Dynamic Programming and Optimal Control

Script

Prof. Raffaello D'Andrea

Lecture notes

Dieter Baldinger   Thomas Mantel   Daniel Rohrer

HS 2010

Contents

Contents **1**

1 Introduction **3**
1.1 Class Objective 3
1.2 Key Ingredients 3
1.3 Open Loop versus Closed Loop Control 4
1.4 Discrete State and Finite State Problem 5
1.5 The Basic Problem 7

2 Dynamic Programming Algorithm **9**
2.1 Principle of Optimality 9
2.2 The DPA 10
2.3 Chess Match Strategy Revisited 11
2.4 Converting non-standard problems 13
2.4.1 Time Lags 13
2.4.2 Correlated Disturbances 13
2.4.3 Forecasts 14
2.5 Deterministic, Finite State Systems 15
2.5.1 Convert DP to Shortest Path Problem 15
2.5.2 DP algorithm 16
2.5.3 Forward DP algorithm 16
2.6 Converting Shortest Path to DP 16
2.7 Viterbi Algorithm 17
2.8 Shortest Path Algorithms 18
2.8.1 Label Correcting Methods 19
2.8.2 $\mathbf{A}^*-$ Algorithm 21
2.9 Multi-Objective Problems 21
2.9.1 Extended Principle of Optimality 22
2.10 Infinite Horizon Problems 23
2.11 Stochastic, Shortest Path Problems 23
2.11.1 Main Result 24
2.11.2 Sketch of Proof 25
2.11.3 Prove B 27
2.12 Summary of previous lecture 27
2.13 How do we solve Bellman's Equation? 28

2.13.1 Method 1: Value iteration (VI) 28
2.13.2 Method 2: Policy Iteration (PI) 28
2.13.3 Third Method: Linear Programming 31
2.13.4 Analogies and Connections 32
2.14 Discounted Problems 34

3 Continuous Time Optimal Control **36**
3.1 The Hamilton Jacobi Bellman (HJB) Equation 37
3.2 Aside on Notation 40
3.2.1 The Minimum Principle 42
3.3 Extensions 45
3.3.1 Fixed Terminal State 45
3.3.2 Free initial state, with cost 46
3.4 Linear Systems and Quadratic Costs 48
3.4.1 Summary 52
3.5 General Problem Formulation 52

Bibliography **56**

Chapter 1

Introduction

1.1 Class Objective

The class objective is to make multiple decisions in stages to minimize a cost that captures undesirable outcomes.

1.2 Key Ingredients

1. Underlying discrete time system:

$$x_{k+1} = f_k\left(x_k, u_k, w_k\right) \qquad k = 0, 1, \ldots, N-1$$

   k: discrete time index

   x_k: state

   u_k: control input, decision variable

   w_k: disturbance or noise, random parameters

   N: time horizon

   f_k: function, captures system evolution

2. Additive cost function:

$$\underbrace{g_N(x_N)}_{\text{terminal cost}} + \underbrace{\sum_{k=0}^{N-1} \underbrace{g_k\left(x_k, u_k, w_k\right)}_{\text{state cost}}}_{\text{accumulated cost}}$$

   g_k is a given nonlinear function.

   · The cost is a function of the control applied

   · Because the w_k are random, typically consider the expected cost:

$$\underset{w_k}{\mathrm{E}}\left(g_N(x_N) + \sum_{k=0}^{N-1} g_k\left(x_k, u_k, w_k\right)\right)$$

Example 1: Inventory Control:
Keeping an item stocked in a warehouse. Too little, you run out (bad). Too much, cost of storage and misuse of capital (bad).

x_k: stock in the warehouse at the *beginning* of k^{th} time period

u_k: stock ordered and immideately delivered at the *beginning* of k^{th} time period

w_k: demand during k^{th} period, with some given probability distribution

Dynamics:

$$x_{k+1} = x_k + u_k - w_k$$

Excess demand is backlogged and corresponds to negative values.

Cost:

$$\mathrm{E}\left(R\left(x_N\right) + \sum_{k=0}^{N-1}\left(r\left(x_k\right) + c\,u_k\right)\right)$$

$r\left(x_k\right)$: penaltise too much stock or negative stock

$c\,u_k$: cost of items

$R\left(x_n\right)$: terminal cost from items at the end that can't be sold or demand that can't be met

Objective: The objective is to minimize the cost subject to $u_k \geq 0$

1.3 Open Loop versus Closed Loop Control

Open Loop: Come up with control inputs $u_0, \ldots, u_{N-1}$ *before* $k = 0$. In Open Loop the objective is to calculate $\{u_0, \ldots, u_{N-1}\}$.

Closed Loop: Wait until time k to make decision. *Asumes* x_k is measurable. Closed Loop will always give better performance, but is computationally much more expensive. In Closed Loop the objective is to calculate the *optimal* rule: $u_k = \mu_k\left(x_k\right)$. $\Pi = \{\mu_0, \ldots, \mu_{N-1}\}$ ist a policy or control law.

Example 2:

$$\mu_k(x_k) = \begin{cases} s_k - x_k & \text{if } x_k < s_k \\ 0 & \text{otherwise} \end{cases}$$

s_k is some threshold.

1.4 Discrete State and Finite State Problem

When state x_k takes on discrete values or is finite in size, it is often convenient to express the dynamics in terms of transition probabilities:

$$P_{ij}(u,k) := Prob\,(x_{k+1} = j \mid x_k = i, u_k = u)$$

i: start state

j: possible future state

u: control input

k: time

This is equivalent to: $x_{k+1} = w_k$, where w_k has the following

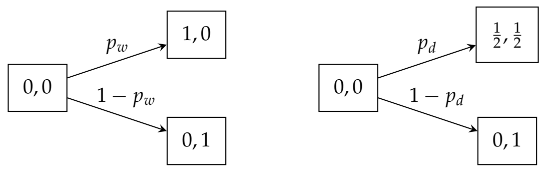
$$Prob\,(w_k = j \mid x_k = i, u_k = u) := P_{ij}(u,k)$$

Example 3: Optimizing Chess Playing Strategies:

- Two game chess match with an opponent, the objective is to come up with a strategy that maximizes the chance to win.
- Each game can have 2 outcomes:
  a) Win by one player: 1 point for the winner, 0 points for the loser.
  b) Draw: 0.5 points for each player.
- If games are tied 1-1 at the end of 2 games, go into sudden death mode until someone wins.
- Decision variable for player, two player styles:
  1) Timid play: Draw with probability p_d, lose with probability $1 - p_d$
  2) Bold play: Win with probability p_w, lose with probability $1 - p_w$
- Asume $p_d > p_w$ as a necessary condition for problem to make sense.

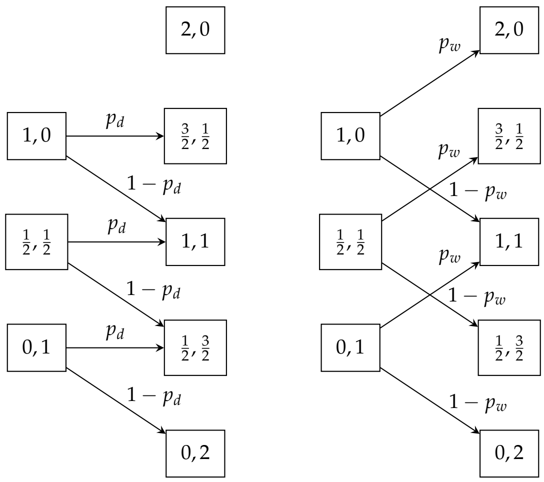
Problem: What playing style should be chosen? Since it doesn't make sense to play Timid if we are tied 1-1 at the end of 2 games, it is a 2-stage-finite problem.

Transition Probability Graph The graphis below show all possible outcomes.



(a) Timid Play (b) Bold Play

Figure 1.1: First Game



(a) Timid Play (b) Bold Play

Figure 1.2: Second Game

Closed Loop Strategy Play timid iff player is ahead.
The probability of winning is:

$$p_d\, p_w + p_w\, ((1 - p_d)\ p_w + p_w\ (1 - p_w)) = p_w^2\, (2 - p_w) + p_w\ (1 - p_w)\ p_d$$

For $\{p_w = 0.45, p_d = 0.9\}$ and $\{p_w = 0.5, p_d = 1\}$ the probabilities to win are 0.54 and 0.625.

Open Loop Strategy Possibilities:

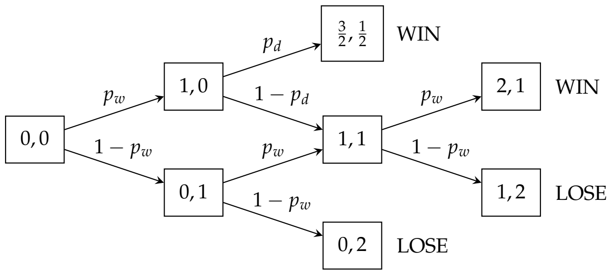


Figure 1.3: Closed Loop Strategy

1) Timid for first 2 games: $p_d^2\, p_w$

2) Bold in both: $p_w^2\,(3-2\,p_w)$

3) Bold in first, timid in second game: $p_w\, p_d + p_w\ (1-p_d)\ p_w$

4) Timid in first, bold in second game: $p_w\, p_d + p_w\ (1-p_d)\ p_w$

Clearly 1) is not the optimal OL strategy, because $p_d^2\, p_w \leq p_d\, p_w + \ldots$
Best strategy yields:

$$p_w^2 + p_w\,(1-p_w)\,max\,(2\,p_w, p_d)$$

if $p_d > 2\,p_w$. The optimal OL strategy is 3) or 4). It can be shown that if $p_w \leq 0.5$, then the probability of winning is ≤ 0.5.

1.5 The Basic Problem

Summarize basic problem setup:

- $x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, \ldots, N-1$

$x_k \in S_k$	state space
$u_k \in C_k$	control space
$w_k \in D_k$	disturbance space

- $u_k \in U(x_k) \subset C_k$. Constrained not only as a function of time, but also of current state.

