MPRI (2.8.2): Leçon 5
Hamilton-Jacobi-Bellman equation

Laurent Fribourg
24 février 2021
Markov Decision Processes and Dynamic Programming
Outline

Mathematical Tools

The Markov Decision Process

Bellman Equations for Discounted Infinite Horizon Problems

Bellman Equations for Undiscounted Infinite Horizon Problems

Dynamic Programming

Conclusions
Markov Chains

**Definition (Markov chain)**

Let the **state space** $X$ be a bounded compact subset of the Euclidean space, the discrete-time dynamic system $(x_t)_{t \in \mathbb{N}} \in X$ is a Markov chain if it satisfies the **Markov property**

$$
\mathbb{P}(x_{t+1} = x \mid x_t, x_{t-1}, \ldots, x_0) = \mathbb{P}(x_{t+1} = x \mid x_t),
$$

Given an initial state $x_0 \in X$, a Markov chain is defined by the **transition probability** $p$

$$
p(y|x) = \mathbb{P}(x_{t+1} = y \mid x_t = x).
$$
Markov Decision Process

**Definition (Markov decision process [1, 4, 3, 5, 2])**

A **Markov decision process** is defined as a tuple $M = (X, A, p, r)$ where

- $X$ is the state space,
- $A$ is the action space,
- $p(y|x, a)$ is the transition probability with
  \[ p(y|x, a) = \mathbb{P}(x_{t+1} = y|x_t = x, a_t = a), \]
- $r(x, a, y)$ is the reward of transition $(x, a, y)$. 
Policy

**Definition (Policy)**

A *decision rule* $\pi_t$ can be

- **Deterministic**: $\pi_t : X \to A$,
- **Stochastic**: $\pi_t : X \to \Delta(A)$,

A *policy* (strategy, plan) can be

- **Non-stationary**: $\pi = (\pi_0, \pi_1, \pi_2, \ldots)$,
- **Stationary (Markovian)**: $\pi = (\pi, \pi, \pi, \ldots)$.

**Remark**: MDP $M$ + stationary policy $\pi \Rightarrow$ *Markov chain* of state $X$ and transition probability $p(y|x) = p(y|x, \pi(x))$. 
Example: Racing

- A robot car wants to travel far, quickly
- Three states: Cool, Warm, Overheated
- Two actions: Slow, Fast
- Going faster gets double reward
Exercise: the Parking Problem

A driver wants to park his car as close as possible to the restaurant.

- The driver cannot see whether a place is available unless he is in front of it.
- There are $P$ places.
- At each place $i$ the driver can either move to the next place or park (if the place is available).
- The closer to the restaurant the parking, the higher the satisfaction.
- If the driver doesn’t park anywhere, then he/she leaves the restaurant and has to find another one.
5.5 Example: optimal parking

A driver is looking for a parking space on the way to his destination. Each parking space is free with probability \( p \) independently of whether other parking spaces are free or not. The driver cannot observe whether a parking space is free until he reaches it. If he parks \( s \) spaces from the destination, he incurs cost \( a, s = 0, 1, \ldots \). If he passes the destination without having parked the cost is \( D \). Show that an optimal policy is to park in the first free space that is no further than \( s^* \) from the destination, where \( s^* \) is the greatest integer \( s \) such that \( (Dp + 1)q^s \geq 1 \).

Solution. When the driver is \( s \) spaces from the destination it only matters whether the space is available \( (x = 1) \) or full \( (x = 0) \). The optimality equation gives

\[
F_s(0) = qF_{s-1}(0) + pF_{s-1}(1),
\]

\[
F_s(1) = \min \left\{ \begin{array}{ll}
    s, & \text{(take available space)} \\
    qF_{s-1}(0) + pF_{s-1}(1), & \text{(ignore available space)}
\end{array} \right.
\]

where \( F_0(0) = D, F_0(1) = 0 \).

Now we solve the problem using the idea of a OSLA rule. It is better to stop now (at a distance \( s \) from the destination) than to go on and take the first available space if \( s \) is in the stopping set

\[ S = \{ s : s \leq k(s - 1) \} \]

where \( k(s - 1) \) is the expected cost if we take the first available space that is \( s - 1 \) or closer. Now

\[ k(s) = ps + qk(s - 1), \]

with \( k(0) = qD \). The general solution is of the form \( k(s) = -q/p + s + cq^s \). So after substituting and using the boundary condition at \( s = 0 \), we have

\[ k(s) = -\frac{q}{p} + s + \left(D + \frac{1}{p}\right)q^{s+1}, \quad s = 0, 1, \ldots. \]

So

\[ S = \{ s : (Dp + 1)q^s \geq 1 \}. \]

This set is closed (since \( s \) decreases) and so by Theorem 5.3 this stopping set describes the optimal policy.

We might let \( D \) be the expected distance that that the driver must walk if he takes the first available space at the destination or further down the road. In this case, \( D = 1 + qD \), so \( D = 1/p \) and \( s^* \) is the greatest integer such that \( 2q^s \geq 1 \).
5.4 Optimal stopping over a finite horizon

One way that the total-expected cost can be finite is if it is possible to enter a state from which no further costs are incurred. Suppose \( u \) has just two possible values: \( u = 0 \) (stop), and \( u = 1 \) (continue). Suppose there is a termination state, say \( 0 \), that is entered upon choosing the stopping action. Once this state is entered the system stays in that state and no further cost is incurred thereafter. We let \( c(x, 0) = k(x) \) (stopping cost) and \( c(x, 1) = c(x) \) (continuation cost). This defines a stopping problem.

Suppose that \( F_s(x) \) denotes the minimum total cost when we are constrained to stop within the next \( s \) steps. The dynamic programming equation is

\[
F_s(x) = \min\{k(x), c(x) + E[F_{s-1}(x_1) \mid x_0 = x, u_0 = 1]\},
\]

with \( F_0(x) = k(x) \), \( c(0) = 0 \).

Consider the set of states in which it is at least as good to stop now as to continue one more step and then stop:

\[
S = \{ x : k(x) \leq c(x) + E[k(x_1) \mid x_0 = x, u_0 = 1] \}.
\]

Clearly, it cannot be optimal to stop if \( x \not\in S \), since in that case it would be strictly better to continue one more step and then stop. If \( S \) is closed then the following theorem gives us the form of the optimal policies for all finite-horizons.

**Theorem 5.3.** Suppose \( S \) is closed (so that once the state enters \( S \) it remains in \( S \).) Then an optimal policy for all finite horizons is: stop if and only if \( x \in S \).

**Proof.** The proof is by induction. If the horizon is \( s = 1 \), then obviously it is optimal to stop only if \( x \in S \). Suppose the theorem is true for a horizon of \( s - 1 \). As above, if \( x \not\in S \) then it is better to continue for more one step and stop rather than stop in state \( x \). If \( x \in S \), then the fact that \( S \) is closed implies \( x_1 \in S \) and so \( F_{s-1}(x_1) = k(x_1) \). But then \((5.2)\) gives \( F_s(x) = k(x) \). So we should stop if \( s \in S \). \( \square \)

The optimal policy is known as a one-step look-ahead rule (OSLA rule).
Example 4 (The tetris game).

Model:

- **State space**: configuration of the wall and next piece and terminal state when the well reach the maximum height.

- **Action space**: position and orientation of the current space in the wall.

- **Dynamics**: new configuration of the well and new random piece.

- **Reward**: number of deleted rows.

