EN530.603 Applied Optimal Control
Lecture 8: Dynamic Programming

October 14, 2020

Lecturer: Marin Kobilarov

Dynamic Programming (DP) is conerned with the computation of an *optimal policy*, i.e. an optimal control signal expressed as a function of the state and time

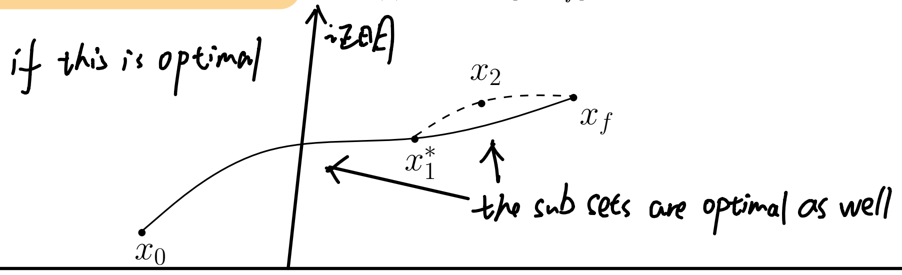
$$u = u(x, t).$$

This means that we seek solutions from *any* state $x(t)$ to a desired goal defined either as reaching a single state state x_f at time t_f, or reaching a surface defined by

$$\psi(x(t_f), t_f) = 0.$$ end state constraint

This *global* viewpoint is in contrast to the calculus of variations approach where we typically look at *local* variations of trajectories in the vicinity of a given initial path $x(\cdot)$ often starting from a given state $x(0) = x_0$.

The DP principle is based on the idea that once an optimal path $x^*(t)$ for $t \in [t_0, t_f]$ is computed then any segment of this path is optimal, i.e. $x^*(t)$ for $t \in [t_1, t_f]$ is the optimal path from $x^*(t_1)$ to $x^*(t_f)$.



For instance, consider an optimal path $x^*(t)$ between x_0 and x_f, i.e. a path with cost $J^*_{(x_0,x_f)}$ that is less than the cost of any other trajectory between these two points. According to the DP principle, for any given time $t_1 \in (t_0, t_f)$ the trajectory $x^*(t)$ from $x_1^* = x^*(t_1)$ to $x(t_f)$ is also otpimal. The principle can be easily proved by contradition. It it were not optimal then the optimal path must pass through a state x_2 which does not lie on $x^*(t)$, i.e.

$$J_{(x_1^*, x_2, x_f)} < J_{(x_1^*, x_f)}.$$

But then we have

$$J_{(x_0, x_1^*)} + J_{(x_1^*, x_2, x_f)} < J_{(x_0, x_1^*)} + J_{(x_1^*, x_f)} = J^*_{(x_0, x_f)},$$

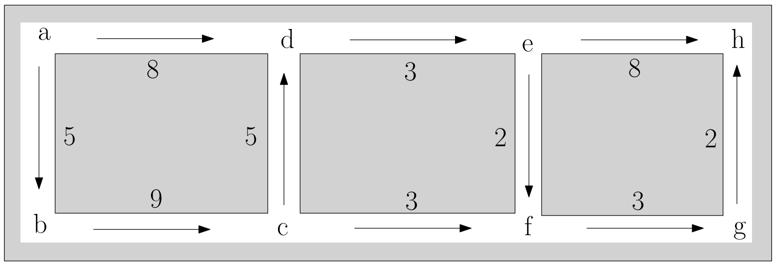
which is a contradition.

This property is key in dynamic programming, since it allows the computation of optimal trajectory segments which can then be pieced together in one globally optimal tarjectory, rather than say enumerating and comparing all possible trajectories ending at the goal.

Continuous & discrete decision 都可

DP applies to both continuous and discrete decision domains. The fully discrete setting occurs when the state x takes values from a finite set e.g. $x \in \{a, b, c, \dots\}$ and when the time variable t can only take a finite number of values, e.g. $t \in \{t_0, t_1, t_2, \dots\}$. As we will see there are efficient algorithms which can solve discrete problems and compute optimal policy from any state, as long as the size of these discrete sets is not too large.

