vector space.
A mapping $f : \mathcal{X} \rightarrow \mathcal{X}$ is called $L$-Lipschitz is

$$\forall (x,y) \in \mathcal{X}^2, |f(x) - f(y)| \leq L|x - y|.$$

If $L < 1$, $f$ is called a contraction.

## Banach's fixed point theorem

Let $(\mathcal{X}, |\cdot|)$ be a *complete* normed vector space and $f : \mathcal{X} \rightarrow \mathcal{X}$ be a contraction. Then

- there exists a unique fixed point $x^\star$ satisfying $f(x^\star) = x^\star$
- for every $x_0 \in \mathcal{X}$, the sequence defined by $x_{n+1} = f(x_n)$ for all $n \geq 0$ satisfies

$$\lim_{n \rightarrow \infty} x_n = x_\star$$

# Exploiting the Bellman Operator

## Proposition

The operator

$$\begin{array}{rcl} T^\pi : \left(\mathbb{R}^S, ||\cdot||_\infty\right) & \longrightarrow & \left(\mathbb{R}^S, ||\cdot||_\infty\right) \\ V & \mapsto & T^\pi(V)(s) = r(s,\pi(s)) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,\pi(s))V(s') \end{array}$$

is a $\gamma$-contraction.

**Proof.**

$$\begin{array}{rcl} ||T^\pi(V) - T^\pi(V')||_\infty & = & \sup_{s\in\mathcal{S}} |T^\pi(V)(s) - T^\pi(V')(s)| \\ & = & \sup_{s\in\mathcal{S}} |\gamma \sum_{s'\in\mathcal{S}} p(s'|s,\pi(s))(V(s') - V'(s'))| \\ & \leq & \gamma \sum_{s'\in\mathcal{S}} p(s'|s,\pi(s))||V - V'||_\infty \\ & = & \gamma||V - V'||_\infty . \end{array}$$

# Exploiting the Bellman Operator

**Proposition**

The operator

$$\begin{array}{rcl} T^\pi : (\mathbb{R}^S, ||\cdot||_\infty) & \longrightarrow & (\mathbb{R}^S, ||\cdot||_\infty) \\ V & \mapsto & T^\pi(V)(s) = r(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,\pi(s)) V(s') \end{array}$$

is a $\gamma$-contraction.

**Consequence** :

- $V^\pi$ is the unique fixed point of $T^\pi$
- $V^\pi$ can be approximated by an iterative scheme :

$$V^\pi = \lim_{k\to\infty} V_k$$

where

$$\begin{cases} V_0 & \in \mathbb{R} \\ V_{k+1} & = T^\pi(V_k) \text{ for all } k \geq 0. \end{cases}$$

# Summary : Policy Evaluation

Two methods for computing $V^\pi(s)$ for all $s$ :

- solving linear equations (matrix inversion)
- iterating the Bellman operator $T^\pi$

**Other possibility :** Monte-Carlo simulation

- ➜ provides only an approximation
- ➜ ... but doesn't require the knowledge of $r(s,a)$ and $p(\cdot|s,a)$...

# Back to Retail Store Management

- Constant policy : $\pi(s) = \max(M - s, c)$
- Threshold policy : $\pi(s) = \mathbb{1}_{(s \leq m_1)}(m_2 - s)$

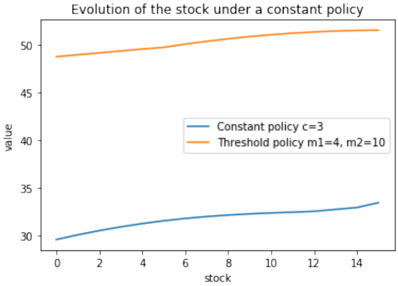



FIGURE – $V^\pi$ for two different policies, $\gamma = 0.97$

# Outline

# Bellman equations for the optimal values

## Proposition

$V^\star(s) = \max_\pi \ V^\pi(s)$ satisfy the **Bellman equations** :

$$V^\star(s) \quad = \quad \max_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^\star(s') \right]$$

Moreover, an optimal policy is given by $\pi^\star = (\pi^\star, \pi^\star, \dots)$ where

$$\pi^\star(s) \in \operatorname*{argmax}_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^\star(s') \right].$$

➜ $\pi^\star$ is the greedy policy with respect to $V^\star$ :

## Definition

The **greedy policy** with respect to a value $V$, $\pi = \text{greedy}(V)$ is

$$\pi(x) = \operatorname*{argmax}_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V(s') \right]$$