- **Objective function**: $\mathbb{E}[\sum_{t=1}^{T} r_t]$ with $T$ the random time when a terminal state is reached. *(remark: it has been proved that the game eventually terminates with probability 1 for any playing strategy).*

**Problem**: Compute the optimal strategy.

Solution: unknown! The state space is huge! $|X| = 10^{61}$ for a problem with maximum height 20, width 10 and 7 different pieces.
Question

How do we evaluate a policy and compare two policies?

⇒ Value function!
Optimization over Time Horizon

- **Finite time horizon** $T$: deadline at time $T$, the agent focuses on the sum of the rewards up to $T$.
- **Infinite time horizon with discount**: the problem never terminates but rewards which are closer in time receive a higher importance.
- **Infinite time horizon with terminal state**: the problem never terminates but the agent will eventually reach a termination state.
- **Infinite time horizon with average reward**: the problem never terminates but the agent only focuses on the (expected) average of the rewards.
State Value Function

- **Finite time horizon** $T$: deadline at time $T$, the agent focuses on the sum of the rewards up to $T$.

$$V^\pi(t, x) = \mathbb{E} \left[ \sum_{s=t}^{T-1} r(x_s, \pi_s(x_s)) + R(x_T) \mid x_t = x; \pi \right],$$

where $R$ is a value function for the final state.
State Value Function

- **Infinite time horizon with discount**: the problem never terminates but rewards which are closer in time receive a higher importance.

\[
V^\pi(x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, \pi(x_t)) \mid x_0 = x, \pi\right],
\]

with discount factor \(0 \leq \gamma < 1\):

- **small** = short-term rewards, **big** = long-term rewards
- for any \(\gamma \in [0, 1)\) the series always converge (for bounded rewards)
State Value Function

- **Infinite time horizon with terminal state**: the problem never terminates but the agent will eventually reach a *termination state*.

\[
V^\pi(x) = \mathbb{E}\left[ \sum_{t=0}^{T} r(x_t, \pi(x_t)) | x_0 = x; \pi \right],
\]

where \( T \) is the first *(random)* time when the *termination state* is achieved.
State Value Function

- *Infinite time horizon with average reward*: the problem never terminates but the agent only focuses on the (expected) average of the rewards.

\[
V^\pi(x) = \lim_{T \to \infty} \mathbb{E}\left[ \frac{1}{T} \sum_{t=0}^{T-1} r(x_t, \pi(x_t)) \mid x_0 = x; \pi \right].
\]
State Value Function

*Technical note*: the expectations refer to all possible stochastic trajectories.
A non-stationary policy $\pi$ applied from state $x_0$ returns

$$(x_0, r_0, x_1, r_1, x_2, r_2, \ldots)$$

with $r_t = r(x_t, \pi_t(x_t))$ and $x_t \sim p(\cdot | x_{t-1}, a_t = \pi(x_t))$ are random realizations.
The value function (discounted infinite horizon) is

$$V^\pi(x) = \mathbb{E}_{(x_1, x_2, \ldots)} \left[ \sum_{t=0}^{\infty} \gamma^t r(x_t, \pi(x_t)) \mid x_0 = x; \pi \right],$$
Optimal Value Function

**Definition (Optimal policy and optimal value function)**

The solution to an MDP is an optimal policy $\pi^*$ satisfying

$$\pi^* \in \arg \max_{\pi \in \Pi} V^\pi$$

in all the states $x \in X$, where $\Pi$ is some policy set of interest. The corresponding value function is the optimal value function $V^* = V^{\pi^*}$.

*Remark:* $\pi^* \in \arg \max(\cdot)$ and not $\pi^* = \arg \max(\cdot)$ because an MDP may admit more than one optimal policy.
Example: the MVA student dilemma
- **Model**: all the transitions are Markov, states $x_5, x_6, x_7$ are terminal.
- **Setting**: infinite horizon with terminal states.
- **Objective**: find the policy that maximizes the expected sum of rewards before achieving a terminal state.
\[ V_7 = -1000 \]
\[ V_6 = 100 \]
\[ V_5 = -10 \]
\[ V_4 = -10 + 0.9V_6 + 0.1V_4 \approx 88.9 \]
\[ V_3 = -1 + 0.5V_4 + 0.5V_3 \approx 86.9 \]
\[ V_2 = 1 + 0.7V_3 + 0.3V_1 \]
\[ V_1 = \max \{0.5V_2 + 0.5V_1, 0.5V_3 + 0.5V_1\} \]
\[ V_1 = V_2 = 88.3 \]
State-Action Value Function

**Definition**

In discounted infinite horizon problems, for any policy $\pi$, the state-action value function (or Q-function) $Q^\pi : X \times A \mapsto \mathbb{R}$ is

$$Q^\pi(x, a) = \mathbb{E}\left[ \sum_{t \geq 0} \gamma^t r(x_t, a_t) | x_0 = x, a_0 = a, a_t = \pi(x_t), \forall t \geq 1 \right],$$

and the corresponding optimal Q-function is

$$Q^*(x, a) = \max_\pi Q^\pi(x, a).$$
State-Action Value Function

The relationships between the V-function and the Q-function are:

\[ Q^\pi(x, a) = r(x, a) + \gamma \sum_{y \in X} p(y|x, a) V^\pi(y) \]

\[ V^\pi(x) = Q^\pi(x, \pi(x)) \]

\[ Q^*(x, a) = r(x, a) + \gamma \sum_{y \in X} p(y|x, a) V^*(y) \]

\[ V^*(x) = Q^*(x, \pi^*(x)) = \max_{a \in A} Q^*(x, a). \]
Outline

Mathematical Tools

The Markov Decision Process

Bellman Equations for Discounted Infinite Horizon Problems

Bellman Equations for Uniscounted Infinite Horizon Problems

Dynamic Programming

Conclusions
Question

*Is there any more compact way to describe a value function?*

⇒ *Bellman equations!*
The Bellman Equation

Proposition

For any stationary policy $\pi = (\pi, \pi, \ldots)$, the state value function at a state $x \in X$ satisfies the Bellman equation:

$$V^\pi(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x)) V^\pi(y).$$
The Bellman Equation

Proof.
For any policy $\pi$,

$$V^\pi(x) = \mathbb{E}\left[ \sum_{t \geq 0} \gamma^t r(x_t, \pi(x_t)) \mid x_0 = x; \pi \right]$$

$$= r(x, \pi(x)) + \mathbb{E}\left[ \sum_{t \geq 1} \gamma^t r(x_t, \pi(x_t)) \mid x_0 = x; \pi \right]$$

$$= r(x, \pi(x))$$

$$+ \gamma \sum_y \mathbb{P}(x_1 = y \mid x_0 = x; \pi(x_0)) \mathbb{E}\left[ \sum_{t \geq 1} \gamma^{t-1} r(x_t, \pi(x_t)) \mid x_1 = y; \pi \right]$$

$$= r(x, \pi(x)) + \gamma \sum_y p(y \mid x, \pi(x)) V^\pi(y).$$
The Optimal Bellman Equation

**Bellman’s Principle of Optimality** [1]:

“An optimal policy has the property that, whatever the initial state and the initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.”
The Optimal Bellman Equation

**Proposition**

The optimal value function $V^*$ (i.e., $V^* = \max_\pi V^\pi$) is the solution to the *optimal Bellman equation*:

$$V^*(x) = \max_{a \in A} \left[ r(x, a) + \gamma \sum_y p(y|x, a) V^*(y) \right].$$

and the optimal policy is

$$\pi^*(x) = \arg \max_{a \in A} \left[ r(x, a) + \gamma \sum_y p(y|x, a) V^*(y) \right].$$
The Optimal Bellman Equation

