Deterministic Dynamic Programming

V. Leclère (CERMICS, ENPC)

April 27, 2018

Contents

Contents

Controlled Dynamic System

A controlled dynamic system is defined by its *dynamic*

$$x_{t+1} = f_t(x_t, u_t)$$

and initial state x_0.

The variables

- x_t is the *state* of the system,
- u_t is the *control* applied to the system at time t.

Example :

- x_t is the position and speed of a satellite, u_t the acceleration due to the engine (at time t).
- x_t is the stock of products available, u_t the consumption at time t
- ...

Optimization Problem

We want to solve the following optimization problem

$$
\begin{aligned}
\min_{u_0,\dots,u_{T-1}} \quad & \sum_{t=0}^{T-1} L_t(x_t, u_t) + K(x_T) & & \text{(1a)}\\
s.t. \quad & x_0 \text{ given} & & \text{(1b)}\\
& x_{t+1} = f_t(x_t, u_t) & t = 0 \dots T-1 \quad & \text{(1c)}\\
& u_t \in U_t(x_t) & t = 0 \dots T-1 \quad & \text{(1d)}
\end{aligned}
$$

Where

- $L_t(x, u)$ is the cost incurred between t and $t+1$ for a starting state x with control u;
- $K(x)$ is the final cost incurred for the final state x;
- f_t is the dynamic of the dynamical system;
- $U_t(x)$ is the set of admissible controls at time t with starting state x.

Note : this is a Shortest Path Problem on an acircuitic directed

Open-Loop vs Closed Loop solution I

- Problem (1) can be solved directly by solving KKT conditions (Pontryagin approach), which will yields a sequence of optimal controls $(u_0^\sharp, \dots, u_{T-1}^\sharp)$. This is a so called open-loop solution as it is decided once (at time $t = 0$) and never questionned. This type of solution is easy to store and use but not robust to errors or imprecisions.
- Dynamic Programming approach yields an optimal policy $\{\pi_t^\sharp\}_{t\in[\![0,T-1]\!]}$. This is a so-called closed-loop solution as the control u_t is choosen at time t according to the actual state t. It is more complex to use and compute, but more robust to errors or imprecisions.
- In a deterministic and exact setting an open-loop solution is equivalent to a closed loop solution.

Open-Loop vs Closed Loop solution II

Let's take an example : you want to go from your room to the classroom in the least amount of time.

- An open-loop solution consist in studying the map, finding the optimal path, learning it, and then blindfold yourself before just applying it.
- A closed loop solution consist in studying the map, making a set of rules (called policies or strategies), keeping your eyes open to be able to apply this rules to what you see.

In particular we note that :

- It easier to remember just one path, than a whole set of rules.
- If anything happen (unprecision, some randomness...), an open-loop solution might be dangerous.

Policy

Definition

An admissible policy for problem (1) is a sequence of function π_t mapping the set $\mathbb{X}_t$ of possible state at time t into the set $\mathbb{U}_t$ of possible controls and such that

$$\forall t \in [\![0, T-1]\!], \quad \forall x \in \mathbb{X}_t, \qquad \pi_t(x) \in U_t(x).$$

Contents

Problem time-decomposition I

The problem can be written

$$
\begin{aligned}
\min_{u_0,\dots,u_{t-1}} \quad & \Big\{ \sum_{\tau=0}^{t-1} L_\tau(x_\tau, u_\tau) + \min_{u_t,\dots,u_{T-1}} \quad \sum_{\tau=t}^{T-1} L_\tau(x_\tau,u_\tau) + K(x_T) \Big\} \\
s.t. \quad & x_0 \text{ given} \\
& x_{\tau+1} = f_t(x_\tau, u_\tau) && \tau = 0 \dots t-1 \\
& x_{\tau+1} = f_\tau(x_\tau, u_\tau) && \tau = t \dots T-1 \\
& u_\tau \in U_\tau(x_\tau) && \tau = 0 \dots t-1 \\
& u_\tau \in U_\tau(x_\tau) && \tau = t \dots T-1
\end{aligned}
$$

Problem time-decomposition II

Which can be decomposed as

$$
\begin{aligned}
\min_{u_0,\dots,u_{t-1}} \quad & \Big\{ \sum_{\tau=0}^{t-1} L_\tau(x_\tau, u_\tau) + V_t(x_t) \Big\} \\
s.t. \quad & x_0 \text{ given} \\
& x_{\tau+1} = f_t(x_\tau, u_\tau) && \tau = 0 \dots t-1 \\
& u_\tau \in U_\tau(x_\tau) && \tau = 0 \dots t-1
\end{aligned}
$$