For instance, consider the problem (Kirk 3.4) of a motorist navigating through a maze-like environment consisting of one-way roads and intersections marked by letters



The goal is to travel between a and h in an optimal way where the total cost is the sum of costs along each road given by the numbers above. One way to solve the problem is to enumerate all paths and pick the least-cost one. This is doable in this problem but does not scale if the there were much more roads and intersections. Another way is to solve the problem recursively by computing optimal subpaths starting backwards from the goal h.

Denote the cost along an edge ab by J_{ab} and the optimal cost of going between two nodes a and b by J^*_{ab}. Then we have

$$J_{eh} = 8, \quad J_{gh} = 2, \quad J_{fg} = 3$$

Clearly, we also have

$$J_{eh} = 8, \quad J^*_{gh} = 2, \quad J^*_{fh} = 5$$

but

$$J^*_{eh} = \min\{J_{eh}, J_{ef} + J^*_{fh}\} = 7$$

In other words the optimal cost at e can be computed by looking at the already compuetd optimal cost at f and the increment J_{ef} to get from e to f. Once we have J^*_{eh} the procedure is repeated recursively from all nodes connected to e and f, emanating backwards until the start a is reached and the cost J^*_{ah} is computed. The costs of the form J^*_{eh} in this case are called *optimal cost-to-go* since they denote the optimal cost to reach the goal.

discrete example

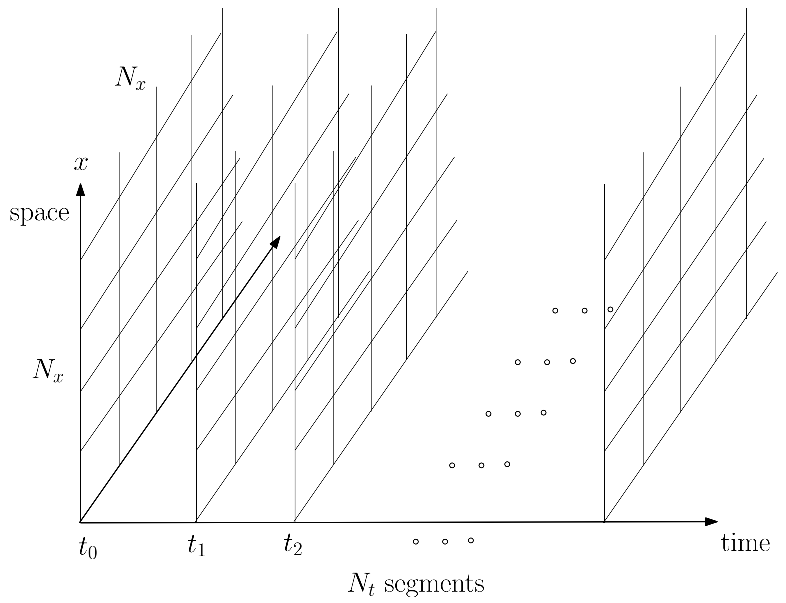
In the optimal control setting, we are tpically dealing with continuous spaces, e.g. $x \in \mathbb{R}^n$ and $t \in [t_0, t_f]$. One way to apply the discrete DP approach is to discretize space and time using a a grid of N_x discrete values along each state dimension and N_t discete values along time. The possible number of discrete states (or nodes) will then be

$$\#\text{of nodes } = N_t N_x^n,$$

Vanilla: 把t分成很多小段,再用 discrete 方法处理

or in other words their number is exponential in the state dimension. The case $n = 2$ is illustrated below where each cell could be regarded as a node in a graph similar to the maze navigation.

Costly



This illustrates one of the fundamental problems in control and DP known as the *curve of dimensionlity*. In practice,, this means that applying the discrete DP procedure is only feasible for a few dimensions, e.g. $n \neq 5$ and coarse discretizations, e.g. $N_x, N_t < 100$. But there are exceptions.

An additional complication when transforming a continuous problem into a discrete one is that when the system can only take on a finite number of states its continuous dynamics can be grossly violated.