Proof.
For any policy $\pi = (a, \pi')$ (possibly non-stationary),

$$V^*(x) \overset{(a)}{=} \max_\pi \mathbb{E}[\sum_{t \geq 0} \gamma^t r(x_t, \pi(x_t)) | x_0 = x; \pi]$$

$$\overset{(b)}{=} \max_{(a, \pi')} \left[ r(x, a) + \gamma \sum_y p(y|x, a) V^\pi'(y) \right]$$

$$\overset{(c)}{=} \max_a \left[ r(x, a) + \gamma \sum_y p(y|x, a) \max_{\pi'} V^\pi'(y) \right]$$

$$\overset{(d)}{=} \max_a \left[ r(x, a) + \gamma \sum_y p(y|x, a) V^*(y) \right].$$
The Bellman Operators

Notation. w.l.o.g. a discrete state space $|X| = N$ and $V^\pi \in \mathbb{R}^N$.

**Definition**

For any $W \in \mathbb{R}^N$, the Bellman operator $\mathcal{T}^\pi : \mathbb{R}^N \rightarrow \mathbb{R}^N$ is

$$\mathcal{T}^\pi W(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x))W(y),$$

and the optimal Bellman operator (or dynamic programming operator) is

$$\mathcal{T} W(x) = \max_{a \in A} [r(x, a) + \gamma \sum_y p(y|x, a)W(y)].$$
The Bellman Operators

**Proposition**

Properties of the Bellman operators

1. **Monotonicity**: for any \( W_1, W_2 \in \mathbb{R}^N \), if \( W_1 \leq W_2 \) component-wise, then

\[
\mathcal{T}^\pi W_1 \leq \mathcal{T}^\pi W_2,
\]

\[
\mathcal{T} W_1 \leq \mathcal{T} W_2.
\]

2. **Offset**: for any scalar \( c \in \mathbb{R} \),

\[
\mathcal{T}^\pi (W + cl_N) = \mathcal{T}^\pi W + \gamma c l_N,
\]

\[
\mathcal{T} (W + cl_N) = \mathcal{T} W + \gamma c l_N,
\]
The Bellman Operators

Proposition

3. *Contraction in $L_\infty$-norm:* for any $W_1, W_2 \in \mathbb{R}^N$

\[
\|T^{\pi} W_1 - T^{\pi} W_2\|_{\infty} \leq \gamma \|W_1 - W_2\|_{\infty},
\]

\[
\|T W_1 - T W_2\|_{\infty} \leq \gamma \|W_1 - W_2\|_{\infty}.
\]

4. *Fixed point:* For any policy $\pi$

$V^{\pi}$ is the *unique fixed point* of $T^{\pi}$,

$V^*$ is the *unique fixed point* of $T$.

Furthermore for any $W \in \mathbb{R}^N$ and any stationary policy $\pi$

\[
\lim_{k \to \infty} (T^{\pi})^k W = V^{\pi},
\]

\[
\lim_{k \to \infty} (T)^k W = V^*.
\]
The Bellman Equation

Proof.
The contraction property (3) holds since for any $x \in X$ we have

$$|\mathcal{T}W_1(x) - \mathcal{T}W_2(x)|$$

$$= \max_a \left[ r(x, a) + \gamma \sum_y p(y|x, a)W_1(y) \right] - \max_{a'} \left[ r(x, a') + \gamma \sum_y p(y|x, a')W_2(y) \right]$$

$$(a) \leq \max_a \left[ r(x, a) + \gamma \sum_y p(y|x, a)W_1(y) \right] - \left[ r(x, a) + \gamma \sum_y p(y|x, a)W_2(y) \right]$$

$$= \gamma \max_a \sum_y p(y|x, a)|W_1(y) - W_2(y)|$$

$$\leq \gamma \|W_1 - W_2\|_\infty \max_a \sum_y p(y|x, a) = \gamma \|W_1 - W_2\|_\infty,$$

where in $(a)$ we used $\max_a f(a) - \max_{a'} g(a') \leq \max_a (f(a) - g(a))$. ■
Question

*How do we compute the value functions / solve an MDP?*

⇒ *Value/Policy Iteration algorithms!"*
System of Equations

The Bellman equation

\[ V^\pi(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x)) V^\pi(y). \]

is a linear system of equations with \( N \) unknowns and \( N \) linear constraints.

The optimal Bellman equation

\[ V^*(x) = \max_{a \in A} [r(x, a) + \gamma \sum_y p(y|x, a) V^*(y)]. \]

is a (highly) non-linear system of equations with \( N \) unknowns and \( N \) non-linear constraints (i.e., the \( \max \) operator).
Value Iteration: the Idea

1. Let $V_0$ be any vector in $R^N$
2. At each iteration $k = 1, 2, \ldots, K$
   - Compute $V_{k+1} = TV_k$
3. Return the greedy policy

$$\pi_K(x) \in \operatorname{arg\ max}_{a \in A} \left[ r(x, a) + \gamma \sum_y p(y|x, a) V_K(y) \right].$$
Value Iteration: the Guarantees

- From the *fixed point* property of $\mathcal{T}$:

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

- From the *contraction* property of $\mathcal{T}$

  $$||V_{k+1} - V^*||_\infty = ||\mathcal{T}V_k - \mathcal{T}V^*||_\infty \leq \gamma ||V_k - V^*||_\infty \leq \gamma^{k+1} ||V_0 - V^*||_\infty \to 0$$

- *Convergence rate*. Let $\epsilon > 0$ and $||r||_\infty \leq r_{max}$, then after at most

  $$K = \frac{\log(r_{max}/\epsilon)}{\log(1/\gamma)}$$

  iterations $||V_K - V^*||_\infty \leq \epsilon$. 

Value Iteration: the Complexity

One application of the optimal Bellman operator takes $O(N^2|A|)$ operations.
Value Iteration: Extensions and Implementations

**Q-iteration.**

1. Let $Q_0$ be any Q-function
2. At each iteration $k = 1, 2, \ldots, K$
   - Compute $Q_{k+1} = \mathcal{T} Q_k$
3. Return the greedy policy
   $$\pi_K(x) \in \arg \max_{a \in A} Q(x,a)$$

**Asynchronous VI.**

1. Let $V_0$ be any vector in $\mathbb{R}^N$
2. At each iteration $k = 1, 2, \ldots, K$
   - Choose a state $x_k$
   - Compute $V_{k+1}(x_k) = \mathcal{T} V_k(x_k)$
3. Return the greedy policy
   $$\pi_K(x) \in \arg \max_{a \in A} \left[ r(x,a) + \gamma \sum_y p(y|x,a) V_k(y) \right].$$
Policy Iteration: the Idea

1. Let \( \pi_0 \) be any stationary policy
2. At each iteration \( k = 1, 2, \ldots, K \)
   - Policy evaluation given \( \pi_k \), compute \( V^{\pi_k} \).
   - Policy improvement: compute the greedy policy
     \[
     \pi_{k+1}(x) \in \arg\max_{a \in A} [r(x, a) + \gamma \sum_y p(y|x, a) V^{\pi_k}(y)].
     \]
3. Return the last policy \( \pi_K \)