- $w_k \sim P_R\,(\cdot|x_k, u_k)$. Noise distribution can depend on current state and applied control.

Consider *policies*, or *control laws*,

$$\Pi = \{\mu_0, \mu_1, \dots, \mu_{N-1}\}$$

where μ_k maps state x_k into controls $u_k = \mu_k(x_k)$, such that $\mu_k(x_k) \subset U(x_k) \quad \forall x_k \subset S_k$.
The set of all π called *Admissible Policies*, denoted Π.

- Given a policy $\pi \in \Pi$, the expected cost of starting at state x_0:

$$J_\pi(x_0) := \mathop{\mathrm{E}}_{w_k}\left(g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k)\right)$$

- Optimal Policy: $J_{\pi^*}(x_0) \leq J_\pi(x_0) \quad \forall\, \pi \in \Pi$
- Optimal Cost: $J^*(x_0) := J_{\pi^*}(x_0)$

Chapter 2

Dynamic Programming Algorithm

At the heart of the DP algorithm is the following very simple and intuitive idea.

2.1 Principle of Optimality

Let $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{n-1}^*\}$ be an optimal policy. Assume that in the process of using π^*, a state x_i occurs at time i. Consider the subproblem whereby at time i we are at state x_i and we want to minimize

$$\mathop{\mathrm{E}}_{w_k}\left(g_n(x_n) + \sum_{k=i}^{N-1} g_k(x_k, \mu_k(x_k), w_k)\right).$$

Then the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ is optimal for this problem. The proof is simple: prove by contradiction. If the above were not optimal, you could find a different policy that would give a lower cost. Applying the same policy to the original problem from i would therefore give a lower cost, which contradicts that π^* was an optimal policy.

Example 4: Deterministic Scheduling Problem

- Have 4 machines A, B, C, D, that are used to make something.
- A must occur before B. C before D.

The solution is obtained by calculating the optimal cost for each node, beginning at the bottom of the tree. See figure 2.1.

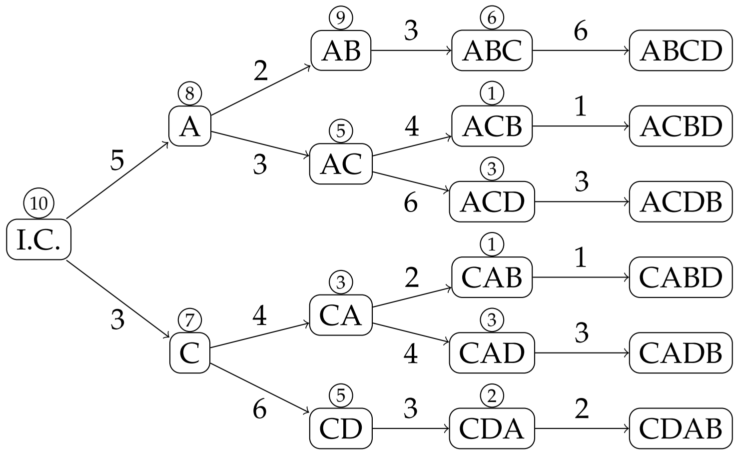


Figure 2.1: Problem of example 4 with optimal cost for each node written above it (in circles).

2.2 The DPA

For every initial state x_0, the optimal cost $J^*(x_0)$ is equal to $J_0(x_0)$, given by the last step of the following recursive algorithm, which proceeds backwards in time from $N-1$ to 0:

Initialization: $J_N(x_N) = g_N(x_N) \quad \forall\, x_n \subset S_N$

Recursion: For the recursion we use

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{\mathrm{E}}_{w_k} \Big(g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big).$$

where expectation is taken with respect to $P_R(\cdot|x_k, u_k)$.

Furthemore, if $u_k^* = \mu_k^*(x_k)$ minimizes the recursion equation for each x_k and k, the policy $\pi^* = \{\mu_0^*, \ldots, \mu_{N-1}^*\}$ is optimal.

Comments

- For each recursion step, we have to perform the optimization over **all** possible values $x_k \in S_k$, since we don't know a priori which states we will actually visit.
- This pointwise optimization is what gives us μ_k^*.

Proof 1: (Read section 1.5 in [1] if you are mathematically inclined).

- Denote $\pi^k := \{\mu_k, \mu_{k+1}, \dots, \mu_{N-1}\}$.
- Denote

$$J_k^*(x_k) = \min_{\pi^k} \mathop{\mathrm{E}}_{w_k,\dots,w_{N-1}} \left(g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right).$$

  optimal cost when starting at time k, we find ourselves at state x_k.

- $J_N^*(x_N) = g_N(x_N)$, finally.
- We will show that $J_k^* = J_k$ generated by the DPA which give us the desired result when $k = 0$.

Induction: $J_N^*(x_N) = J_N(x_N)$, $\therefore$ true for $k = N$
Assume true for $k+1$: $J_{k+1}^*(x_{k+1}) = J_{k+1}(x_{k+1}) \quad \forall\, x_{k+1} \in S_{k+1}$.
Then, since $\pi^k = \{\mu_k, \pi^{k+1}\}$, have

$$J_k^*(x_k) = \min_{(\mu_k,\pi^{k+1})} \mathop{\mathrm{E}}_{w_k,\dots,w_{N-1}} \left(g_k(x_k, \mu_k(x_k), w_k) + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\left(x_i, \mu_i(x_i), w_i\right) \right)$$

by principle of optimality:

$$= \min_{\mu_k} \mathop{\mathrm{E}}_{w_k} \left(g_k(x_k, \mu_k(x_k), w_k) + \min_{\pi^{k+1}} \mathop{\mathrm{E}}_{w_{k+1},\dots,w_{N-1}} \left(g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\left(x_i, \mu_i(x_i), w_i\right) \right) \right)$$

by definition of J_{k+1}^* and update equation:

$$= \min_{\mu_k} \mathop{\mathrm{E}}_{w_k} \left(g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}^*(f_k(x_k, \mu_k(x_k), w_k)) \right)$$

by induction hypothesis:

$$= \min_{\mu_k} \mathop{\mathrm{E}}_{w_k} \left(g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}(f_k(x_k, \mu_k(x_k), w_k)) \right)$$

$$= \min_{u_k \in U_k(x_k)} \mathop{\mathrm{E}}_{w_k} \left(g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \right)$$

$$= J_k(x_k)$$

In other words: Search over a function is simply like solving what the function does partwise. □

$J_k(x_k)$ is called the *cost-to-go* at state x_k.
$J_k(\cdot)$ is called the *cost-to-go* function.

2.3 Chess Match Strategy Revisited

Recall

· Timid Play, prob. tie $= p_d$, prob. loss $= 1 - p_d$

- Bold Play, prob. win $= p_w$, prob. loss $= 1 - p_w$
- 2 game match, + tie breaker if necessary

Objective Find policy which maximizes probability of winning. We will solve with DP, replace min by max.
Asume $p_d > p_w$.

Define $x_k =$ difference between our score and opponent score at the end of game k. Recall 1 point for win, 0 for loss and 0.5 for tie.

Define $J_k(x_k) =$ probability of winning match at time k if state $= x_k$.

Start Recursion

$$J_2(x_2) = \begin{cases} 1 & \text{if } x_2 > 0 \\ p_w & \text{if } x_2 = 0 \\ 0 & \text{if } x_2 < 0 \end{cases}$$

Recursive Equation

$$J_k(x_k) = \max\left[\underbrace{p_d\, J_{k+1} + (1-p_d)\, J_{k+1}(x_k - 1)}_{\text{timid}}, \underbrace{p_w\, J_{k+1} + (1-p_w)\, J_{k+1}(x_k-1)}_{\text{bold}}\right]$$

Convince yourself that this is equivalent to the formal definitions:

$$J_k(x_k) = \max_{u_k} \mathop{\mathrm{E}}_{w_k}\Big(g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k))\Big)$$

Note: There is only a terminal cost in this problem.

$$J_1(x_1) = \max\left[p_d\, J_2(x_1) + (1-p_d)\, J_2(x_1 - 1), p_w\, J_2(x_1+1) + (1-p_w)\, J_2(x_1-1)\right]$$

- If $x_1 = 1$: $\max\left[\underbrace{p_d + (1-p_d)\, p_w}_{\text{timid}}, \underbrace{p_w + (1-p_w)\, p_w}_{\text{bold}}\right]$
  Which is bigger? Timid - Bold $= (p_d - p_w)(1-p_w) > 0 \therefore$ Timid is optimal, and $J_1(1) = p_d + (1-p_d)\, p_w$.
- If $x_1 = 0$: $\max\left[\underbrace{p_d\, p_w + (1-p_d)\, 0}_{\text{timid}}, \underbrace{p_w + (1-p_w)\, 0}_{\text{bold}}\right]$
  Optimal is p_w, and $J_1(0) = p_w$, Bold is optimal strategy.
- If $x_1 = -1$: $\max\left[0, p_w^2\right]$
  $J_1(-1) = p_w^2$, optimal strategy is Bold.

$$\begin{aligned} J_0(0) &= \max\left[p_d\, J_1(0) + (1-p_d)\, J_1(-1), p_w\, J_1(1) + (1-p_w)\, J_1(-1)\right] \\ &= \max\left[p_d\, p_w + (1-p_d)\, p_w^2, p_w\left(p_d + (1-p_d)\, p_w\right) + (1-p_w)\, p_w^2\right] \\ &= \max\left[p_d\, p_w + (1-p_d)\, p_w^2, p_d\, p_w + (1-p_d)\, p_w^2 + (1-p_w)\, p_w^2\right] \end{aligned}$$

$\therefore J_0(0) = p_d\, p_w + (1-p_d)\, p_w^2 + (1-p_w)\, p_w^2$, the optimal strategy is Bold.

Optimal Strategy If ahead, play Timid.