缺点

This motivates the study of DP theory not only in discrete domains but also in

- continuous state space X, and discrete time $t \in \{t_0, t_1, t_2, ...\}$
- continuous state space X, and continuous time $t \in [t_0, t_f]$ (our standard case).

To proceed we will consider the restricted optimal control problems

$$\begin{aligned} J &= \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L\left(x(t), u(t), t\right) \mathrm{d}t, \\ &\text{subject to } \dot{x}(t) = f(x(t), u(t), t), \text{ with given } x(t_0), t_0, t_f \end{aligned} \tag{1}$$

1 Discrete-time DP

1.1 Bellman equations

Assume that the time line is discretized into the discrete set

$$\{t_0, t_1, \ldots, t_N\}.$$

When time is discrete, we work with discrete trajectories, i.e. sequences of states at these discrete times. A continuous trajectory $x(t)$ is approximated by a discrete trajectory $x_{0:N}$ given by

$$x_{0:N} = \{x_0, x_1, \ldots, x_N\},$$

where

$$x_i \approx x(t_i), \quad i = 0, \dots, N.$$

The dynamics then must be expressed in a discrete form according to

$$x_{i+1} = f_i(x_i, u_i),$$

where f_i is a numerical approximation of the dynamics. For instance, using the simplest Euler rule resulting from finite differences

$$\dot{x}(t_i) \approx \frac{x(t_i + \Delta t) - x(t_i)}{\Delta t}$$

fi表示 i时刻的state = Xi

f(ti,Xi,ui)是dynamic. 理解为state变化率

we have

$$f_i(x_i, u_i) = x_i + \Delta t_i f(t_i, x_i, u_i),$$

Δt 内 control不变　Δt要很小

where Δt_i is the time step defined by $\Delta t_i = t_{i+1} - t_i$.

The cost function along each segment $[t_i, t_{i+1}]$ is approximated by

$$\int_{t_i}^{t_{i+1}} L(x, u, t)dt \approx L_i(x_i, u_i),$$

where L_i is the discrete cost. For instance, in the simplest first order Euler approximation we have

$$L_i(x, u) = \Delta t_i L(x(t_i), u(t_i), t_i)$$

error不大

To summarize we have approximated our continuous problem (2) by the dsicrete problem

final state cost

$$J \approx J_0 = \phi(x_N, t_N) + \sum_{i=0}^{N-1} L_i\left(x_i, u_i\right), \tag{2}$$

$$\text{subject to} \;\; x_{i+1} = f_i(x_i, u_i), \;\; \text{with given } x(t_0), t_0, t_N$$

In order to establish a link to DP define the cost from discrete stage i to N by

$$J_i = \phi(x_N, t_N) + \sum_{k=i}^{N-1} L_k\left(x_k, u_k\right).$$

V function

Similarly to the maze navigation problem, let us define the optimal cost-to-go function (or *optimal value function*) at stage i as the optimal cost from state x at stage i to the end stage N:

$$V_i(x) \triangleq J_i^*(x) = \min_{u_{i:N-1}} J_i(x, u_{i:N-1})$$

Vi: i~N的 optimal cost

The optimal cost-to-go at x is computed by looking at all states $x' = f_i(x, u)$ that can be reached using control inputs u and selecting u which results in minimum cost-to-go from x' in addition to the cost to get to x'. More formally,

状态转移方程

$$V_i(x) = \min_u \left[L_i(x, u) + V_{i+1}\left(x'\right)\right],$$

where $x' = f_i(x, u)$. This can be equivalently written as *the Bellman equation*

$$V_i(x) = \min_u \left[L_i(x, u) + V_{i+1}\left(f_i(x, u)\right)\right],$$

with $V_N(x) = \phi(x, t_N)$.