Remark: usually \( K \) is the smallest \( k \) such that \( V^{\pi_k} = V^{\pi_{k+1}} \).
Policy Iteration: the Guarantees

**Proposition**

The policy iteration algorithm generates a sequences of policies with non-decreasing performance

\[ V^{\pi_{k+1}} \geq V^{\pi_k}, \]

and it converges to \( \pi^* \) in a finite number of iterations.
Policy Iteration: the Guarantees

Proof.
From the definition of the Bellman operators and the greedy policy $\pi_{k+1}$

$$V^{\pi_k} = T^{\pi_k} V^{\pi_k} \leq T V^{\pi_k} = T^{\pi_{k+1}} V^{\pi_k},$$

(1)

and from the monotonicity property of $T^{\pi_{k+1}}$, it follows that

$$V^{\pi_k} \leq T^{\pi_{k+1}} V^{\pi_k},$$

$$T^{\pi_{k+1}} V^{\pi_k} \leq (T^{\pi_{k+1}})^2 V^{\pi_k},$$

$$\ldots$$

$$(T^{\pi_{k+1}})^{n-1} V^{\pi_k} \leq (T^{\pi_{k+1}})^n V^{\pi_k},$$

$$\ldots$$

Joining all the inequalities in the chain we obtain

$$V^{\pi_k} \leq \lim_{n \to \infty} (T^{\pi_{k+1}})^n V^{\pi_k} = V^{\pi_{k+1}}.$$

Then $(V^{\pi_k})_k$ is a non-decreasing sequence.
Policy Iteration: the Guarantees

Proof (cont’d).
Since a finite MDP admits a finite number of policies, then the termination condition is eventually met for a specific $k$. Thus eq. 1 holds with an equality and we obtain

$$V^{\pi_k} = T V^{\pi_k}$$

and $V^{\pi_k} = V^*$ which implies that $\pi_k$ is an optimal policy. ■
Policy Iteration

*Notation.* For any policy $\pi$ the reward vector is $r^\pi(x) = r(x, \pi(x))$ and the transition matrix is $[P^\pi]_{x,y} = p(y|x, \pi(x))$. 
Policy Iteration: the Policy Evaluation Step

- **Direct computation.** For any policy $\pi$ compute
  \[ V^{\pi} = (I - \gamma P^{\pi})^{-1} r^{\pi}. \]

  **Complexity:** $O(N^3)$ (improvable to $O(N^{2.807})$).
  **Exercise:** prove the previous equality.

- **Iterative policy evaluation.** For any policy $\pi$
  \[ \lim_{n \to \infty} T^{\pi} V_0 = V^{\pi}. \]

  **Complexity:** An $\epsilon$-approximation of $V^{\pi}$ requires $O(N^2 \log \frac{1}{\epsilon} \log \frac{1}{1/\gamma})$ steps.

- **Monte-Carlo simulation.** In each state $x$, simulate $n$ trajectories $((x_t^i)_{t \geq 0})_{1 \leq i \leq n}$ following policy $\pi$ and compute
  \[ \hat{V}^{\pi}(x) \sim \frac{1}{n} \sum_{i=1}^{n} \sum_{t \geq 0} \gamma^t r(x_t^i, \pi(x_t^i)). \]

  **Complexity:** In each state, the approximation error is $O(1/\sqrt{n})$. 

48
Policy Iteration: the Policy Improvement Step

- If the policy is evaluated with $V$, then the policy improvement has complexity $O(N|A|)$ (computation of an expectation).
- If the policy is evaluated with $Q$, then the policy improvement has complexity $O(|A|)$ corresponding to

$$\pi_{k+1}(x) \in \arg\max_{a \in A} Q(x, a),$$
Comparison between Value and Policy Iteration

Value Iteration
- Pros: each iteration is very \textit{computationally efficient}.
- Cons: convergence is only \textit{asymptotic}.

Policy Iteration
- Pros: converge in a \textit{finite} number of iterations (often small in practice).
- Cons: each iteration requires a full \textit{policy evaluation} and it might be expensive.
Things to Remember

- The Markov Decision Process framework
- The discounted infinite horizon setting
- State and state-action value function
- Bellman equations and Bellman operators
- The value and policy iteration algorithms
Bibliography I

R. E. Bellman.
*Dynamic Programming.*

D.P. Bertsekas and J. Tsitsiklis.
*Neuro-Dynamic Programming.*
Athena Scientific, Belmont, MA, 1996.

W. Fleming and R. Rishel.
*Deterministic and stochastic optimal control.*

R. A. Howard.
*Dynamic Programming and Markov Processes.*

M.L. Puterman.
Outline

1. Introduction

2. Dynamic Programming for 1-Player

3. Dynamic Programming for 2-Players
Differential Games
Outline

1 Introduction

2 Dynamic Programming for 1-Player

3 Dynamic Programming for 2-Players
Zermelo problem
Example 2: Zermelo problem with obstacles

\[ x' = V_{\text{boat}} \cos(\theta) + V_{\text{current}} - ay^2 \]
\[ y' = V_{\text{boat}} \sin(\theta) \]
Zermelo problem with obstacles: feedback control law
Exemple: Ariane V

Objectif
Minimize the ergol consumption to steer the (given) payload $M_{CU}$ to the GTO (or GEO).

Collaboration with Cnes (projet OPALE 2007-2010)
The physical model involves 7 state variables, the position $\overrightarrow{OG}$ of the rocket in the 3D space, its velocity $\overrightarrow{V}$ and its mass $m$.

The forces acting on the rocket are: Gravity $\overrightarrow{P}$, Drag $\overrightarrow{F_D}$, Thrust $\overrightarrow{F_T}$, and Coriolis $\overrightarrow{\Omega}$.

Newton Law:

$$m \frac{d\overrightarrow{V}}{dt} = \overrightarrow{P} + \overrightarrow{F_D} + \overrightarrow{F_T} - 2m\overrightarrow{\Omega} \wedge \overrightarrow{V} - m\overrightarrow{\Omega} \wedge (\overrightarrow{\Omega} \wedge \overrightarrow{OG}),$$
The related equation

State variables:
\( r \) = altitude
\( v \) = modulus of the velocity
\( \gamma \) = angle between the direction earth-rocket and the direction of the rocket’s velocity.
\( L \) = latitude
\( \ell \) = longitude
\( \chi \) = azimuth
\( m \) = masse of the engine

Control:
\( \alpha \) = angle between the thrust direction and the direction of the rocket’s velocity.
\[
\begin{align*}
\dot{r} &= v \cos \gamma \\
\dot{v} &= -g(r) \cos \gamma - \frac{F_D(r, v)}{m} - \frac{F_T(r, v, a)}{m} \cos \alpha \\
&\quad - \Omega^2 r \cos \ell (\cos \gamma \cos \ell - \sin \gamma \sin \ell \sin \chi) \\
\dot{\gamma} &= \sin \gamma \left( \frac{g(r)}{v} - \frac{v}{r} \right) - \frac{F_T(r, v, a)}{vm} \sin \alpha \\
&\quad - 2\Omega \cos \ell \cos \chi - \Omega^2 \frac{r}{v} \cos \ell (\sin \gamma \cos \ell - \cos \gamma \sin \ell \sin \chi) \\
\dot{L} &= \frac{v \sin \gamma \cos \chi}{r \cos \ell} \\
\dot{\ell} &= \frac{v}{r} \sin \gamma \sin \chi \\
\dot{\chi} &= -\frac{v}{r} \sin \gamma \tan \ell \cos \chi - 2\Omega (\sin \ell - \cotan \gamma \cos \ell \sin \chi) + \\
&\quad \Omega^2 \frac{r}{v} \sin \ell \cos \ell \cos \chi \frac{1}{\sin \gamma}
\end{align*}
\]
The plane of motion is the equatorial plane $\ell \equiv 0$, and $\chi \equiv 0$.