2.4 Converting non-standard problems to Basic Problem

2.4.1 Time Lags

Assume update equation is of the following form:

$$x_{x+1} = f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k)$$

Define $y_k = x_{k-1}, s_k = u_{k-1}$

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ s_{k+1} \end{pmatrix} = \underbrace{\begin{pmatrix} f_k(x_k, y_k, s_k, w_k) \\ x_k \\ u_k \end{pmatrix}}_{\tilde{f}_k}$$

Let $\tilde{x}_k = (x_k, y_k, s_k)$, $\tilde{x}_{k+1} = \tilde{f}_k(\tilde{x}_k, u_k, w_k)$.
The control is $u_k = \mu_k(x_k, u_{k-1}, x_{k-1})$.
This can be generalized to more than one time lag.

2.4.2 Correlated Disturbances

If disturbances are not independent, but can be modeled as the output of a system driven by independent disturbances $\rightarrow$ *Colored Noise*.

Example 5:

$w_k = C_k\, y_{k+1}$
$y_{k+1} = A_k\, y_k + \xi_k$

A_k, C_k are given, $\{\xi_k\}$ is independent.
As usual, $x_{k+1} = f_k(x_k, u_k, w_k)$.

$$\therefore \begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k\big(x_k, u_k, C_k\left(A_k\, y_k + \xi_k\right)\big) \\ A_k\, y_k + \xi_k \end{pmatrix} \text{ and } u_k = \mu_k(x_k, y_k)$$

which is now in the standard form. In general, y_k cannot be measured and must be estimated.

2.4.3 Forecasts

When state information includes knowledge of probability distributions. At the beginning of each period k, we receive information about w_{k+1} probability distribution. In particular, assume w_{k+1} could have the following probability distributions $\{Q_1, Q_2, \ldots, Q_m\}$, with a priori probabilities $p_1, \ldots, p_m$. At time k, we receive *forecast* i that Q_i is used to generate w_{k+1}. Model as follows: $y_{k+1} = \xi_k, \xi_k$ is a random variable, taking value i with probability p_i. In particular, w_k has probability distribution Q_{y_k}.

Then have $\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, w_k) \\ \xi_k \end{pmatrix}$. New state $\tilde{x}_k = (x_k, y_k)$.
Since y_k is known at time k, we have a Basic Problem formulation. New disturbance $\tilde{w}_k = (w_k, \xi_k)$, depends on current state, which is allowed. DPA takes on the following form:

$$J_N(x_N, y_N) = g_N(x_N)$$

$$\begin{aligned} J_k(x_k, y_k) &= \min_{u_k} \mathop{\mathrm{E}}_{w_k} \mathop{\mathrm{E}}_{\xi_k} \left(g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k), \xi_k\big) \,\Big|\, y_k \right) \\ &= \min_{u_k} \mathop{\mathrm{E}}_{w_k} \left(g_k(x_k, u_k, w_k) + \sum_{i=1}^{m} p_i\, J_{k+1}\big(f_k(x_k, u_k, w_k), i\big) \,\Big|\, y_k \right) \end{aligned}$$

where conditional expectation simply means that w_k has probability distribution Q_{y_k}. $y_k \in \{1, \ldots, m\}$, expectation over w_k is taken with respect to the distribution Q_{y_k}.

2.5 Deterministic, Finite State Systems

Recall Basic Problem

$$x_{k+1} = f_k(x_k, u_k, w_k) \qquad k = 0, \dots, N-1$$
$$g_k(x_k, u_k, w_k) \text{ cost at stage } k.$$

Consider Problems where

1. $x_k \in S_k$, S_k is a finite set
2. No disturbances w_k

We assume, without loss of generality, that there is only one way to go from state $i \in S_k$ to $j \in S_{k+1}$ (If there is more than one way, pick one with lowest cost at stage k).

2.5.1 Convert DP to Shortest Path Problem

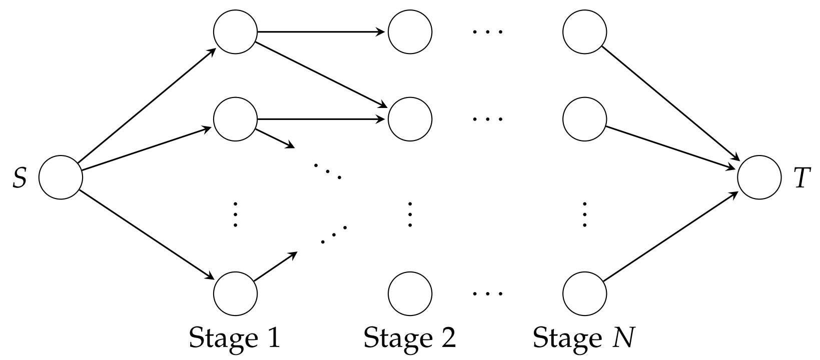


Figure 2.2: General Shortest Path Problem

a_{ij}^k = cost to go from state $i \in S_k$ to state $j \in S_{k+1}$, time k. This is equal ∞ if there is no way to go from $i \in S_k$ to $j \in S_{k+1}$.

a_{iT}^N = terminal cost of state $i \in S_N$

In other words,

$$a_{ij}^k = g_k(i, u_k^{ij}), \quad \text{where } j = f_k(i, u_k^{ij})$$
$$a_{iT}^N = g_N(i)$$

2.5.2 DP algorithm

$$
\begin{aligned}
J_N(i) &= a_{iT}^N && i \in S_N \\
J_k(i) &= \min_{j \in S_{k+1}} \left(a_{ij}^k + J_{k+1}(j) \right) && i \in S_k, \quad k = 0, \dots, N-1
\end{aligned}
$$

This solves shortest path problem.

2.5.3 Forward DP algorithm

By inspection, the problem is symmetric. Shortest path from S to T is the same as from T to S, motivating following algorithm, $\tilde{J}_k(j)$ optimal cost to *arrive* to state j:

$$
\begin{aligned}
\tilde{J}_N(j) &= a_{sj}^0 && j \in S_1 \\
\tilde{J}_k(j) &= \min_{i \in S_{N-k}} \left(a_{ij}^{N-k} + \tilde{J}_{k+1}(i) \right) && j \in S_{N-k+1}, \quad k = 1, \dots, N \\
\tilde{J}_0(T) &= \min_{i \in S_N} \left(a_{iT}^N + \tilde{J}_1(i) \right) \\
\tilde{J}_0(T) &= J_0(s)
\end{aligned}
$$

2.6 Converting Shortest Path to DP

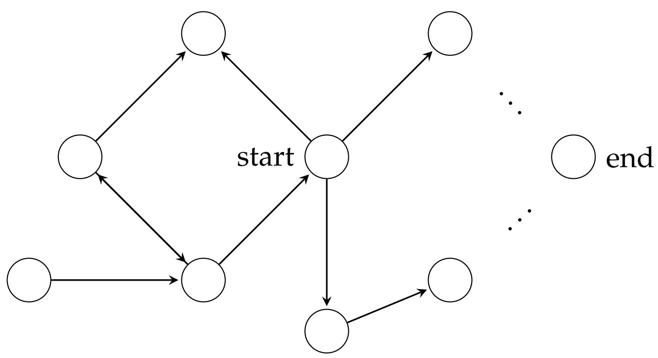


Figure 2.3: Another Path Problem, in which circles are allowed

As an example for a mental picture, one could imagine cities on a map.

- Let $\{1, 2, \ldots, N, T\}$ be the nodes of graph, a_{ij} the cost to move from i to j, $a_{ij} = \infty$ if there is no edge. i and j denote nodes, as opposed to previous section where they were states.

- Assume that all cycles have non-negative cost. This isn't an issue if all edges have cost ≥ 0.

- Note that with above assumption, an optimal path $\leq$ length N (visits all nodes).

Setup problem where we require exactly N moves, degenerate moves are allowed ($a_{ii} = 0$).

$J_k(i)$ optimal cost of getting from i to T in $N - k$ moves

$J_N(i) = a_{iT}$ (can be infinite, of course)

$J_k(i) = \min_j \left(a_{ij} + J_{k+1}(j) \right)$ (optimal $N - k$ move is $a_{ij}+$ optimal $N - k - 1$ move from j).

- Notice that degenerate moves are allowed. (remove in the end)

- Terminate procedure if $J_k(i) = J_{k+1}(i) \quad \forall\, i$.

2.7 Viterbi Algorithm

This is a powerful combination of D.P. and Bayes Rule for optimal estimation.

- Given Markov Chain, with state transition probabilities p_{ij}.

$$p_{ij} = P\left(x_{k+1} = j | x_k = i\right) \qquad 1 \leq i, j \leq M$$

- $p(x_0)$ = initial probability for starting state.

- Can only *indirectly* observe state via measurement.

$$r(z; i, j) = P\left(meas = z | x_k = i, x_{k+1} = j\right) \quad \forall\, k$$

  where P is the likelihood function.

Objective: Given $Z_N = \{z_1, \dots, z_N\}$ measurements, construct $\hat{X}_N = \{\hat{x}_0, \dots, \hat{x}_N\}$ that maximizes over all $X_N = \{x_0, \dots, x_N\}$ $P_R(X_N|Z_N)$. Most likely state.

- Recall $P_R(X_N, Z_N) = P_R(X_N|Z_N)\, P_R(Z_N)$.