Necessary + sufficient conditions for u^*

1.2 Discrete-time linear-quadratic case

# assumption 1

① trans is linear

Bellman equation has closed form solution when f_i is linear, i.e. when

$$x_{i+1} = A_i x_i + B_i u_i \longleftarrow \text{locally, linearized}$$

② cost is quadratic

$$\Downarrow$$

③ V in form $\frac{1}{2}x^T P x$

and when the cost is quadratic, i.e. \# assumption 2

$$\phi(x) = \frac{1}{2} x^T P_f x, \qquad L_i(x,u) = x^T Q_i x + u^T R_i u.$$

To see that assume that the value function is of the form \# assumption 3

$$V_i(x) = \frac{1}{2} x^T P_i x,$$

for $P_i > 0$ with boundary condition $P_N = P_f$. Bellman's principle requires that

$$\frac{1}{2} x^T P_i x = \min_u \left\{ \frac{1}{2} u^T R_i u + \frac{1}{2} x^T Q_i x + \frac{1}{2} \overline{(A_i x + B_i u)^T P_{i+1} (A_i x + B_i u)}^{V_{i+1}} \right\}$$

The minimization results in 对u求导 $= R_i u + B_i^T P_{i+1}(A_i x + B_i u)$.

$$\Downarrow$$

$$u^* = -(R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T P_{i+1} A_i x \equiv K_i x,$$

where

$$K_i = -(R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T P_{i+1} A_i$$

Substituting u^* back into the Bellman equation we obtain

$$x^T P_i x = x^T \left[K_i^T R_i K_i + Q_i + (A_i + B_i K_i)^T P_{i+1} (A_i + B_i K_i) \right] x.$$

Note, setting

$$P_i = K_i^T R_i K_i + Q_i + (A_i + B_i K_i)^T P_{i+1} (A_i + B_i K_i) \longrightarrow V_i(x) = \frac{1}{2} x^T P_i x,$$

with $P_N = P_f$ will satisfy the Bellman equation. This relationship can be cycled backwards starting from $i = N-1$ to $i = 0$ to obtain $P_{N-1}, P_{N-2}, \ldots, P_0$.

$P_N = P_f$

从 end 开始

The recurence can also be expressed without gains K_i according to

$$P_i = Q_i + A_i^T [P_{i+1} - P_{i+1} B_i (R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T P_{i+1}] A_i.$$

Note that we have expressed the optimal control at stage i according to

$$u_i^* = K_i x_i,$$

or in a linear feedback form, similarly to the continuous case. This means that the gains K_i can be computed only once in the beginning and then used from any state x_i. This solution is equivalent to the *discrete Linear-Quadratic-Regulator*.

Bellman's equation (discrete)
HJB equation (continous)

2 Continuous Dynamic Programming

目的都是为了得到u

We next consider the fully continuous case and will derive an analog to Bellman's equation. The continuous analog is called the Hamilton-Jacobi-Bellman equation and is a key result in optimal control.

Consider the *continuous value function* $V(x,t)$ computed over the time interval $[t, t_f]$ defined by

$$V(x(t),t) = \min_{u(t), t\in[t,t_f]} \left[\phi(x(t_f),t_f) + \int_t^{t_f} L(x(\tau),u(\tau),\tau)d\tau\right]$$

As noted earlier, Bellman's principle of optimality states that if a trajectory over $[t_0, t_f]$ is optimal then it is also optimal on any subinterval $[t, t+\Delta] \subset [t_0, t_f]$.

This can be expressed more formally through the recursive relationship

subject to dynamic

$$V(x(t),t) = \min_{u(t), t\in[t,t+\Delta]} \left[\int_t^{t+\Delta} L\left(x(\tau),u(\tau),\tau\right) \mathrm{d}\tau + \underline{V(x(t+\Delta),t+\Delta)}\right], \qquad (3)$$

where the optimization is over the continuous control signal $u(t)$ over the interval $[t, t+\Delta]$.