\[\dot{r} = v \cos \gamma\]
\[\dot{v} = -g(r) \cos \gamma - \frac{F_D(r, v)}{m} + \frac{F_T(r, v, a)}{m} \cos \alpha + \Omega^2 r \cos \gamma\]
\[\dot{\gamma} = \sin \gamma \left( \frac{g(r)}{v} - \frac{v}{r} \right) - \frac{F_T(r, v, a)}{vm} \sin \alpha - 2\Omega - \Omega^2 \frac{r}{v} \sin \gamma\]
\[L = \frac{v}{r} \sin \gamma\]
\[\dot{m} = -b(m(t))\]
The rocket's mass

The evolution of the mass can be summarized as follows:

<table>
<thead>
<tr>
<th>Phase 0 &amp; 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\dot{m}<em>1(t) = -\beta</em>{EAP}$</td>
<td>$\dot{m}_1(t) = 0$</td>
<td>$\dot{m}_1(t) = 0$</td>
</tr>
<tr>
<td>$\dot{m}<em>2(t) = -\beta</em>{E1}$</td>
<td>$\dot{m}<em>2(t) = -\beta</em>{E1}$</td>
<td>$\dot{m}_2(t) = 0$</td>
</tr>
<tr>
<td>$\dot{m}_3(t) = 0$</td>
<td>$\dot{m}_3(t) = 0$</td>
<td>$\dot{m}<em>3(t) = -\beta</em>{E2}$</td>
</tr>
</tbody>
</table>

where $\beta_{EAP}$, $\beta_{E1}$ and $\beta_{E2}$ are the mass flow rates for the boosters, the first and the second stage.

At the changes of phases, we have a (not negligible) discontinuity in the rocket's mass.
The control problem can be formulated as (for a fixed payload)

Minimize $t_f$

$(r, v, \gamma, m, \alpha)$ satisfy the state equation

$\alpha(t) \in [0, \pi/2]$ \ a.e. $t \in (0, t_f)$,

$(r(t_f), v(t_f), \gamma(t_f)) \in C,$

$Q(r(t), v(t) \alpha(t)) \leq C_s \ \text{for} \ t \in (0, t_f),$

$m(t_f) = M_p.$

where the target $C$ corresponds to the GTO orbit, and the function $Q$ is the dynamic pressure.
GTO target

Figure: Full trajectory using the HJB minimal time value function

Reference trajectory, final mass:
HJB trajectory, final mass (after reconstruction):

\[ m_T = 21.57 \text{ (t)} \]

\[ m_T = 22.50 \text{ (t)} \]
The model problem

Let us consider the nonlinear system dynamics

\[
\begin{align*}
\dot{y}(t) &= f(y(t), a(t), b(t)), \quad t > 0, \\
y(0) &= x
\end{align*}
\]  

(D)

where

- \( y(t) \in \mathbb{R}^N \) is the state
- \( a(\cdot) \in \mathcal{A} \) is the control of player 1 (player \( a \))

\[ \mathcal{A} = \text{admissible control functions of player 1} \]
\[ = \{ a : [0, +\infty[ \rightarrow \mathcal{A}, \text{ measurable} \} \]

(e.g. \( \mathcal{A} = \text{piecewise constant functions with values in } \mathcal{A} \)),


68
The model problem

\( b(\cdot) \in \mathcal{B} \) is the control of player 2 (player \( b \)),

\[
\mathcal{B} = \{ b : [0, +\infty[ \rightarrow \mathbb{R}^n, \text{ measurable} \},
\]

\( A, B \subset \mathbb{R}^M \) are given compact sets.
Assume \( f \) is continuous and

\[
|f(x, a, b) - f(y, a, b)| \leq L |x - y| \quad \forall x, y \in \mathbb{R}^N, a \in A, b \in B.
\]

Then, for every given control strategies \( a(\cdot) \in A, b(\cdot) \in \mathcal{B} \), there is a unique trajectory of (D), denoted by \( y_x(t; a, b) \) (Caratheodory).
Trajectories for a pursuit-evasion game

Figure: Optimal trajectories derived by feedback laws
Payoff

The payoff of the game is
\[
    t_x(a(\cdot), b(\cdot)) = \min \{ t : y_x(t; a, b) \in \mathcal{T} \} \leq +\infty,
\]
where \( \mathcal{T} \subseteq \mathbb{R}^N \) is a given closed target.

Goal of the game
Player \( a \) wants to minimize the payoff, he is called the pursuer, whereas Player \( b \) wants to maximize the payoff, he is called the evader.
Example: Pursuit-Evasion Games

We have two players, each one controlling its own dynamics

\[
\begin{cases}
\dot{y}_1 = f_1(y_1, a), & y_i \in \mathbb{R}^{N/2}, \ i = 1, 2 \\
\dot{y}_2 = f_2(y_2, b)
\end{cases}
\] (PEG)

The target is

\[
\mathcal{T}_\delta \equiv \{ |y_1 - y_2| \leq \delta \}, \quad \delta > 0, \text{ or } \mathcal{T}_0 \equiv \{ (y_1, y_2) : y_1 = y_2 \}.
\]

Then, \( t_x(a(\cdot), b(\cdot)) \) is the capture time corresponding to the strategies \( a(\cdot) \) and \( b(\cdot) \) of the first and second players.
Dynamic Programming for 1-Player

In this section we assume $B = \{ \bar{b} \}$, so the system can be rewritten as

\[
\begin{align*}
\dot{y} &= f(y, a), \quad t > 0, \\
y(0) &= x.
\end{align*}
\]

Define the value function

\[
T(x) \equiv \inf_{a(\cdot) \in A} t_x(a).
\]

$T(\cdot)$ is the minimum-time function, i.e. it is the best possible outcome of the game for player $a$, as a function of the initial position $x$ of the system.
Reachable set

**DEFINITION**
\[ \mathcal{R} \equiv \{ x \in \mathbb{R}^N : T(x) < +\infty \} , \text{ i.e. the set of starting points from which it is possible to reach the target.} \]

**WARNING**
The reachable set \( \mathcal{R} \) depends on the target and on the dynamics in a rather complicated way.

\( \mathcal{R} \) is *NOT* known in our problem, so we have to determine the couple \((T, \mathcal{R})\) (i.e. it’s a free boundary problem).
Outline

1. Introduction

2. Dynamic Programming for 1-Player

3. Dynamic Programming for 2-Players
Dynamic Programming for 1-Player

**LEMMA (Dynamic Programming Principle)**
For all $x \in \mathcal{R} \setminus \mathcal{T}$, $0 \leq t < T(x)$,

$$T(x) = \inf_{a(\cdot) \in A} \left\{ t + T(y_x(t; a)) \right\}.$$  \hspace{1cm} \text{(DPP)}

**Sketch of the Proof**
The inequality "$\leq$" follows from the intuitive fact that $\forall a(\cdot)$

$$T(x) \leq t + T(y_x(t; a)).$$
Sketch of the proof

The proof of the opposite inequality \( \geq \) is based on the fact that the equality holds if \( a(\cdot) \) is optimal for \( x \).