For a given Z_N, maximizing $P_R(X_N, Z_N)$ over X_N gives same result as maximizing $P_R(X_N|Z_N)$ over X_N.

$$\begin{aligned}
P_R(X_N, Z_N) &= P_R(x_0, \dots, x_N, z_1, \dots, z_N) \\
&= P_R(x_1, \dots, x_N, z_1, \dots, z_N|x_0)\, P_R(x_0) \\
&= P_R(x_2, \dots, x_N, z_2, \dots, z_N|x_0, x_1, z_1)\, P_R(x_1, z_1|x_0)\, P_R(x_0) \\
&= P_R(x_2, \dots, x_N, z_2, \dots, z_N|x_0, x_1, z_1)\, P_R(z_1|x_0, x_1)\, P_R(x_1|x_0)\, P_R(x_0) \\
&= P_R(x_2, \dots, x_N, z_2, \dots, z_N|x_0, x_1, z_1)\, r(z_1; x_0, x_1) p_{x_0, x_1} P_R(x_0)
\end{aligned}$$

One more step:

$$\begin{aligned}
&P_R(x_2, \dots, x_N, z_2, \dots, z_N|x_0, x_1, z_1) \\
&= P_R(x_3, \dots, x_N, z_3, \dots, z_N|x_0, x_1, z_1, z_2, x_2)\, P_R(x_2, z_2|x_0, x_1, z_1) \\
&= P_R(x_3, \dots, x_N, z_3, \dots, z_N|x_0, x_1, z_1, z_2, x_2)\, P_R(z_2|x_0, x_1, z_1, x_2)\, P_R(x_2|x_0, x_1, z_1) \\
&= P_R(x_3, \dots, x_N, z_3, \dots, z_N|x_0, x_1, z_1, z_2, x_2)\, r(z_2; x_1, x_2) p_{x_1, x_2}
\end{aligned}$$

Keep going, and one gets:

$$P_R(X_N, Z_N) = P_R(x_0) \prod_{k=1}^{N} p_{x_{k-1}, x_k} r(z_k; x_{k-1}, x_k)$$

Assume, that all quantities > 0. If $= 0$, can modify algorithm.
Since the above is a strictly positive property (by above assumptions), and log function is monotonically increasing as a function of its argument, we can maximize

$$\log\left(P_R(X_N, Z_N)\right) = \min_{X_N} \left(-\log(P_R(x_0)) + \sum_{k=1}^{N} -\log\left(p_{x_{k-1}, x_k} r(z_k; x_{k-1}, x_k)\right) \right)$$

Forward DP At time k, we can calculate cost to **arrive** to any state. We don't have to wait until the end to solve the problem.

2.8 Shortest Path Algorithms

Look at alternatives to DP for problems that are finite and deterministic.

$$\text{Path} \equiv \text{Length} \equiv \text{Cost}$$

2.8.1 Label Correcting Methods

Assume $a_{ij} \geq 0$. Arclength = cost to go from node i to node $j \geq 0$.

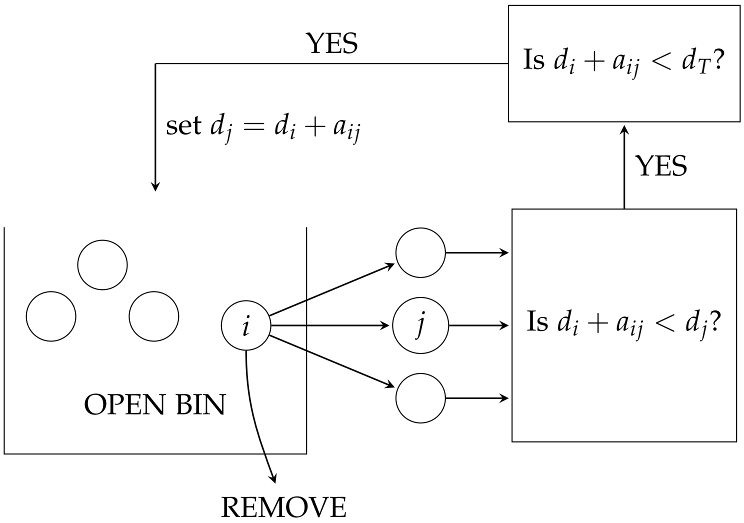


Figure 2.4: Diagram of the label correcting algorithm.

Let d_i be shortest path to i so far.

Step 0 Place Node S in OPEN BIN, set $d_S = 0$, $d_j = \infty \quad \forall j$.

Step 1 Remove a node i from OPEN, and execute Step 2 for all children j of i.

Step 2 If $d_i + a_{ij} < \min(d_j, d_T)$, set $d_j = d_i + a_{ij}$, set i to be the parent of j. If $j \neq T$, place j in OPEN if it is not already there.

Step 3 If OPEN is empty, done. If not, go back to Step 1.

Example 6: Deterministic Scheduling Problem (revisited)

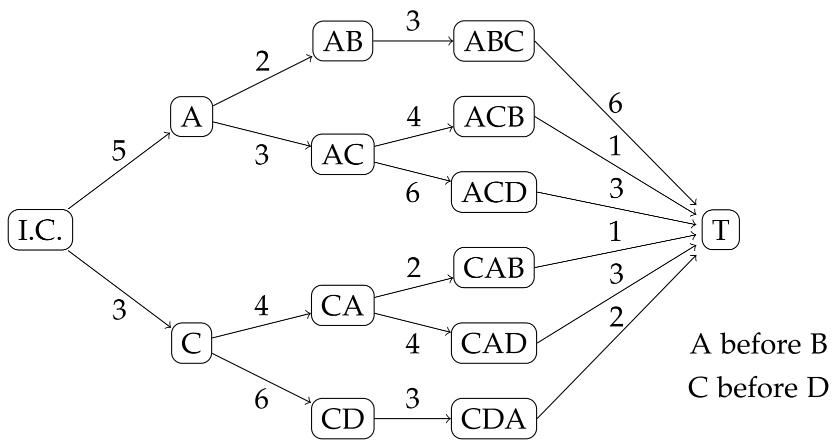


Figure 2.5: Deterministic Scheduling Problem.

Iteration #	Remove	OPEN	d_t	OPTIMAL
0	–	$S(0)$	∞	–
1	S	$A(5), C(3)$	∞	–
2	C	$A(5), CA(7), CD(9)$	∞	–
3	CD	$A(5), CA(7), CDA(12)$	∞	–
4	CDA	$A(5), CA(7)$	14	$CDAB$
5	CA	$A(5), CAB(9), CAD(11)$	14	$CDAB$
6	CAD	$A(5), CAB(9)$	14	$CDAB$
7	CAB	$A(5)$	10	$CABD$
8	A	$AB(7), AC(8)$	10	$CABD$
9	AC	$AB(7)$	10	$CABD$
10	AB	–	10	$CABD$

Done, optimal cost $= 10$, optimal path $= CABD$.

Different ways to remove items from OPEN give different, well known, algorithms.

Depth-First Search Last in, first out. What we did in example. Finds feasible path quickly. Also good if you have limited memory.

Best-First Search Remove best label. Dijkstra's method. Remove step is more expensive, but can give good performance.

Brendth-First Search First in, first out. Bellman-Ford.

2.8.2 A*− Algorithm

Workhouse for many AI applications, path planning.
Basic idea: Replace test $d_i + a_{ij} < d_T$ by $d_i + a_{ij} + h_j < d_T$, where h_j is a *lower bound* to the shortest distance from j to T. Indeed, if $d_j + a_{ij} + h_j \geq d_T$, clear that path going through j will not be optimal.

2.9 Multi-Objective Problems

Example 7: Motivation: care about time *and* fuel.

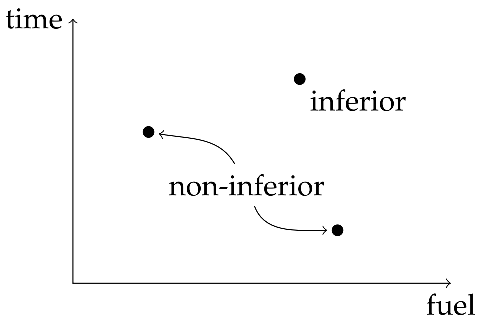


Figure 2.6: Possibilities in the time-fuel graph.

A vector $x = (x_1, x_2, \ldots, x_M) \in S$ is non-inferior if there are no other $y \in S$ so that $y_l \leq x_l$, $l = 1, \ldots, M$, with strict inequality for one of these ls.

Given a problem with M cost functions $f_1(x), \ldots, f_M(x) \quad x \in X$ is a non-inferior solution if the vector $(f_1(x), \ldots, f_M(x))$ is a non-inferior vector of set $\{(f_1(y), \ldots, f_M(y)) \,|\, y \in X\}$.

Reasonable goal: find *all* non-inferior solutions, then use another criterion to pick which one you actually want to use.

How this applies to deterministic, finite state DP (which are equivalent to shortest path problems):

$$x_{k+1} = f_k(x_k, u_k) \qquad \text{Dynamics}$$

$$g_N^l + \sum_{k=0}^{N-1} g_k^l(x_k, u_k) \qquad l = 1, \dots, M$$

2.9.1 Extended Principle of Optimality

If $\{u_k, \dots, u_{N-1}\}$ is a non-inferior control sequence for the tail subproblem that starts at x_k, then $\{u_{k+1}, \dots, u_{N-1}\}$ is also non-inferior for the tail subproblem that starts at $f_k(x_k, u_k)$. Simple proof: by contradiction.

Algorithm First define what we will do recursion over:

- $F_k(x_k)$: the set of M-tuples (vectors of size M) of cost to go at x_k which are non-inferior.
- $F_N(x_N) = \{(g_N^1(x_N), \dots, g_N^M(x_N))\}$. Only one element in set for each x_N.
- Given $F_{k+1}(x_{k+1})\ \forall x_{k+1}$, generate for each state x_k the set of vectors $(g_k^l(x_k, u_k) + c^1, \dots, g_k^M(x_k, u_k) + c^M)$, such that $(c^1, \dots, c^M) \in F_{k+1}(f_k(x_k, u_k))$.

These are all possible costs that are consistent with $F_{k+1}(x_{k+1})$. Then to obtain $F_k(x_k)$ simply extract all non-inferior elements.

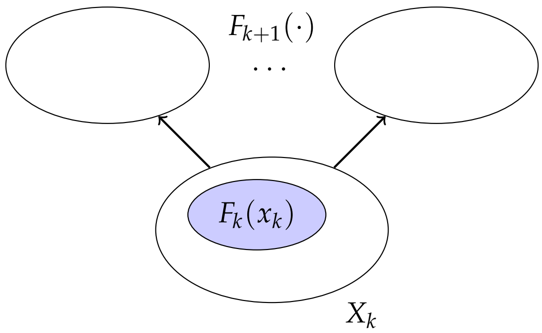


Figure 2.7: Possible sets for F_{k+1}.

When we calculate $F_0(x_0)$, will have all non-inferior solutions.