Where

$$
\begin{aligned}
V_t(x_t) = \min_{u_t,\dots,u_{T-1}} \quad & \Big\{ \sum_{\tau=t}^{T-1} L_\tau(x_\tau, u_\tau) + K(x_T) \Big\} \\
s.t. \quad & x_0 \text{ given} \\
& x_{\tau+1} = f_t(x_\tau, u_\tau) && \tau = t \dots T-1 \\
& u_\tau \in U_\tau(x_\tau) && \tau = t \dots T-1
\end{aligned}
$$

Problem time-decomposition III

Usually we write

$$\min_{u_0} \left\{ L_0(x_0, u_0) + \min_{u_1,\dots,u_{T-1}} \sum_{t=1}^{T-1} L_t\big(x_t, u_t\big) + K\big(x_T\big) \right\}$$

$$\begin{aligned} s.t. \quad & x_{t+1} = f_t(x_t, u_t) \\ & x_1 = f_0(x_0, u_0) \\ & u_t \in U_t(x_t) \end{aligned}$$

Or, more simply,

$$\min_{u_0} L_0(x_0, u_0) + V_1\big(f_0(x_0, u_0)\big)$$

where $V_1(x)$ is the value of the problem starting at time $t = 1$ with state $x_1 = x$.

Bellman value function

More generically, we denote $V_{t_0}(x)$ the optimal value of the problem starting at time t with state x:

$$
\begin{aligned}
V_{t_0}(x) = \min_{u_{t_0},\dots,u_{T-1}} \quad & \sum_{t=t_0}^{T-1} L_t\big(x_t, u_t\big) + K\big(x_T\big) && \text{(2a)}\\
s.t. \quad & x_{t+1} = f_t(x_t, u_t), \qquad x_{t_0} = x && \text{(2b)}\\
& u_t \in U_t(x_t) && \text{(2c)}
\end{aligned}
$$

Bellman Equation

Theorem

We have the Bellman equation (we assume existence of minimizers)

$$\begin{aligned} V_T(x) &= K(x) && \forall x \in \mathbb{X}_T \\ V_t(x) &= \min_{u_t \in U_t(x)} L_t(x, u_t) + V_{t+1} \circ \underbrace{f_t(x, u_t)}_{x_{t+1}} && \forall x \in \mathbb{X}_t. \end{aligned}$$

And the optimal policy is given by

$$\pi_t^\sharp(x) \in \arg\min_{u_t \in U_t(x)} \Big\{ L_t(x, u_t) + V_{t+1} \circ \underbrace{f_t(x, u_t)}_{x_{t+1}} \Big\} \qquad \forall x \in \mathbb{X}_t.$$

Dynamic Programming - linear quadratic case I

Consider the following optimization problem

$$\min_{u_0,\dots,u_{t-1}} \quad \sum_{t=0}^{T-1} u_t' R_t u_t + x_{t+1}' Q_{t+1} x_{t+1}$$

$$s.t. \qquad x_{t+1} = A_t x_t + B_t u_t \qquad\qquad \forall t = 0..T-1$$

where $R_t \succ 0$ and $Q_t \succeq 0$ are given matrices.
Solve this problem by Dynamic Programming.

Dynamic Programming - linear quadratic case II

We have

$$V_t(x) = x' K_t x, \qquad \forall t \in [\![0, T]\!]$$

where

$$\begin{cases} K_T & = 0 \\ \tilde{Q}_{t+1} & = K_{t+1} + Q_{t+1} \\ K_t & = A_t' \tilde{Q}_{t+1} A_t + A_t' \tilde{Q}_{t+1} B_t (R_t + B_t' \tilde{Q}_{t+1} B_t)^{-1} B_t' \tilde{Q}_{t+1} A_t \end{cases}$$

And

$$\pi_t^\sharp(x) = L_t x$$

with

$$L_t = -(R_t + B_t' \tilde{Q}_{t+1} B_t)^{-1} B_t' \tilde{Q}_{t+1} A_t.$$

Dynamic Programming Algorithm - discrete state case

Data: Problem parameters
Result: optimal control and value;
$V_T \equiv K$;
for $t : T-1 \to 0$ **do**
    for $x \in \mathbb{X}_t$ **do**
        $V_t(x) = \infty$;
        for $u \in U_t(x)$ **do**
            $v_u = L_t(x,u) + V_{t+1} \circ f_t\big(x,u\big)$;
            if $v_u < V_t(x)$ **then**
                $V_t(x) = v_u$;
                $\pi_t(x) = u$;

Algorithm 1: Dynamic Programming Algorithm (discrete case)

Number of flops: $O(T \times |\mathbb{X}_t| \times |\mathbb{U}_t|)$.