For any \( \varepsilon > 0 \) we can find a minimizing control \( a_\varepsilon \) such that

\[
T(x) + \varepsilon \geq t + T(y_x(t; a_\varepsilon))
\]

split the trajectory and pass to the limit for \( \varepsilon \to 0 \).
Sketch of the proof

To prove rigorously the above inequalities the following two properties of $\mathcal{A}$ are crucial:

1. $a(\cdot) \in \mathcal{A}$ implies that $\forall s \in \mathbb{R}$ the function $t \mapsto a(t + s)$ is still in $\mathcal{A}$;
2. $a_1, a_2 \in \mathcal{A}$ implies that for any given $s > 0$ the new control

$$a(t) \equiv \begin{cases} 
  a_1(t) & t \leq s, \\
  a_2(t) & t > s.
\end{cases}$$

belongs to $\mathcal{A}$ (concatenation property)
Concatenation is a crucial property

Note that the DP Principle works for

$$A = \{ \text{piecewise constants functions into } A \}$$

but not for

$$A = \{ \text{continuous functions into } A \}.$$ 

because joining together two continuous controls we are not guaranteed that the resulting control is continuous.
Getting the Bellman equation

Let us derive the Hamilton-Jacobi-Bellman equation from the DP Principle.
Rewrite (DPP) as

\[ T(x) - \inf_{a(\cdot) \in A} T(y_x(t; a)) = t \]

and divide by \( t > 0 \),

\[ \sup_{a(\cdot) \in A} \left\{ \frac{T(x) - T(y_x(t; a))}{t} \right\} = 1 \quad \text{for } t < T(x). \]

We want to pass to the limit as \( t \to 0^+ \).
Bellman equation

Assume $T$ is differentiable at $x$ and that the limit for $t \to 0^+$ commutes with the $\sup_{a(\cdot)}$.
Then, if $\dot{y}_x(0; a)$ exists,

$$\sup_{a(\cdot) \in \mathcal{A}} \{ -\nabla T(x) \cdot \dot{y}_x(0, a) \} = 1.$$ 

Then, for $\lim_{t \to 0^+} a(t) = a_0$, we get

$$\sup_{a_0 \in \mathcal{A}} \{ -\nabla T(x) \cdot f(x, a_0) \} = 1. \quad \text{(HJB)}$$

This is the Hamilton-Jacobi-Bellman partial differential equation, for our problem is a first order nonlinear PDE.
Bellman equation

Let us define the Hamiltonian,

$$H_1(x, p) := \max_{a \in A} \{ -p \cdot f(x, a) \} - 1,$$

we can rewrite (HJB) in short as

$$H_1(x, \nabla T(x)) = 0 \text{ in } \mathcal{R} \setminus \mathcal{T}.$$

A natural boundary condition on $\partial \mathcal{T}$ is

$$T(x) = 0, \quad \text{for } x \in \partial \mathcal{T}.$$
T verifies the HJB equation

**PROPOSITION**
If \( T(\cdot) \) is \( C^1 \) in a neighborhood of \( x \in \mathcal{R} \setminus \mathcal{T} \), then \( T(\cdot) \) satisfies for every \( x \)

\[
\max_{a \in A} \{ -\nabla T(x) \cdot f(x, a) \} = 1.
\]

**Sketch of the proof**
Let us prove first the inequality "\( \leq \)".
Fix \( \bar{a}(t) \equiv a_0 \ \forall t \), and set \( y_x(t) = y_x(t; \bar{a}) \). The (DPP) gives

\[
T(x) - T(y_x(t)) \leq t \quad \forall 0 \leq t < T(x).
\]
T verifies the HJB equation

We divide by $t > 0$, getting

$$\frac{T(x) - T(y_x(t))}{t} \leq 1, \quad \forall 0 < t < T(x).$$

Now let $t \to 0^+$ to get

$$-\nabla T(x) \cdot \dot{y}_x(0) = -\nabla T(x) \cdot f(x, a_0) \leq 1,$$

where $\dot{y}_x(0) = f(x, a_0)$ since $\bar{a}(t)$ tends to $a_0$. Then, we get

$$\max_{a \in A} \{-\nabla T(x) \cdot f(x, a)\} \leq 1.$$
T verifies the HJB equation

To prove the inequality \( \geq \), we fix \( \varepsilon > 0 \).
For all \( t \in ]0, T(x)[ \), by (DPP) there exists \( \alpha \in \mathcal{A} \) such that

\[
T(x) \geq t + T(y_x(t; \alpha)) - \varepsilon t.
\]

Then

\[
1 - \varepsilon \leq \frac{T(x) - T(y_x(t; \alpha))}{t} \leq -\frac{1}{t} \int_0^t \frac{\partial}{\partial s} T(y_x(s; \alpha)) \, ds
\]

\[
= -\frac{1}{t} \int_0^t \nabla T(y_x(s)) \cdot \dot{y}_x(s) \, ds = -\frac{1}{t} \int_0^t \nabla T(x) \cdot f(x, \alpha(s)) \, ds
\]

Passing to the limit for \( t \to 0^+ \) we get for every positive \( \varepsilon \)

\[
1 - \varepsilon \leq -\nabla T(x) \cdot f(x, a_0) \text{ for } a_0 \in \mathcal{A}
\]
T verifies the HJB equation

Since $\varepsilon$ is arbitrary, we finally obtain

$$\sup_{a \in A} \{-\nabla T(x) \cdot f(x, a)\} \geq 1.$$ 

and we conclude the proof.

In conclusion:
assuming that $T$ is a differentiable function, we have proved that $T$ satisfies pointwise the Bellman equation in the reachable set $\mathcal{R}$.

**WARNING:** the reachable $\mathcal{R}$ is not given in the problem.
Outline

1. Foreword
2. Discretization for 1-Player
3. Discretization for 2-Players
Time discretization

By applying the change of variable (Kružkov)

$$v(x) = 1 - e^{-T(x)}$$

we rewrite the equation in the new variable

$$v(x) + \sup_{a \in A} [-f(x, a) \cdot \nabla v] = 1 \quad \text{(HJ)}$$

$$v(x) = 0 \text{ on } \mathcal{T}$$
$$v(x) = 1 \text{ on } \partial \mathcal{R}$$

As we have seen, we can drop the second boundary condition.
Time discretization

Time step $h = \Delta t > 0$
Discrete times $t_j = jh$, $j \in \mathbb{N}$

Discrete dynamical system

\[
\begin{aligned}
x_{j+1} &= x_j + hf(x_j, a_j) \\
x_0 &= x
\end{aligned}
\]

We define the reachable set for the discrete dynamical system

\[
\mathcal{R}_h \equiv \{ x \in \mathbb{R}^N : \exists \{ a_j \} \text{ and } j \in \mathbb{N} \text{ such that } x_j \in \mathcal{T} \}
\]
Discrete Minimum Time Function

Let us define

$$n_h(\{a_j\}, x) = \begin{cases} +\infty & x \notin \mathcal{R}_h \\ \min\{\in \mathbb{N} : x_j \in \mathcal{T}\} & \forall x \in \mathcal{R}_h \end{cases}$$

$$N_h(x) = \min_{\{a_j\}} n_h(\{a_j\}, x)$$

The discrete analogue of the minimum time function is $N_h(x)h$. 

90
The discrete time Bellman equation

As in the continuous problem, we change the variable

$$v_h(x) = 1 - e^{-hN_h(x)} ,$$

again we have $0 \leq v_h \leq 1$.