2.10 Infinite Horizon Problems

Consider the time (or iteration) invariant case:

$$x_{k+1} = f(x_k, u_k, w_k) \qquad \begin{aligned} &x_k \in S \\ &u_k \in U \\ &w_k \sim P(\cdot|x_k, u_k) \end{aligned}$$

$$J_\pi(x_0) = \mathrm{E}\left(\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k), w_k)\right), \text{ no terminal cost}$$

Write down DP algorithm:

$$\begin{aligned} J_N(x_N) &= 0 \\ J_k(x_k) &= \min_{u_k \in U} \mathop{\mathrm{E}}_{w_k} \left(g(x_k, u_k, w_k) + J_{k+1}\left(f(x_k, u_k, w_k)\right)\right) \quad \forall k \end{aligned}$$

Question: What happens as $N \to \infty$? Does the problem become easier?
Yes. Reason: lose notion of time. For very large class of problems, have *Bellman Equation*:

$$J^*(x) = \min_u \mathop{\mathrm{E}}_w \left(g(x, u, w) + J^*\left(f(x, u, w)\right)\right) \quad \forall x \in S$$

Bellman Equation involves solving for optimal cost to go function $J^*(x) \forall x \in S$.
$u = \mu(x)$ gives optimal policy ($\mu(\cdot)$ is obtained from solution to Bellman Equation: for every x there is a u).

- Efficient methods for solving Bellman Equation
- Technical conditions on when this can be done.

2.11 Stochastic, Shortest Path Problems

$$\begin{aligned} &x_{k+1} = w_k && x_k \in S, \text{ finite set} \\ &P_R(w_k = j | x_k = i, u_k = u) = p_{ij}(u) && u_k \in U(x_k), \text{ finite set} \end{aligned}$$

We have a finite number of states. The transition from one state to the next is dictated by $p_{ij}(u)$: probability that next state is j given current state is i. u is the constrol input, we can control what these transition probabilites are, finite set of options $u \in U(i)$. Problem data is time (or iteration) independent.
Cost:
Given initial state i and a policy $\pi = \{\mu_0, \mu_1, \ldots\}$.

$$J_\pi(i) = \lim_{N\to\infty} \mathrm{E}\left(\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k))|x_o = i\right).$$

- Optimal cost from state i: J^*(i)
- Stationary policy $\pi = \{\mu, \mu, \ldots\}$. Denote $J_\mu(i)$ as the resulting cost. $\{\mu, \mu, \ldots\}$ is simply referred to as μ. μ is optimal if

$$J_\mu(i) = J^*(i) = \min_\pi J_\pi(i)$$

Assumptions:

- Existence of a cost-free termination state t

$$p_{tt}(u) = 1 \quad \forall u$$
$$g(t, u) = 0 \quad \forall u.$$

  Sufficient condition to make cost meaningfull. Think of this as a destination state.

- $\exists$ integer m such that for all admissible policies

$$\rho_\pi = \max_{i=1,\ldots,n} P_R\left(x_m \neq t|x_0 = i, \pi\right) < 1$$

This is a strong assumption, which is only required for proofs.

2.11.1 Main Result

A) Given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence

$$J_{k+1}(i) = \min_{u\in U(i)}\left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u)J_k(j)\right) \quad \forall i$$

converges to optimal cost $J^*(i)$ for each i.

B) Optimal cost satisfies Bellman's Equation:

$$J^*(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J^*(j) \right) \quad \forall i$$

which has a **unique** solution.

2.11.2 Sketch of Proof

A0) First prove that cost is bounded.

Recall $\exists\, m$ such that $\forall$ policies π,

$$\rho_\pi := \max_i P_R(x_m \neq t | x_0 = i, \pi) < 1$$

Since all problem data is finte, $\rho := \max_\pi \rho_\pi < 1$

$$\therefore P_R(x_{2m} \neq t | x_0 = i, \pi) = P_R(x_{2m} \neq t | x_m \neq t, x_0 = i, \pi) \cdot P_R(x_m \neq t | x_0 = i, \pi) \leq \rho^2$$

Generally, $P_R(x_{km} \neq t | x_0 = i, \pi) \leq \rho^k$.
Furthermore, the cost incurred between the m periods km and $(k+1)m - 1$ is

$$m\rho^k \max_{i,u} |g(i,u)| =: \rho^k M \qquad M := m \cdot \max_{i,u} |g(i,u)|$$

$$\therefore J_\pi(i) \leq \sum_{k=0}^{\infty} M\rho^k = \frac{M}{1-\rho} \text{ finite}$$

A1)

$$\begin{aligned} J_\pi(x_0) &= \lim_{N\to\infty} \mathrm{E}\left(\sum_{k=0}^{N-1} g\left(x_k, \mu_k(x_k)\right) \right) \\ &= \mathrm{E}\left(\sum_{k=0}^{mK-1} g\left(x_k, \mu_k(x_k)\right) \right) + \lim_{N\to\infty} \mathrm{E}\left(\sum_{k=mK}^{N-1} g\left(x_k, \mu_k(x_k)\right) \right) \end{aligned}$$

by previous, we know that

$$\left| \lim_{N\to\infty} E\left(\sum_{k=mK}^{N-1} g\left(x_k, \mu_k(x_k)\right) \right) \right| \leq \frac{M\rho^K}{1-\rho}$$

As expected, we can make tail as small as we want.

A2) Recall that we can view J_0 as terminal cost funtion, with $J_0(i)$ given. Bound its expected vallue:

$$\begin{aligned}
|E\left(J_0(x_{mK})\right)| &= \left|\sum_{i=1}^{n} P_R(x_{mK}=i|x_0,\pi)J_0(i)\right| \\
&\leq \left(\sum_{i=1}^{n} P_R(x_{mK}=i|x_0,\pi)\right)\max_i |J_0(i)| \\
&\leq \rho^K \max_i |J_0(i)|
\end{aligned}$$

A3) "Sandwich"

$$\begin{aligned}
&\mathrm{E}\left(J_0(x_{mK})\right) + \mathrm{E}\left(\sum_{k=0}^{mK-1} g(x_k,\mu_k(x_k))\right) = \\
&\qquad\qquad \mathrm{E}\left(J_0(x_{mK})\right) + J_\pi(x_0) - \lim_{N\to\infty}\mathrm{E}\left(\sum_{k=mK}^{N-1} g(x_k,\mu_k(x_k))\right)
\end{aligned}$$

Recall if $a = b + c$, then

$$\begin{aligned}
&a \leq b + |c| \leq b + \bar{c}, \quad \bar{c} \geq |c| \\
&a \geq b - |c| \geq b - \bar{c}
\end{aligned}$$

Follows that

$$\begin{aligned}
-\rho^K \max_i |J_0(i)| - \frac{M\rho^K}{1-\rho} + J_\pi(x_0) &\leq \mathrm{E}\left(J_0(x_{mK}) + \sum_{k=0}^{mK-1} g(x_k,\mu_k(x_k))\right) \\
&\leq \rho^K \max_i |J_0(i)| + \frac{M\rho^K}{1-\rho} + J_\pi(x_0)
\end{aligned}$$

A4) We take minimum over all policies, the middle term is exactly our DP recursion of part A. Take limits, get

$$\lim_{K\to\infty} J_{mK}(x_0) = J^*(x_0)$$

Now we are almost done. But since

$$|J_{mK+1}(x_0) - J_{mK}(x_0)| \leq \rho^K M$$

have

$$\lim_{k\to\infty} J_k(x_0) = J^*(x_0)$$

Summary

A1: bound the tail over all policies

A2: bound contribution from initial condition $J_0(i)$, over all policies

A3: "sandwich" type of bounds, middle term is DP recursion

A4: optimized over all policies, took limit.

2.11.3 Prove B

Prove that optimal cost satisfies Bellman's equation

$$J_{k+1}(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right).$$

In Part A, we showed that $J_k(\cdot) \to J_*(\cdot)$, just take limits on both sides. To prove uniqueness, just use solution of Bellman equation as initial condition of DP iteration.

2.12 Summary of previous lecture

Dynamics

$$x_{k+1} = w_k$$
$$P_R\{w_k = j | x_k = j, u_k = u\} = p_{ij}, \quad u \in U(i) \quad \text{finite set}$$

$x_k \in S$ finite $\quad S = \{1, 2, \ldots, n, t\}$
$p_{tt} = 1 \qquad \forall u \in U(t)$

Cost Given $\pi = \{\mu_0, \mu_1, \ldots\}$

$$J_\pi(i) = \lim_{N \to \infty} \mathrm{E} \left(\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k)) | x_0 = i \right) \quad \forall i \in S$$

$g(t, u) = 0 \quad \forall u \in U(t)$

$$J^*(i) = \min_\pi J_\pi(i) \quad \text{optimal cost}$$

Note: $J_\pi(t) = 0 \quad \forall \pi \quad \Rightarrow J^*(t) = 0$

Result

A) Given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence

$$J_{k+1}(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right) \quad i \in S \setminus t = \{1, \ldots, n\}$$

converges to $J^*(i)$.
Note: there is a lot of a short-cut here, can include terminal state t, provided we pick $J_0(t) = 0$. Does not change equations.

B)

$$J^*(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u)\, J^*(j) \right) \quad i \in S \setminus t$$

Bellman's Equation: Also gives optimal policy, which is in fact stationary.

2.13 How do we solve Bellman's Equation?

2.13.1 Method 1: Value iteration (VI)

Use the DP recursion of result A:

$$J_{k+1}(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u)\, J_k(j) \right) \quad i \in S \setminus t$$

until it converges. $J_0(i)$ can be set to a guess, if the guess is good, it will speed up convergence. How do we know that we are close to converging? Exploit problem structure to get bounds, look at [1].

2.13.2 Method 2: Policy Iteration (PI)

Iterate over policy instead of values. Need following result:

C) For *any* stationary policy μ, the costs $J_\mu(i)$ are the unique solutions of

$$J(i) = g(i, \mu(i)) + \sum_{j=1}^{n} p_{ij}(\mu(i))\, J(j) \quad i \in S \setminus t$$

Furthermore, given any initial conditions $J_0(i)$, the sequence

$$J_{k+1}(i) = g(i, \mu(i)) + \sum_{j=1}^{n} p_{ij}(\mu(i))\, J_k(j)$$

converges to $J_\mu(i)$ for each i.
Proof: trivial. Consider problem where *only* allowable control at state i is $\mu(i)$, and apply parts A and B. Special case of general theorem.