We can now state the key principle of dynamic programming in continuous space: optimal trajectories must satisfy the *Hamilton-Jacobi-Bellman equations* (HJB) given by:

HJB eqn

$$-\partial_t V(x,t) = \min_{u(t)} \{L(x,u,t) + \nabla_x V(x,t)^T \underbrace{f(x,u,t)}_{\text{dynamics}}\} \qquad (4)$$

即求满足(4)的V function

To prove HJB, assume that Δ is small expand $V(x(t+\Delta),t+\Delta)$ according to

$$\underline{V(x(t+\Delta),t+\Delta)} = V(x(t),t) + \left[\partial_t V(x(t),t) + \nabla_x V(x(t),t)^T\dot{x}\right]\Delta + o(\Delta)$$

Substituting the above in (3) and taking $\Delta \to 0$ we have

$$V(x(t),t) = \min_{u(t)} \left\{L\left(x(t),u(t),t\right)\Delta + V(x(t),t) + \left[\partial_t V(x(t),t) + \nabla_x V(x(t),t)^T\dot{x}\right]\Delta\right\},$$

which is equivalent to $\cancel{V(x(t),t)} = \min_{u(t)} \left\{L\left(x(t),u(t),t\right)\Delta + \cancel{V(x(t),t)} + \left[\partial_t V(x(t),t) + \nabla_x V(x(t),t)^T\dot{x}\right]\Delta\right\},$

$$0 = \min_{u(t)} \left\{L\left(x(t),u(t),t\right) + \underline{\partial_t V(x(t),t)} + \nabla_x V(x(t),t)^T f(x(t),u(t),t)\right\}\Delta.$$

☆ u的式子

与u(t)无关, 移出, 得到(3)

Since this must hold for any Δ then we obtain the HJB equation.

根据这个解出V

Example 1. A simple nonlinear system. Consider the scalar system

$$\dot{x} = -x^3 + u$$

with cost

$$J = \frac{1}{2}\int_t^\infty [q(x) + u^2]dt$$

First we find

L $\quad \nabla_x V^T f. \quad \partial_t V = 0$

$$u^* = \arg\min_u \left\{\frac{1}{2}[q(x)+u^2] + \nabla_x V(x,t)(-x^3+u)\right\}$$

The solution is

$$u^* = -\nabla_x V(x,t)$$

对u求导: $u + \nabla_x V$, for optimality, $u + \nabla_x V = 0$

要得到u 得先得到V

$$u^* = \arg\min_u \{L + \nabla_x V^T \cdot f\}$$

解出u (含V) 再求V

must hold for any x, hard to find such V

The HJB equation becomes

$$-\partial_t V(t,x) = \frac{1}{2}[q(x) - \nabla_x V(t,x)^2] - \nabla_x V(t,x)x^3$$

If we can solve this PDE globally then we're done. But this might be difficult

Example 2. Linear-quadratic case. Consider the scalar system

$$\dot{x} = x + u \qquad u^* = \arg\min_u \{ \frac{1}{2}u^2 + \nabla_x V \cdot \underbrace{f}_{=x+u} \}$$

with cost

$$J = \frac{1}{2}x(t_f)^2 + \int_0^{t_f} \frac{1}{2}u(t)^2 dt$$

We have

$$H(x,u,\nabla_x V,t) = \frac{1}{2}u^2 + \overset{\nabla_x V^T \cdot f}{\nabla_x V[x+u]}$$

The optimal control is computed by satisfying $\partial_u H = 0$, i.e.

$$u^*(t) = -\nabla_x V(x,t)$$

Furthermore, we have $\partial_u^2 H = 1 > 0$ so u^* is globally optimal!

Substituting the control, the HJB equation becomes

$$-\partial_t V(t,x) = -\frac{1}{2}\nabla_x V(t,x)^2 + \nabla_x V(t,x)x$$

Consider the possible value function

$$-\partial_t V(x,t) = \min\{L(x,u,t) + \nabla_x V(x,t)^T f(x,u,t)\}$$

假设V是这种形式
解出对应解

$$V(t,x) = \frac{1}{2}P(t)x^2$$

① trans is linear

② cost is quadratic

$\Downarrow$

③ V in form $\frac{1}{2}x^T P x$

The HJB equation becomes

$$-\frac{1}{2}\dot{P}x^2 = -\frac{1}{2}P^2x^2 + Px^2$$

which is equivalent to

$$\dot{P} + 2P - P^2 = 0$$

Using separation of variables and the final condition $P(t_f) = 1$, the solution is

at t_f, the cost will just be $\frac{1}{2}x(t_f)^2$

$$P(t) = \frac{2}{e^{2(t-tf)} + 1}$$

The optimal (feedback) control law becomes

$$u = -P(t)x.$$

2.1 General Linear-Quadratic Regulator (LQR)

Consider the linear system $\dot{x} = Ax + Bu$ with cost function

$$J = \frac{1}{2} x^T(t_f) P_f x(t_f) + \int_0^{t_f} \frac{1}{2} \left(x^T Q x + u^T R u \right) \mathrm{d}t$$