Curses of dimensionality

1. State. If we consider 3 independent states each taking 10 values, then $|\mathbb{X}_t| = 10^3 = 1000$. In practice DP is not applicable for states of dimension more than 5.
2. Decision. The decision are often vector decisions, that is a number of independent decision, hence leading to huge $|U_t(x)|$.

Contents

Dynamic Programming equation

$$V_{t_0}(x) = \min_{u_{t_0},\dots,u_{T-1}} \quad \sum_{t=t_0}^{T-1} L_t\big(x_t, u_t\big) + K\big(x_T\big)$$

$$s.t. \quad x_{t+1} = f_t(x_t, u_t), \qquad x_{t_0} = x$$

$$u_t \in U_t(x_t)$$

We have the Bellman equation

$$V_T(x) = K(x) \qquad \forall x \in \mathbb{X}_T$$

$$V_t(x) = \min_{u_t \in U_t(x)} L_t(x, u_t) + V_{t+1} \circ \underbrace{f_t(x, u_t)}_{x_{t+1}} \qquad \forall x \in \mathbb{X}_t.$$

And the optimal policy is given by

$$\pi_t^\sharp(x) \in \arg\min_{u_t \in U_t(x)} \Big\{ L_t(x, u_t) + V_{t+1} \circ \underbrace{f_t(x, u_t)}_{x_{t+1}} \Big\} \qquad \forall x \in \mathbb{X}_t.$$

Introducing Bellman's operator

We rewrite Bellman's equation as

$$\begin{cases} V_T &= K, \\ V_t &= \mathcal{T}_t(V_{t+1}) \end{cases}$$

where

$$\mathcal{T}_t(A) : x \mapsto \min_{u \in U_t(x)} \left\{ L_t(x,u) + A \circ f_t(x,u) \right\}$$

Indeed, an optimal policy is given by

$$\pi_t(x) \in \arg\min \mathcal{T}_t(V_{t+1})(x) \ .$$

Introducing Bellman's operator

We rewrite Bellman's equation as

$$\begin{cases} V_T &= K, \\ V_t &= \mathcal{T}_t(V_{t+1}) \end{cases}$$

where

$$\mathcal{T}_t(A) : x \mapsto \min_{u \in U_t(x)} \left\{ L_t(x,u) + A \circ f_t(x,u) \right\}$$

Indeed, an optimal policy is given by

$$\pi_t(x) \in \arg\min \mathcal{T}_t(V_{t+1})(x) \ .$$

Constructing policies from value function

- If you have an admissible policy π, you can compute an upper-bound of the optimization problem, by simply applying this policy, i.e.

$$\begin{cases} u_t & = \pi_t(x_t) \\ x_{t+1} & = f_t(x_t, u_t) \\ UB & = \sum_{t=0}^{T-1} L_t(x_t, u_t) + K(x_T) \end{cases}$$

- If you have an approximation $\tilde{V}_t$ of the actual value functions V_t, then you can derive an admissible (suboptimal) policy as $\pi_t(x) = \arg\min \mathcal{T}_t(\tilde{V}_t)(x)$.

Dynamic Programming Algorithm - Discretization

Data: Problem parameters, discretization point, interpolator
Result: optimal control and value;
$\tilde{V}_T \equiv K$;
Choose discretization points x_t^k of $\mathbb{X}_t$;
for $t : T-1 \to 0$ **do**
  for $k = 1..K$ **do**
    $v_t^k = \infty$;
    for $u \in U_t(x_t^k)$ **do**
      $v_u = L_t(x_t^k, u) + \tilde{V}_{t+1} \circ f_t\big(x_t^k, u\big)$;
      if $v_u < v_t^k$ **then**
        $v_t^k = v_u$;
    $\tilde{V}_t$ is defined as an interpolation of $\big(v_t^k, x_t^k\big)$;

Algorithm 2: Dynamic Programming Algorithm (Discretization - Interpolation)

Properties of the Bellman operator

- Monotonicity:

$$\forall x \in \mathbb{X}, \quad V(x) \leq \overline{V}(x) \quad \Rightarrow \quad \forall x \in \mathbb{X}, \quad \big(\mathcal{T}V\big)(x) \leq \big(\mathcal{T}\overline{V}\big)(x)$$

- Convexity: if L_t is jointly convex in (x,u), V is convex, and f_t is affine then

$$x \mapsto \big(\mathcal{T}V\big)(x) \quad \text{is convex}$$

- Linearity: for any piecewise linear function V, if L_t is also piecewise linear, and f_t affine, then

$$x \mapsto \big(\mathcal{T}V\big)(x) \quad \text{is piecewise linear}$$