Algorithm for PI From now on, $i \in S \setminus t = \{1, 2, \ldots, n\}$.

Stage 1 Given μ^k (stationary policy at iteration k, *not* policy at time k), solve for the $J_{\mu^k}(i)$ by solving

$$J(i) = g(i, \mu^k(i)) + \sum_{j=1}^{n} p_{ij}(\mu^k(i))\, J(j) \quad \forall i$$

n equations, n unknowns (Result C).

Stage 2 Improve policy.

$$\mu^{k+1}(i) = \arg \min_{u \in U(i)} \left(g(i, u) + \sum_{j=1}^{n} p_{ij}(u)\, J_{\mu^k}(j) \right) \quad \forall i$$

Iterate, quit when $J_{\mu^{k+1}}(i) = J_{\mu^k}(i) \quad \forall i$

Theorem Above terminates after a *finite* # of steps, and converges to optimal policy.

Proof two steps

1) We will first show that

$$J_{\mu^k}(i) \geq J_{\mu^{k+1}}(i) \quad \forall i, k$$

2) We will show that what we converge to satisfies Bellman's Equation.

1) For *fixed* k, consider the following recursion in N:

$$J_{N+1}(i) = g(i, \mu^{k+1}(i)) + \sum_{j=0}^{n} p_{ij}(\mu^{k+1}(i))\, J_N(j) \quad \forall i$$

$$J_0(i) = J_{\mu^k}(i)$$

By result C), $J_N \to J_{\mu^{k+1}}$ as $N \to \infty$.

$$\begin{aligned} J_0(i) &= g(i, \mu^k(i)) + \sum_j p_{ij}(\mu^k(i))\, J_0(j) \\ &\geq g(i, \mu^{k+1}(i)) + \sum_j p_{ij}(\mu^{k+1}(i))\, J_0(j) = J_1(i) \\ J_1(i) &\geq g(i, \mu^{k+1}(i)) + \sum_j p_{ij}(\mu^{k+1}(i))\, J_1(j) \end{aligned}$$

since $J_1(i) \leq J_0(i)$. But $g(i, \mu^{k+1}(i)) = J_2(i)$, keep going, get

$$J_0(i) \geq J_1(i) \geq \ldots \geq J_N(i) \geq \ldots$$

take limit

$$J_{\mu^k}(i) \geq J_{\mu^{k+1}}(i) \quad \forall i$$

Since the # of stationary policies is finite, we will eventually have $J_{\mu^k}(i) = J_{\mu^{k+1}}\ \forall i$ for some finite k.

2) It follows from Stage 2 that

$$J_{\mu^{k+1}}(i) = J_{\mu^k}(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_j p_{ij}(u)\, J_{\mu^k}(j) \right)$$

when converged, but this is Bellman's Equation! $\therefore$ have converged to optimal policy.

Discussion

Complexity

Stage 1 linear sytem of equations, size n, comlexity $\approx \mathcal{O}(n^3)$

Stage 2 n minimizations, p choices (p different values of u that I can use), complexity $\approx \mathcal{O}(p\, n^2)$

Put together: $\mathcal{O}(n^2(n+p))$.

Worst case # of iterations: search over all policies, p^n. But in practice, converges very quickly.

Why does Policy Iteration converge so quickly relative to Value Iteration?

Rewrite Value Iteration

Stage 2 $\mu^k = \arg \min_{u \in U(i)} \left(g(i,u) + \sum_j p_{ij}(u)\, J_k(j) \right)$

$\uparrow$ iterate

Stage 1 $J_{k+1}(i) = g(i, \mu^k(i)) + \sum_j p_{ij}(\mu^k(i))\, J_k(j)$

2.13.3 Third Method: Linear Programming

Recall: Bellman's Equation

$$J^*(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}(u)\, J^*(j) \right) \quad i = 1, \ldots, n$$

and Value Iteration:

$$J_{k+1}(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}\, J_k(j) \right) \quad i = 1, \ldots, n$$

We showed that Value Iteration (V.I.) converges to optimal cost to go J^* for all initial "guesses" J_0.
Assume we start V.I. with **any** J_0 that satisfies

$$J_0(i) \leq \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}\, J_0(j) \right) \quad i = 1, \ldots, n$$

If follows that $J_1(i) \geq J_0(i)$ for all i.

$$\therefore J_1(i) \leq \min_{u \in U(i)} \left(g(i,u) + \sum_{j=1}^{n} p_{ij}\, J_1(j) \right) \quad i = 1, \ldots, n$$

$$\therefore J_2(i) \geq J_1(i)\ \forall i$$

In general:

$$J_{k+1}(i) \geq J_k(i)\ \forall i,\ \forall k$$
$$J_k \to J^*$$
$$\therefore J_0(i) \leq J^*(i)\ \forall i$$

Now let J solve the following problem:

$$\max \sum_i J(i) \text{ subject to}$$
$$J(i) \leq g(i,u) + \sum_j p_{ij}(u) J(j) \; \forall i, \forall u \in U(i)$$

Clear that $J(i) \leq J^*(i) \; \forall i$, by previous analysis. Since J^* satisfies the constraints, it follows that J^* achieves the maximum.
This is a Linear Program!

2.13.4 Analogies and Connections

Say I want to solve

$$J = G + PJ, \quad J \in \mathbb{R}^n, \; G \in \mathbb{R}^n, \; P \in \mathbb{R}^{n \times n}.$$

Direct way to solve:

$$\begin{aligned} (I - P)J &= G \\ J &= (I - P)^{-1} G. \end{aligned}$$

This is exactly what we do in Stage 1 of policy iteration: solve for the cost associated wit ha specific policy.
Why is $(I - P)^{-1}$ guaranteed to exist?
For a given policy, let $\bar{P} \in \mathbb{R}^{(n+1) \times (n+1)}$ be the probability matrix that captures our Markov Chain:

$$\bar{P} = \begin{pmatrix} P & P_{(\cdot)t} \\ 0 & 1 \end{pmatrix} \qquad p_{ij}: \text{ probability next state = } j \text{ given current state = } i.$$

Facts

- $\bar{P}$ is a right stochastic matrix: all rows sum up to 1, all elements ≥ 0.
- Perron-Frobenius Therorem: eigenvalues of $\bar{P}$ have absolute value ≤ 1, at least one $= 1$
- Assumption on terminal state: $P^N \to 0$ as $N \to \infty$. We well eventually reach termination state!

Therefore

- the eigenvalues of P have absolute value < 1.

- $(I-P)^{-1}$ exists.

Furthermore:

$$(I-P)^{-1} = I + P + P^2 + \dots$$

Proof:

$$\begin{aligned}(I-P)(I+P+P^2+\dots) &= I + (P+P^2+\dots) - (P+P^2+\dots) \\ &= I\end{aligned}$$

Therefore one way to solve for J is as follows:

$$\begin{aligned}J_1 &= G + PJ_0 \\ J_2 &= G + PJ_1 = G + PG + P^2 J_0 \\ &\vdots \\ J_N &= (I + P + \dots + P^{N-1})G + P^N J_0 \\ &\therefore J_N \to (1-P)^{-1}G \text{ as } N \to \infty!\end{aligned}$$

Analogy

Value Iteration: step of the update
Policy Iteration: infinite numbers of update, or solving system of equations exactly.

- What is truly amazing is that various combinations of policy iteration, value iteration, all converge to Bellman's Equation.

- Recall value iteration

$$J_{k+1}(i) = \min_{u \in U(i)} \left(g(i,u) + \sum_j p_{ij}(u) J_k(j) \right) \qquad i = 1, \dots, n$$

  In practice, you would implement as follows:

$$\begin{aligned}&\bar{J}(i) \leftarrow \min_{u \in U(i)} \left(g(i,u) + \sum_j p_{ij}(u) J(j) \right) \qquad i = 1, \dots, n \\ &J(i) \leftarrow \bar{J}(i) \qquad i = 1, \dots, n\end{aligned}$$

Don't have to do this! Can also do:

$$J(i) \leftarrow \min_{u \in U(i)} \left(g(i,u) + \sum_j p_{ij}(u) J(j) \right) \qquad i = 1, \ldots, n$$

Gauss-Seidel update: generic technique for solving iterative equations.

- Gets even better: *Asynchronous Policy Iterating*.
  - Any number of value update in between policy updates
  - Any number of states updated at each value update
  - Any number of states updated at each policy update.
    Under some mild assumptions, all converge to J^*

2.14 Discounted Problems

$$J_\pi(i) = \lim_{N \to \infty} \mathrm{E} \left(\sum_{k=0}^{N-1} \alpha^k g\left(x_k, \mu_k(x_k)\right) \middle| x_0 = i \right) \quad \alpha < 1,\ i \in \{1, \ldots, n\}$$

No explicit termination state required. No assumption on transition probabilities required.
Bellman's Equation for this problem:

$$J^*(i) = \min_{u \in U(i)} \left(g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u)\, J^*(j) \right) \quad \forall i$$

How do we show this?

- Define associated problem with states $\{1, 2, \ldots, n, t\}$. From state $i \neq t$, when u is applied in new cost $g(i,u)$ next state $= j$ with probability $\alpha p_{ij}(u)$, and t with probability $1 - \alpha$ (since $\sum_j p_{ij}(u) = 1$)
- Clear that since $a < 1$, we have a non-zero probability of making it to state t, therefore our assumption on reaching the termination state is satisfied.
  Suppose we use the same policy in discounted problem as auxiliary problem.
  Note that

$$\begin{aligned} P_R\left(x_{k+1} = j \middle| x_k = i, x_{k+1} \neq t, u\right) &= \frac{\alpha p_{ij}(u)}{\alpha p_{i,1} + \alpha p_{i,2} + \ldots + \alpha p_{i,n}} \\ &= \frac{\alpha p_{ij}(u)}{\alpha} = p_{ij}(u) \end{aligned}$$

So as long as we reach the termination state, the state evolution is governed by the same probabilities. The expected cost of the k$^{\text{th}}$ stage of associated problem $g(x_k, \mu_k(x_k))$ times the probability, that t has not been reached α^k, therefore have $\alpha^k g(x_k, \mu_k(x_k))$.