The Hamiltonian is

$\nabla_x V^T$看作 λ

$$H = \frac{1}{2} \left(x^T Q x + u^T R u \right) + \nabla_x V^T (Ax + Bu)$$

It is positive definite since $\nabla_u^2 H = R$. The optimal control is

$R>0$

$$u^* = -R^{-1} B^T \nabla_x V \longleftarrow u = -R^{-1}B^T\lambda \qquad \lambda(t) = P(t)x(t).$$

which results in the HJB $\quad -\partial_t V(x,t) = \min \left\{ L(x,u,t) + \nabla_x V(x,t)^T f(x,u,t) \right\}$

$$-\partial_t V = \frac{1}{2} x^T Q x + \frac{1}{2} \nabla_x V^T B R^{-1} B^T \nabla_x V + \nabla_x V^T (Ax - BR^{-1} B^T \nabla_x V).$$

Consider the value function

linear system ✓
quadratic cost ✓

assumption

$$V(x(t), t) = \frac{1}{2} x^T(t) P(t) x(t),$$

for a positive definite symmetric $P(t)$. The optimal control law becomes

$$u(t) = -R^{-1} B^T P(t) x(t).$$

We obtain

$$\begin{aligned} -\frac{1}{2} x \dot{P} x &= \frac{1}{2} x^T (Q - PBR^{-1}B^T P + 2PA) x \\ &= \frac{1}{2} x^T (Q - PBR^{-1}B^T P + PA + A^T P) x \end{aligned}$$

to make everything symmetric.

hard to get.

which will always hold true if

$$-\dot{P}(t) = A^T P(t) + P(t) A - P(t) B R^{-1} B^T P(t) + Q.$$

which is the *Ricatti* ODE for $P(t)$. Note that we replaced $2PA$ with $PA + A^T P$ above in order to ensure that $P(t)$ remains symmetric for all t.

terminal constraint.

matrix都应转成 symmetric matrix

The matrix P is known at the terminal point, i.e.

$$P(t_f) = P_f$$

Therefore, $P(t)$ is computed by backward integration of the Ricatti ODE.

Note that P is symmetric, so only $n(n+1)/2$ equations are integrated.

minimum principle

$\downarrow$

2.2 Link b/n MP and DP

instantaneous cost.

$\lambda^T f$ = cost to go   u会使 $\lambda^T f$ 尽可能大

- We have the relationship

$$\underbrace{L(x,u,t)}_{\text{instantaneous cost.}} + \overset{\lambda^T f}{\overset{\parallel}{\nabla_x V(x,t)^T f(x,u,t)}} = H(x,u,\nabla_x V(x,t),t),$$

  i.e. $\nabla_x V$ plays the role of the multiplier λ

- HJB equation can be written as

$$\partial_t V(x,t) = \max_u H(x,u,\nabla_x V(x,t),t)$$

- The optimal control becomes

$$u^* = \max_u H(x,u,\nabla_x V,t)$$

- If the corresponding optimal trajectory is x^* then

$$\partial_t V(x^*,t) = H(x^*,u^*,\nabla_x V(x^*,t),t)$$

- Think of λ as "guiding" the evolution along the gradient of V. Visually, if V can be thought of an expanding front emanating from the goal set, then the vector λ is orthogonal to that front and pointing outward, i.e. it gives the direction of fastest increase of V.

DP in practice.

- If we knew $V(x,t)$ then then DP gives a closed-loop control law

$$u^* = \max_u H(x,u,\nabla_x V,t)$$

- But computing $V(x,t)$ globally is extremely challenging
- Various approximate solutions are possible
- Simplifications (but not drastic) obtained in the time-independent case with infinite time horizon