By the Discrete Dynamic Programming Principle we get

$$v_h(x) = S(v_h)(x) \quad \text{on } \mathcal{R}_h \setminus \mathcal{T} . \quad (HJ_h)$$

where

$$S(v_h)(x) \equiv \min_{a \in A} \left[ e^{-h} v_h(x + hf(x, a)) \right] + 1 - e^{-h}$$

which is complemented by the boundary condition

$$v_h(x) = 0 \quad \text{on } \mathcal{T} \quad (BC)$$
Characterization of $v_h$

Note that $x \in \mathbb{R}^N \setminus \mathcal{R}_h$ implies that $x + hf(x, a) \in \mathbb{R}^N \setminus \mathcal{R}_h$

So we can extend $v_h$ to $\mathbb{R}^N$ setting

$$v_h(x) = 1 \quad \text{on } \mathbb{R}^N \setminus \mathcal{R}_h.$$

**THEOREM**

$v_h$ is the unique bounded solution of $(HJ_h) - (BC)$. 
Local Controllability

Let $\mathcal{T}$ be defined as

$$\mathcal{T} \equiv \{ x : g_i(x) \leq 0 \quad \forall i = 1, \ldots, M \}$$

where $g_i \in C^2(\mathbb{R}^N)$ and $|\nabla g_i(x)| > 0$ for any $x$ such that $g_i(x) = 0$.

**LOCAL CONTROLLABILITY**

$\forall x \in \mathcal{T}$ such that $g_i(x) = 0$ (i.e. belonging to $\partial \mathcal{T}$) $\exists a \in A$ for which

$$f(x, a) \cdot \nabla g_i(x) < 0.$$
Convergence

**THEOREM** (convergence, Bardi-F (1990))
Let the assumptions of the Lemma be satisfied and let $\mathcal{T}$ be compact with nonempty interior.
Then, for $h \to 0^+$

$$v_h \to v \text{ locally uniformly in } \mathbb{R}^N$$

$$h N_h \to T \text{ locally uniformly in } \mathcal{R}$$
Error estimate

Let us assume $Q$ is a compact subset of $\mathcal{R}$ where the following condition holds:

$$\exists C_0 > 0 : \forall x \in Q \text{ there is a time optimal control with total variation less than } C_0 \text{ bringing the system to } \mathcal{T}.$$  \hfill (BV)

**THEOREM** (Bardi-F. (1990))

Let the assumptions of theorem hold true and let $Q$ be a compact subset of $\mathcal{R}$ where (BV) holds.
Then $\exists \bar{h}, C > 0$:

$$|T(x) - h N_h(x)| \leq Ch, \quad \forall x \in Q, \quad \forall h \leq \bar{h}$$
First order scheme

**COROLLARY**
Under the same hypotheses there exists two positive constants $\bar{h}$ and $C$:

$$ |v(x) - v_h(x)| \leq Ch \quad \forall x \in Q, h \leq \bar{h} $$

This means that the rate of convergence of the approximation scheme is 1. Note that also high-order methods can be obtained more accurate schemes for the dynamics (e.g. Runge-Kutta).
Space discretization

We need a grid to obtain a fully discrete scheme. We can use a lattice or a triangulation of a rectangle $Q$ in $\mathbb{R}^2$, $Q \supset T$.

**NOTATIONS**

$x_i$ : nodes of the grid

$d$ : the number of nodes

$k := \text{max diameter of the cells (or triangles)}$

Sets of Indices:

$I_T := \{i \in \mathbb{N} : x_i \in T\}$ (target nodes)

$I_{out} := \{i \in \mathbb{N} : x_i + hf(x_i, a) \notin Q, \forall a\}$ (internal nodes)

$I_{in} := \{i \in \mathbb{N} : x_i + hf(x_i, a) \in Q\}$ (boundary nodes)
Fully discrete scheme

We want to solve the problem

\[ v(x_i) = \min_{a \in A} \left[ e^{-h} v(x_i + hf(x_i, a)) + 1 - e^{-h} \right], \quad \forall x_i \in I_{in}, \]

\[ v(x_i) = 0 \quad \forall x_i \in I_T \]

\[ v(x_i) = 1 \quad \forall x_i \in I_{out} \]

To get a finite dimensional problem we introduce the space of piecewise linear functions

\[ W^k := \{w : Q \to [0, 1] : w \in C(Q) \text{ and } \nabla w = \text{constant in } S_j \} \]
Fixed point problem

This means that we choose a piecewise linear $(P_1)$ reconstruction for $\nu(x_i + hf(x_i, a))$.

In fact, for any $i \in I_{\text{in}}$, $x_i + hf(x_i, a) \in Q$, there exists a vector of coefficients, $\lambda_{ij}(a)$:

$$0 \leq \lambda_{ij}(a) \leq 1, \quad \sum_{j=1}^{d} \lambda_{ij}(a) = 1$$

and

$$x_i + hf(x_i, a) = \sum_{j=1}^{L} \lambda_{ij}(a)x_j \quad \text{(convex combination)}$$

so we are writing $x_i + hf(x_i, a)$ in local coordinates.
Fixed point problem

Let us define the matrix $\Lambda := \{\lambda_{ij}\}$. We define the operator $S : \mathbb{R}^d \to \mathbb{R}^d$

$$S_i(U) := \begin{cases} 
\min_{a \in A}[e^{-h}\Lambda_i(a)U] + 1 - e^{-h}, & \forall i \in l_{in} \\
0, & \forall i \in l_T \\
1, & \forall i \in l_{out}
\end{cases}$$

Since $\beta := e^{-h} \in (0, 1)$, the operator $S = \{S_i\}$ takes its values in $[0, 1]^d$

$$S : [0, 1]^d \to [0, 1]^d$$

and has a unique fixed point.
THEOREM

\[ S : [0, 1]^d \rightarrow [0, 1]^d \text{ and} \]

\[ \| S(U) - S(V) \|_\infty \leq \beta \| U - V \|_\infty \]

Sketch of the proof

\( S \) is monotone, i.e.

\[ U \leq V \Rightarrow S(U) \leq S(V) \]

Then, for any \( U \in [0, 1]^d \)

\[ 1 - \beta = S_i(0) \leq S_i(U) \leq S_i(1) = 1, \quad \forall i \in I_{in} \]

where \( 1 \equiv (1, 1, \ldots, 1) \). This implies, \( S : [0, 1]^d \rightarrow [0, 1]^d \)
S is a contraction

For any \( i \in I_{in} \), we have

\[
S_i(U) - S_i(V) \leq \beta \Lambda_i(\hat{a})(U - V)
\]

and since \( \|\Lambda_i(a)\| \leq 1 \), \( \forall a \in A \), this implies

\[
\|S_i(U) - S(V)\|_{\infty} \leq \beta \|U - V\|_{\infty}.
\]
Monotone convergence

The scheme

\[ U^{n+1} = S(U^n) \]

converges for every initial condition \( U^0 \).

However, choosing \( U^0 \in [0, 1]^d \)

\[ U^0_i = \begin{cases} 0 & \forall i \in I_T \\ 1 & \text{elsewhere} \end{cases} \]

we have

\[ U^0 \in U^+ \equiv \{ U \in [0, 1]^L : U \geq S(U) \} \]
Monotone convergence

The sequence $U^n$ starting from that $U^0$ (which is a discrete supersolution) is monotone decreasing. This stems from the monotonicity of the discrete operator $S$, so

$$U^n \searrow U^*$$

by the fixed point argument. Note that monotonicity allows to accelerate convergence.
How the informations flow

The information flows from the target to the other nodes of the grid. In fact, on the nodes in $Q \setminus T$, $U_i^0 = 1$.