# Bellman equations for the optimal values

**Proposition**

$V^\star(s) = \max_\pi \ V^\pi(s)$ satisfy the **Bellman equations** :

$$V^\star(s) \quad = \quad \max_{a\in\mathcal{A}} \left[ r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) V^\star(s') \right]$$

Moreover, an optimal policy is given by $\pi^\star = (\pi^\star, \pi^\star, \dots)$ where

$$\pi^\star(s) \in \operatorname*{argmax}_{a\in\mathcal{A}} \left[ r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a) V^\star(s') \right].$$

→ $\pi^\star$ is the greedy policy with respect to $V^\star$ :

**Intuition :** greedy($V$) is the policy that "improves" a policy with value $V$ by taking the best possible first action and then following the policy

# Solving the Bellman equations

**Proposition**

The optimal value function $V^\star$ satisfies

$$V^\star(s) = \max_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^\star(s') \right]$$

- a system of $S$ non-linear equations for computing $(V^\star(s))_{s \in \mathcal{S}}$.

➜ no hope for a simple "matrix inversion" technique...

# Bellman operator to the rescue

## Optimal Bellman operator

The optimal Bellman operator (or dynamic programming operator) is

$$\begin{array}{rcl} T^\star : \mathbb{R}^S & \longrightarrow & \mathbb{R}^S \\ V & \mapsto & T^\star(V) \end{array}$$

where

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

The optimal value function satisfies

$$\boldsymbol{V^\star = T^\star(V^\star)}$$

➜ $V^\star$ is a fixed point of the optimal Bellman operator $T^\star$.

# Optimal Bellman Operator

## Properties

The optimal Bellman operator is a $\gamma$-contraction :

$$\forall\ V, V' \in \mathbb{R}^S, \quad ||T^\star(V) - T^\star(V')||_\infty < \gamma \, ||V - V'||_\infty \, .$$

As a consequence :

- $T^\star$ admits a unique fixed point, $V^\star$
- for every $V_0$, the sequence $V_{n+1} = T^\star(V_n)$ converges to $V^\star$

# Optimal Bellman Operator

## Properties

The optimal Bellman operator is a $\gamma$-contraction :

$$\forall\ V, V' \in \mathbb{R}^S, \quad ||T^\star(V) - T^\star(V')||_\infty < \gamma\, ||V - V'||_\infty \,.$$

As a consequence :

- $T^\star$ admits a unique fixed point, $V^\star$
- for every $V_0$, the sequence $V_{n+1} = T^\star(V_n)$ converges to $V^\star$

**Proof.** Uses that for all functions $|\max f - \max g| \leq \max |f - g|$.

$$\begin{aligned}
\|T^\star(U) - T^\star(V)\|_\infty &= \max_{s\in\mathcal{S}} \left| \max_{a\in\mathcal{A}} \left\{ r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a)U(s') \right\} - \max_{a'\in\mathcal{A}} \left\{ r(s,a') - \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a')V(s') \right\} \right| \\
&\leq \max_{s\in\mathcal{S}} \max_{a\in\mathcal{A}} \left| r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a)U(s') - r(s,a) - \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a)V(s') \right| \\
&= \gamma \max_{s\in\mathcal{S}} \max_{a\in\mathcal{A}} \left| \left\{ \sum_{s'\in\mathcal{S}} p(s'|s,a)(U(s') - V(s')) \right\} \right| \\
&\leq \gamma \max_{s\in\mathcal{S}} \max_{a\in\mathcal{A}} \sum_{s'\in\mathcal{S}} |p(s'|s,a)| \|U - V\|_\infty \leq \gamma \|U - V\|_\infty
\end{aligned}$$

# Optimal Bellman Operator

**Properties**

The optimal Bellman operator is a $\gamma$-contraction :

$$\forall\; V, V' \in \mathbb{R}^S, \quad \|T^\star(V) - T^\star(V')\|_\infty < \gamma \|V - V'\|_\infty .$$

As a consequence :

- $T^\star$ admits a unique fixed point, $V^\star$
- for every $V_0$, the sequence $V_{n+1} = T^\star(V_n)$ converges to $V^\star$

➜ provides a method for approximating $V^\star$

# Outline

1. Solving a Known MDP : Finite Horizon
2. Solving a Known MDP : the Discounted Case
   - Policy Evaluation
   - Computing the Optimal Policy
3. Value Iteration, Policy Iteration