Connections: for a given policy, have

$$\bar{P} = \begin{pmatrix} \alpha P & \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} (1-\alpha) \\ 0 & 1 \end{pmatrix}$$

clear that $(\alpha P)^N = \alpha^N P^N \to 0$

Chapter 3

Continuous Time Optimal Control

Consider the following system

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \qquad 0 \le t \le T \\ x(0) &= x_0 \qquad \text{no noise!} \end{aligned}$$

- State $x(t) \in \mathcal{R}$
- Time $t \in \mathcal{R}, T$ is the terminal time
- Control $u(t) \in U \subset \mathcal{R}^m$, U control constraint set

Assume

- f is continuously differentiable with respect to x. (Less stringent requirement: Lipschitz)
- f is continuous with respect to u
- $u(t)$ is piecewise continuous

See Appendix A in[1] for Details.
Assume: Existence and uniqueness of solutions.

Example 8:

$$\begin{aligned} \dot{x}(t) &= x(t)^{1/3} \\ x(0) &= 0 \end{aligned}$$

Solutions: $x(t) = 0 \quad \forall t \qquad x(t) = (\frac{2}{3}t)^{3/2}$
Not uinque!

Example 9:

$$\dot{x}(t) = x(t)^2$$
$$x(0) = 1$$

Solutions: $x(t) = \frac{1}{1-t}$, finite escape time, $x(1) = \infty$
The solution does not exist on an interval that includes 1, e.g. $[0,2]$.

Objective Minimize $h(x(T)) + \int_0^T g(x(t), u(t))\, \mathrm{d}t$ where g and h are continuously differentiable with respect to x, g is continuous with respect to u. Very similar to discrete time problem ($\sum \to \int$, $x_{k+1} \to \dot{x}$) except for technical assumptions.

3.1 The Hamilton Jacobi Bellman (HJB) Equation

Continuous time analog of DP algorithm. Derive it informally by discretizing problem and taking limits. Not a rigorous derivation, but it does capture the main ideas.

- Divide time horizon into N pieces, define $\delta = T/N$
- $x_k := x(k\delta)$, $u_k := u(k\delta) \quad k = 0, 1, \ldots, N$
- Approximate differential equation by

$$\dot{x}(k\delta) = f(x(k\delta, u(k\delta)))$$
$$\frac{x_{k+1} - x_k}{\delta} = f(x_k, u_k),\ x_{k+1} = x_k + f(x_k, u_k)\,\delta$$

  Approximate cost function:

$$h(x_N) + \sum_{k=0}^{N-1} g(x_k, u_k)\,\delta$$

- Define $J^*(t, x)$ = optimal cost to go at time t and state x for the continuous problem.
- Define $\tilde{J}^*(t, x)$ = discrete approximation of optimal cost to go.
  Apply DP algorithm:
  - terminal condition: $\tilde{J}^*(N\delta, x) = h(x)$
  - recursion:

$$\tilde{J}^*(k\delta, x) = \min_{u \in U} \left[g(x,u)\,\delta + \tilde{J}^*((k+1)\,\delta, x + f(x,u)\,\delta) \right] \quad k = 0, \ldots, N-1$$

- Do Taylor Expansion of $\tilde{J}^*$, because $\delta \to 0$

$$\tilde{J}^*((k+1)\,\delta, x + f(x,u)\,\delta) = \tilde{J}^*(k\delta, x) + \frac{\partial \tilde{J}^*(k\delta, x)}{\partial t}\,\delta + \left(\frac{\partial J^*(k\delta, x)}{\partial x}\right)^T f(x,u)\,\delta + o(\delta)$$

$$\lim_{\delta \to 0} \frac{o(\delta)}{\delta} = 0$$

"little oh notation": $o(\delta)$ are quadratic terms or higher in δ.

- Substitute back into DP recursion by δ:

$$0 = \min_{u \in U} \left(g(x,u) + \frac{\partial \tilde{J}^*(k\delta, x)}{\partial t} + \left(\frac{\partial \tilde{J}^*(k\delta, x)}{\partial x}\right)^T f(x,u) + \frac{o(\delta)}{\delta} \right)$$

Now let $t = k\delta$, and let $\delta \to 0$. Assuming $\tilde{J}^* \to J$, have

$$0 = \min_{u \in U} \left(g(x,u) + \frac{\partial J^*(t,x)}{\partial t} + \left(\frac{\partial J^*(t,x)}{\partial x}\right)^T f(x,u) \right) \quad \forall x,\, t \tag{3.1}$$

$$J^*(T,x) = h(x)$$

HJB equation (3.1):

  - Partial Differential Equation, very difficult to solve
  - $u = \mu(t,x)$ that minimizes R.H.S. of HJB is optimal policy

Example 10: Consider the system

$$\dot{x}(t) = u(t) \quad |u(t)| \le 1$$

Cost is $\frac{1}{2}\,x^2(T)$, only terminal cost.

Intuitive solution: $u(t) = \mu(t,x) = -\operatorname{sgn}(x) = \begin{cases} -1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x < 0 \end{cases}$

What is cost to go associated with this policy?

$$V(t,x) = \frac{1}{2}\,(\max\{0, |x| - (T-t)\})^2$$

Verify that this is indeed the cost to go associated with the policy outlined above.
For fixed t:

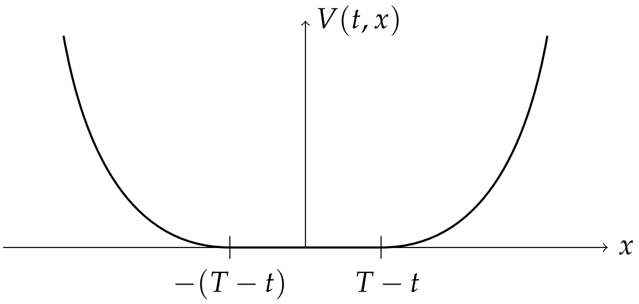


Figure 3.1: Example 10 for fixed t.

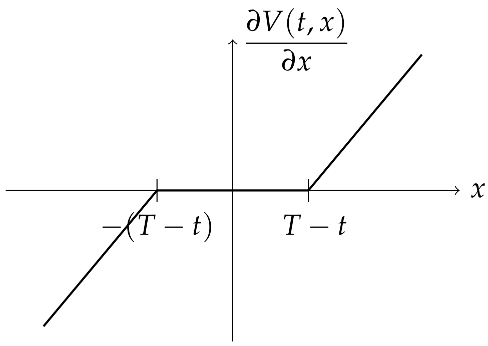


Figure 3.2: First derivative.

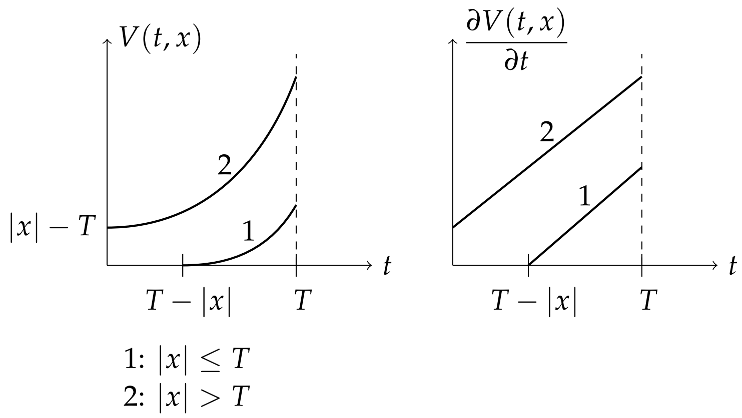


Figure 3.3: Example 10 for fixed x.

$$\frac{\partial V}{\partial x}(t,x) = \operatorname{sgn}(x)\ \max\{0, |x| - (T-t)\}$$

For fixed x:
Does $V(t,x)$ satisfy HJB?

- First check: does it satisfy boundary condition?
  $V(T,x) = \frac{1}{2}x^2 = h(x)$ ✓

- Second check:

$$\begin{aligned}\min_{|u|\leq 1}\left(\frac{\partial V(t,x)}{\partial t} + \frac{\partial V(t,x)}{\partial x} f(x,u)\right) &= \min_{|u|\leq 1}(1+\operatorname{sgn}(x),u)\,\max\{0,|x|-(T-t)\} \\ &= 0 \quad \text{by choosing } u = \operatorname{sgn}(x)\end{aligned}$$

  $\therefore V(t,x)$ satisfies HJB equation, $\therefore V(t,x) = J^*(t,x)$.

  Furthermore $u = -\operatorname{sgn}(x)$ is an optimal solution. Not unique!

  Note: Verifying that $V(t,x)$ satisfies HJB is not trivial, even for this simple example. Imagine solving for it!

Another issue: $\frac{1}{2}x^2(T)$ will give same optimal policy as cost $|x(T)|$, so different costs give same optimal policy, but some cost are "nicer" to work with than other.

3.2 Aside on Notation

Let $F(t,x)$ be a continuosly differentiable function. Then

1. $\dfrac{\partial F(t,x)}{\partial t}$: partial derivative of F with respect to the first argument.
2. $\dfrac{\partial F(t,\bar{x}(t))}{\partial t} = \left.\dfrac{\partial F(t,x(t))}{\partial t}\right|_{x=\bar{x}(t)}$: shorthand notation
3. $\dfrac{\mathrm{d}F(t,\bar{x}(t))}{\mathrm{d}t} = \dfrac{\partial F(t,\bar{x}(t))}{\partial t} + \dfrac{\partial F(t,\bar{x}(t))}{\partial x}\dot{\bar{x}}(t)$: total derivative

Example 11: For $F(t,x) = tx$

$$\frac{\partial F(t,x)}{\partial t} = x \qquad \frac{\partial F(t,\bar{x}(t))}{\partial t} = \bar{x}(t) \qquad \frac{\mathrm{d}F(t,\bar{x}(t))}{\mathrm{d}t} = \bar{x}(t) + t\dot{\bar{x}}(t)$$

Lemma 3.2.1: Let $F(t,x,u)$ be a continous differentiable function and let U be a convex set. Assume that $\mu^*(t,x) := \arg\min_{u\in U} F(t,x,u)$ is continous differentiable.