But starting from the first step of the algorithm the value of the internal nodes immediately decreases since, by the local controllability assumption, the Euler scheme drives them to the target where the value is set to 0.
Outline

1 Introduction

2 Dynamic Programming for 1-Player

3 Dynamic Programming for 2-Players
Dynamic Programming for 2-Players

What is the value function for the 2-players game?

**WARNING:**
It is *not*

\[ \inf_{a(\cdot) \in A} \sup_{b(\cdot) \in B} J(x, a, b) \]

because this would give to Player-\(a\) a big advantage choosing his control since he would know completely the future response of Player-\(b\) to any control function \(a(\cdot) \in A\).
Nonanticipating Strategies

A more reasonable information pattern can be modeled by means of the notion of nonanticipating strategies introduced by Varayia, Roxin and Elliott-Kalton

1-st Player

\[ \Delta \equiv \{ \alpha : B \to A | b(t) = \tilde{b}(t) \ \forall t \leq t' \ \text{implies} \]
\[ \alpha[b](t) = \alpha[\tilde{b}](t) \ \forall t \leq t' \}, \]

2-nd Player

\[ \Gamma \equiv \{ \beta : A \to B | a(t) = \tilde{a}(t) \ \forall t \leq t' \ \text{implies} \]
\[ \beta[a](t) = \beta[\tilde{a}](t) \ \forall t \leq t' \}. \]
Lower Value of a game

Now we can define the lower value of the game

\[ T(x) \equiv \inf_{\alpha \in \Delta} \sup_{b \in \mathcal{B}} t_x(\alpha[b], b), \]

or, after the change of variable,

\[ V(x) \equiv \inf_{\alpha \in \Delta} \sup_{b \in \mathcal{B}} J(x, \alpha[b], b) \]

where the payoff is

\[ J(x, a, b) = \int_0^{t_x(a,b)} e^{-t} \, dt \]
Value of a game

Similarly the upper value of the game is

\[ \tilde{T}(x) := \sup_{\beta \in \Gamma} \inf_{a \in A} t_x(a, \beta[a]), \]

or

\[ \tilde{V}(x) := \sup_{\beta \in \Gamma} \inf_{a \in A} J(x, a, \beta[a]). \]

**DEFINITION**

We say that the game has a value if the upper and lower values coincide, i.e. if \( T = \tilde{T} \) or \( V = \tilde{V} \).
Dynamic Programming Principle for 2 Players

**LEMMA**
For all $0 \leq t < T(x)$

$$T(x) = \inf_{\alpha \in \Delta} \sup_{b \in B} \{ t + T(y_x(t; \alpha[b], b)) \}, \quad \forall x \in R \setminus T,$$

and

$$V(x) = \inf_{\alpha \in \Delta} \sup_{b \in B} \left\{ \int_0^t e^{-s} ds + e^{-t} V(y_x(t; \alpha[b], b)) \right\}, \quad \forall x \in T^c \equiv R^N \setminus T.$$

The proof is similar to the 1-player case but more technical due to the essential use of non-anticipating strategies.
Isaacs equation

**Isaacs' Lower Hamiltonian**

\[ H(x, p) := \min_{b \in B} \max_{a \in A} \{-p \cdot f(x, a, b)\} - 1. \]

The upper values \( \tilde{T} \) and \( \tilde{V} \) satisfy a similar DP Principle.

**Isaacs' Upper Hamiltonian**

\[ \tilde{H}(x, p) := \max_{a \in A} \min_{b \in B} \{-p \cdot f(x, a, b)\} - 1. \]
Basic References

DETERMINISTIC CONTROL PROBLEMS AND GAMES
M. Bardi, I. Capuzzo Dolcetta, Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations, Birkhäuser, 1997

A. I. Subbotin, Generalized solutions of first-order PDEs, Birkhäuser, Boston, 1995

OTHER NUMERICAL METHODS
Viability kernel method via Viability Theory (Saint-Pierre, Quincampoix, Cardaliaguet...)
Stable bridges methods (Patsko, Kumkov, ...)

STOCHASTIC CONTROL PROBLEMS
Outline

1. Foreword

2. Discretization for 1-Player

3. Discretization for 2-Players
Minimum time problem with variable velocity

Eikonal equation with velocity $c(x, y) = |x + y|$.
The Tag-Chase Game

Dynamics
\[ f_P(y, a, b) = v_P a \quad f_E(y, a, b) = v_E b \]
\[ v_P = 2 \quad v_E = 1 \]

Admissible control sets
\[ A = B = B(0, 1) \]

Relative coordinates
\[ \ddot{x} = (x_E - x_P) \cos \theta - (y_E - y_P) \sin \theta \]
\[ \ddot{y} = (x_E - x_P) \sin \theta - (y_E - y_P) \cos \theta \]
Optimal trajectories
The Tag-Chase game with directional constraints

Dynamics

\[ f_P(y, a, b) = v_P a \quad f_E(y, a, b) = v_E b \]

\[ v_P = 2 \quad v_E = 1 \]

Admissible control sets

\[ A = B(0, 1) \]

\[ B = B(0, 1) \setminus S \]

where

\[ S = (\rho \cos \theta, \rho \sin \theta), \theta \in (\theta_1, \theta_2), \ |\rho| \leq 1 \]

So the pursuer has a forbidden cone of directions.
Value Function
Optimal Trajectories

Test 4: \( P = (-0.5, 0.8), E = (-0.5, 0.0) \)
The Homicidal chauffeur

Dynamics

\[
\begin{align*}
\dot{x}_P &= v_P \sin \theta \\
\dot{y}_P &= v_P \cos \theta \\
\dot{x}_E &= v_E \sin b \\
\dot{y}_E &= v_E \cos b \\
\dot{\theta} &= \frac{R}{v_p} a
\end{align*}
\]
Value Function

The value function is discontinuous on the barriers
Optimal Trajectories
Optimal Trajectories
Optimal Trajectories (Merz Thesis)
Optimal Trajectories (computed)
The Tag-chase game in a courtyard, $V_p > V_e$
The Tag-chase game in a courtyard, $V_p > V_e$
Basic References

APPROXIMATION OF DETERMINISTIC CONTROL PROBLEMS AND DIFFERENTIAL GAMES

The following references will contain many information and several links to the literature

M. Falcone, R. Ferretti, Semi-Lagrangian Approximation Schemes for Linear and Hamilton-Jacobi Equations, SIAM, 2014

M. Bardi, I. Capuzzo Dolcetta, Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations, Birkhäuser, 1997

Example: the Retail Store Management Problem

*Description.* At each month $t$, a store contains $x_t$ items of a specific goods and the demand for that goods is $D_t$. At the end of each month the manager of the store can order $a_t$ more items from his supplier. Furthermore we know that

- The *cost* of maintaining an inventory of $x$ is $h(x)$.
- The *cost* to order $a$ items is $C(a)$.
- The *income* for selling $q$ items is $f(q)$.
- If the demand $D$ is bigger than the available inventory $x$, customers that cannot be served leave.
- The *value of the remaining inventory* at the end of the year is $g(x)$.
- *Constraint:* the store has a maximum capacity $M$. 