# Value Iteration

- **Idea :** Iterate the operator $T^\star$ until $V$ doesn't change much

**Algorithm 1:** Value Iteration

**Input :** $\epsilon =$ stopping parameter
$V_0 =$ any function (e.g. $V_0 \leftarrow 0_S$)

$V \leftarrow V_0$
**while** $\|V - T^\star(V)\| \geq \epsilon$ **do**
  $V \leftarrow T^\star(V)$
**end**
**Return:** $\pi = \text{greedy}(V)$

**Theorem**

Value iteration converges in at most $\log\left(\frac{||T^\star(V_0) - V_0||_\infty}{\epsilon}\right)/\log(1/\gamma)$ iterations and outputs a policy $\pi$ satisfying $||V^\pi - V^\star|| \leq \frac{\gamma\epsilon}{1-\gamma}$.

# Policy Iteration

- **Idea :** alternate between policy evaluation and **policy improvement**

## Greedy policy

Recall that $\pi' = \text{greedy}(V)$ is the policy defined as

$$\pi'(s) \in \underset{a\in\mathcal{A}}{\text{argmax}} \left[ r(s,a) + \gamma \sum_{s'\in\mathcal{S}} p(s'|s,a)V(s') \right]$$

## Policy improvement theorem

For any policy $\pi$, $\pi' = \text{greedy}(V^\pi)$ improves over $\pi$ : $V^{\pi'} \geq V^\pi$.

**Proof.** uses some monotonicity property : $U \geq V \Rightarrow T^\pi(U) \geq T^\pi(V)$

1. by definition of the greedy policy, $T^{\pi'}(V^\pi) = T^\star(V^\pi) \geq T^\pi(V^\pi) = V^\pi$
2. the monotonicity property yields (by induction) $(T^{\pi'})^n(V^\pi) \geq V^\pi$ for all $n \in \mathbb{N}$
3. using that $\lim_{n\to\infty}(T^{\pi'})^n(V^\pi) = V^{\pi'}$ concludes.

# Policy Iteration

- **Idea :** alternate between policy evaluation and policy improvement .

**Algorithm 2:** Policy Iteration

**Input** : $\pi_0 =$ any policy (e.g. chosen at random)

$\pi \leftarrow \pi_0$
$\pi' \leftarrow \mathrm{NULL}$
**while** $\pi \neq \pi'$ **do**
  $\pi' \leftarrow \pi$
  Evaluate policy $\pi$ : compute $V^\pi$
  Improve policy $\pi$ : $\pi \leftarrow \text{greedy}(V^\pi)$
**end**
**Return:** $\pi$

**Theorem**

Policy iteration terminates after a finite number of steps and outputs the optimal policy $\pi^\star$.

# Policy Iteration

**Why is that ?**

Policy iteration generates a sequence of policies $\pi_0, \pi_1, \dots$ such that

$$\pi_{k+1} = \text{greedy}(V^{\pi_k}).$$

By the policy improvement theorem,

$$V^{\pi_{k+1}} \geq V^{\pi_k}$$

and if $\pi_{k+1} \neq \pi_k$ there must exists $s$ such that $V^{\pi_{k+1}}(s) > V^{\pi_k}(s)$ (otherwise $V^{\pi_k} = V^{\pi_{k+1}} = T^{\pi_{k+1}}(V^{\pi_{k+1}}) = T^{\star}(V^{\pi_k})$ thus $\pi_k = \pi^{\star}$)

➜ as there is a finite number of possible values of $V^{\pi}$ (finite number of policies), the sequence must be stationary at some point.

# Implementation of VI and PI

Both algorithm can be implemented using Q-Values .

## Definitions

The ***Q*-value** of a stationary policy is the expected cumulative reward when performing action $a$ in state $s$ and then following policy $\pi$ :

$$Q^\pi(s,a) = \mathbb{E}^\pi\left[\sum_{t=1}^{\infty}\gamma^{t-1}r(s_t,a_t)\middle| s_1 = s, a_1 = a\right]$$

The **optimal Q-value** is $Q^\star(s,a) = \max_\pi Q^\pi(s,a) = Q^{\pi^\star}(s,a)$.

**Properties** :

- $V^\pi(s) = Q^\pi(s,\pi(s))$
- $V^\star(s) = Q^\star(s,\pi^\star(s))$

# Implementation of VI and PI

Both algorithm can be implemented using Q-Values .

## Definitions

The ***Q*-value** of a stationary policy is the expected cumulative reward when performing action $a$ in state $s$ and then following policy $\pi$ :

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{\pi}(s')$$

The **optimal Q-value** is $Q^{\star}(s,a) = \max_{\pi} Q^{\pi}(s,a) = Q^{\pi^{\star}}(s,a)$.

**Properties** :

- $V^{\pi}(s) = Q^{\pi}(s,\pi(s))$
- $V^{\star}(s) = Q^{\star}(s,\pi^{\star}(s))$

# Implementation of VI and PI

Q-Values are convenient for policy improvement.

### $Q$-value associated to a value $V$

To each value function $V$, we can associate the corresponding $Q$-value

$$Q(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a)V(s')$$

**Property** : $\pi' = \text{greedy}(V)$ can be rewritten $\pi'(s) = \mathrm{argmax}_a \, Q(s,a)$.

# Implementation of VI and PI

Q-Values are convenient for policy improvement.

## $Q$-value associated to a value $V$

To each value function $V$, we can associate the corresponding $Q$-value

$$Q(s, a) = r(s, a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, a)V(s')$$

**Property** : $\pi' = \text{greedy}(V)$ can be rewritten $\pi'(s) = \text{argmax}_a Q(s, a)$.

## Definition

The **greedy policy** with respect to a Q-value $Q$, $\pi = \text{greedy}(Q)$ is

$$\pi(s) = \underset{a}{\text{argmax}}\, Q(s, a)$$

# Implementation of VI and PI

Q-Values are convenient for policy improvement.

**$Q$-value associated to a value $V$**

To each value function $V$, we can associate the corresponding $Q$-value

$$Q(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V(s')$$

**Property** : $\pi' = \text{greedy}(V)$ can be rewritten $\pi' = \text{greedy}(Q)$.

**Definition**

The **greedy policy** with respect to a Q-value $Q$, $\pi = \text{greedy}(Q)$ is

$$\pi(s) = \operatorname*{argmax}_a Q(s,a)$$

# Implementation of VI and PI

Q-Values are convenient for policy improvement.

**$Q$-value associated to a value $V$**

To each value function $V$, we can associate the corresponding $Q$-value

$$Q(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a)V(s')$$

**Property** : $\pi' = \text{greedy}(V)$ can be rewritten $\pi' = \text{greedy}(Q)$.

**Definition**

The **greedy policy** with respect to a Q-value $Q$, $\pi = \text{greedy}(Q)$ is

$$\pi(s) = \operatorname*{argmax}_a Q(s,a)$$

**Remark** : $\pi^\star = \text{greedy}(Q^\star)$

# Value Iteration versus Policy Iteration

In these implementations, we propose to store $Q$-values.

## Value Iteration

Initialize $Q_0$.

At iteration $k$ :

$$V_k(s) = \max_a Q_{k-1}(s,a) \quad \text{(apply } T^\star\text{)}$$

$$Q_k(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_k(s)$$

Output : $\pi_K(s) = \underset{a\in\mathcal{A}}{\text{argmax}}\ Q_K(s,a)$

## Policy iteration

Initialize $\pi_0$

At iteration $k$ :

$$Q_{k-1}(s,a) = Q^{\pi_{k-1}}(s,a) \text{ (policy evaluation)}$$

$$\pi_k(s) = \underset{a\in\mathcal{A}}{\text{argmax}}\ Q_{k-1}(s,a)$$

Output : $\pi_K$

# Value Iteration versus Policy Iteration

In these implementations, we propose to store $Q$-values.

### Value Iteration

Initialize $Q_0$.

At iteration $k$ :

$$V_k(s) = \max_a Q_{k-1}(s,a) \quad \text{(apply } T^\star\text{)}$$

$$Q_k(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_k(s)$$

Output : $\pi_K(s) = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q_K(s,a)$

### Policy iteration

Initialize $\pi_0$

At iteration $k$ :

$$Q_{k-1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{\pi_{k-1}}(s')$$

$$\pi_k(s) = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q_{k-1}(s,a)$$

Output : $\pi_K$

# Value Iteration versus Policy Iteration

In these implementations, we propose to store $Q$-values.

**Value Iteration**

Initialize $Q_0$.

At iteration $k$ :

$$V_k(s) = \max_a Q_{k-1}(s,a) \quad \text{(apply } T^\star\text{)}$$

$$Q_k(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_k(s)$$

Output : $\pi_K(s) = \operatorname{argmax}_{a \in \mathcal{A}} Q_K(s,a)$

**Policy iteration**

Initialize $\pi_0$

At iteration $k$ :

$$Q_{k-1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{\pi_{k-1}}(s')$$

$$\pi_k(s) = \operatorname{argmax}_{a \in \mathcal{A}} Q_{k-1}(s,a)$$

Output : $\pi_K$

**Space Complexity** : $O(SA)$ in both cases

- VI : Storing Q Values + Values
- PI : Storing Q values + Policy

# Value Iteration versus Policy Iteration

In these implementations, we propose to store $Q$-values.

**Value Iteration**

Initialize $Q_0$.

At iteration $k$ :

$$V_k(s) = \max_a Q_{k-1}(s,a) \quad \text{(apply } T^\star\text{)}$$

$$Q_k(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_k(s)$$

Output : $\pi_K(s) = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q_K(s,a)$

**Policy iteration**

Initialize $\pi_0$

At iteration $k$ :

$$Q_{k-1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{\pi_{k-1}}(s')$$

$$\pi_k(s) = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q_{k-1}(s,a)$$

Output : $\pi_K$

**Per Iteration Time Complexity** : $O(S^2 A) + O(S^3)$

- VI : Compute Q values + compute $S$ max
- PI : Compute Q values + compute $S$ argmax + Policy Evaluation

# Value Iteration versus Policy Iteration

In these implementations, we propose to store $Q$-values.

| Value Iteration | Policy iteration |
| --- | --- |
| Initialize $Q_0$.<br>At iteration $k$ :<br>$V_k(s) = \max_a Q_{k-1}(s,a)$ (apply $T^\star$)<br>$Q_k(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'\|s,a) V_k(s)$<br>Output : $\pi_K(s) = \text{argmax}_{a \in \mathcal{A}} Q_K(s,a)$ | Initialize $\pi_0$<br>At iteration $k$ :<br>$Q_{k-1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'\|s,a) V^{\pi_{k-1}}(s')$<br>$\pi_k(s) = \text{argmax}_{a \in \mathcal{A}} Q_{k-1}(s,a)$<br>Output : $\pi_K$ |

**Number of Iterations ?** : PI often requires few iterations

- VI : wait for $V_{k+1} \simeq V_k$ (requires a termination criterion)
- PI : wait for $\pi_{k+1} = \pi_k$ (finite number of iterations)

# Value Iteration versus Policy Iteration

In these implementations, we propose to store $Q$-values.

### Value Iteration

Initialize $Q_0$.

At iteration $k$ :

$$V_k(s) = \max_a Q_{k-1}(s,a) \quad \text{(apply } T^\star\text{)}$$

$$Q_k(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_k(s)$$

Output : $\pi_K(s) = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q_K(s,a)$

### Policy iteration

Initialize $\pi_0$

At iteration $k$ :

$$Q_{k-1}(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V^{\pi_{k-1}}(s')$$

$$\pi_k(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\, Q_{k-1}(s,a)$$

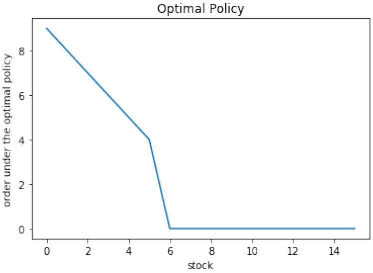
Output : $\pi_K$

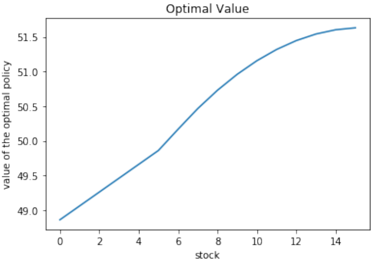
**Guarantees** :

- VI : a policy with value very close to $V^\star$ (often $\pi^\star$)
- PI : an optimal policy $\pi^\star$

# Back to Retail Store Management

Both VI and PI permit to find the optimal policy




$\pi^\star$ is a threshold policy with $m_1 = 5, m_2 = 9$
(with my choices of parameters)

# Summary

We learned how to find the optimal policy in an MDP with finite state and action spaces :

- using backwards induction for a finite horizon $H$
- using Policy and Value iteration for an infinite horizon with a discount $\gamma \in (0, 1)$

Those two types of techniques are often indifferently referred to as

**Dynamic Programming**.

We are now ready to propose **reinforcement learning algorithms**, that :

- operate without the knowledge of $r(s, a)$ and $p(\cdot|s, a)$
- or in very large state spaces in which standard Dynamic Programming is intractable