Then:

$$
\begin{aligned}
1) \quad & \frac{\partial \min_{u \in U} F(t,x,u)}{\partial t} = \frac{\partial F(t,x,\mu^*(t,x))}{\partial t} \ \forall t,x \\
2) \quad & \frac{\partial \min_{u \in U} F(t,x,u)}{\partial x} = \frac{\partial F(t,x,\mu^*(t,x))}{\partial x} \ \forall t,x
\end{aligned}
$$

Example 12: Let $F(t,x,u) = (1+t)u^2 + ux + 1$, $t \geq 0$, U is the real line (no constraint on u), then

$$
\begin{aligned}
\min_u F(t,x,u): \quad & 2(1+t)u + x = 0, \to u = -\frac{x}{2(1+t)} \\
\therefore \mu^*(t,x) &= -\frac{x}{2(1+t)}, \\
\min_u F(t,x,u) &= \frac{(1+t)x^2}{4(1+t)^2} - \frac{x^2}{2(1+t)} + 1 \\
&= -\frac{x^2}{4(1+t)} + 1
\end{aligned}
$$

$$
\begin{aligned}
1) \quad & \frac{\partial \min_{u \in U} F(t,x,u)}{\partial t} = \frac{x^2}{4(1+t)^2} \\
& \frac{\partial F(t,x,\mu^*(t,x))}{\partial t} = u^2|_{u=\mu^*(t,x)} = \frac{x^2}{4(1+t)^2} \\
2) \quad & \frac{\partial \min_{u \in U} F(t,x,u)}{\partial x} = \frac{x}{2(1+t)} \\
& \frac{\partial F(t,x,\mu^*(t,x))}{\partial x} = u|_{u=\mu^*(t,x)} = \frac{x}{2(1+t)}
\end{aligned}
$$

Proof 2: Proof of Lemma 3.2.1 when u is unconstrained ($U \in \mathbb{R}^m$). Let

$$G(t,x) = \min_{u \in U} F(t,x,u) = F(t,x,\mu^*(t,x)).$$

Then

$$\frac{\partial G(t,x)}{\partial t} = \frac{\partial F(t,x,\mu^*(t,x))}{\partial t} + \underbrace{\frac{\partial F(t,x,\mu^*(t,x))}{\partial u}}_{=0 \text{ because } \mu^*(t,x) \text{ minimizes}} \frac{\partial \mu^*(t,x)}{\partial t}.$$

□

This can be done similar for $\dfrac{\partial G(t,x)}{\partial x}$.

3.2.1 The Minimum Principle

HJB gives us a *lot of information*: optimal cost to go for all time and for all possible states, also gives optimal feedback law $u = \mu^*(t, x)$. What if we only cared about optimal control trajectory for a specific initial condition $x(0) = x_0$?
Can we exploit the fact that we are asking for much less to simplify the mathematical conditions?
Starting Point: HJB

$$\begin{aligned} 0 &= \min_{u \in U} \left(g(x,u) + \frac{\partial J^*(t,x)}{\partial t} + \left(\frac{\partial J^*(t,x)}{\partial x} \right)^T f(x,u) \right) && \forall t, x \\ J^*(T,x) &= h(x) && \forall x \end{aligned}$$

- Let $\mu^*(t, x)$ be te corresponding optimal strategy (feedback law)
- Let
$$F(t,x,u) = g(x,u) + \frac{\partial J^*(t,x)}{\partial t} + \left(\frac{\partial J^*(t,x)}{\partial x} \right)^T f(x,u)$$

So HJB equation gives us

$$G(t,x) = \min_{u \in U} \left(F(t,x,u) \right) = 0$$

Apply Lemma

$$\begin{aligned} 1) \quad & \frac{\partial G(t,x)}{\partial t} = 0 = \frac{\partial^2 J^*(t,x)}{\partial t^2} + \left(\frac{\partial^2 J^*(t,x)}{\partial x \partial t} \right)^T f(x, \mu^*(t,x)) && \forall t, x \\ 2) \quad & \frac{\partial G(t,x)}{\partial x} = 0 = \frac{\partial g(x, \mu^*(t,x))}{\partial x} + \frac{\partial^2 J^*(t,x)}{\partial x \partial t} + \frac{\partial^2 J^*(t,x)}{\partial x^2} f(x, \mu^*(t,x)) \\ & \qquad + \frac{\partial f(x, \mu^*(t,x))}{\partial x} \frac{\partial J^*(t,x)}{\partial x} && \forall t, x \end{aligned}$$

Consider a **specific** optimal trajectory:

$$u^*(t) = \mu^*\left(t, x^*(t)\right), \qquad \dot{x}^*(t) = f\left(x^*(t), u^*(t)\right), \qquad x^*(0) = x_0$$

$$\begin{aligned} 1) \quad & 0 = \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial J^*(t, x^*(t))}{\partial t} \right) \\ 2) \quad & 0 = \frac{\partial g\left(x^*(t), u^*(t)\right)}{\partial x} + \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial J^*(t, x^*(t))}{\partial x} \right) + \frac{\partial f(x^*(t), u^*(t))}{\partial x} \frac{\partial J^*(t, x^*(t))}{\partial x} \end{aligned}$$

$$p(t) = \frac{\partial J^*(t, x^*(t))}{\partial x} \qquad\qquad p_0(t) = \frac{\partial J^*(t, x^*(t))}{\partial t}$$

1) $\qquad \dot{p}_0(t) = 0 \rightarrow p_0(t) = \text{ constant for } 0 \leq t \leq T$

2) $\qquad \dot{p}(t) = -\frac{\partial f(x^*(t), u^*(t))}{\partial x} p(t) - \frac{\partial g(x^*(t), u^*(t))}{\partial x}, \; 0 \leq t \leq T$

$$\frac{\partial J^*(T, x)}{\partial x} = \frac{\partial h(x)}{\partial x} \rightarrow p(T) = \frac{\partial h(x^*(T))}{\partial x}$$

Put all of this together:
Define Hamiltonian $H(x, u, p) = g(x, u) + p^T f(x, u)$. Let $u^*(t)$ be an optimal control trajectory, $x^*(t)$ resulting state trajectory. Then

$$\begin{aligned}
\dot{x}^*(t) &= \frac{\partial H}{\partial p}(x^*(t), u^*(t), p(t)) && x^*(0) = x_0 \\
\dot{p}(t) &= -\frac{\partial H}{\partial x}(x^*(t), u^*(t), p(t)) && p(T) = \frac{\partial h(x^*(T))}{\partial x} \\
u^*(t) &= \arg\min_{u \in U} H(x^*(t), u, p(t)) \\
H(x^*(t), u^*(t), p(t)) &= \text{ constant} && \forall t \in [0, T]
\end{aligned}$$

($H(\cdot) =$ constant comes from $p_0(t) =$ constant).
Some remarks:

- Set of $2n$ ODE, with **split** boundary conditions. Not trivial to solve.
- Necessary condition, but not sufficient. Can have multiple solutions, not all of them may be optimal.
- If $f(x, u)$ is linear, U is convex, h and g are convex, then the condition is **necessary and sufficient**.

Example 13 (Resource Allocation): Some robots are sent to Mars to build habitats for a later exploration by humans.

$x(t)$ Number of reconfigurable robots, which can build habitats or themselves.

$x(0)$ is given: Number of robots that arrive to Mars.

$$\begin{aligned}
\dot{x}(t) &= u(t)x(t) && x(0) = x_0 \\
\dot{y}(t) &= (1 - u(t))x(t) && y(0) = 0 \\
0 &\leq u(t) \leq 1
\end{aligned}$$

Objective: given terminal time T, find control input $u^*(t)$ that **maximizes** $y(T)$, the number of habitats build.
Note: $y(t) = \int_0^T (1-u(t))x(t)\,\mathrm{d}t$

Solution

$$\begin{aligned}
g(x,u) &= (1-u)x \\
f(x,u) &= ux \\
H(x,u,p) &= (1-u)x + pux \\
\dot{p}(t) &= -\frac{\partial H(x^*(t),u^*(t),p(t))}{\partial x} = -1 + u^*(t) - p(t)u^*(t) \\
p(T) &= 0 \qquad (h(x) \equiv 0) \\
u^*(t) &= \arg\max_{0\le u\le 1} \left(x^*(t) + \left(p(t)x^*(t) - x^*(t)\right)u\right) \\
&\text{get} \begin{cases} u = 0 & \text{if } p(t) < 1 \\ u = 1 & \text{if } p(t) > 1 \end{cases}
\end{aligned}$$

Since $p(T) = 0$, for t close to T, will have $u^*(t) = 0$ and therefore $\dot{p}(t) = -1$.
Therefore at time $t = T-1$, $p(t) = 1$ and that is where switch occurs:

$$\begin{aligned}
\dot{p}(t) &= -p(t) & 0 \le t \le T-1 \qquad & p(T-1) = 1 \\
\therefore p(t) &= \exp(-t)\exp(T-1) & 0 \le t \le T-1 \qquad &
\end{aligned}$$

Conclusion

$$u^*(t) = \begin{cases} 1 & 0 \le t \le T-1 \\ 0 & T-1 \le t \le T \end{cases}$$

How to use in practice:

1. If you can solve HJB, you get a feedback law $u = \mu(x)$. Very convenient, just a controller: meausure the state and apply the control input.

2. Solve for optimal trajectory and use a feedback law (probably linear) to keep you on that trajectory.

3. Solve for optimal trajectory online after measuring state. Do this often.

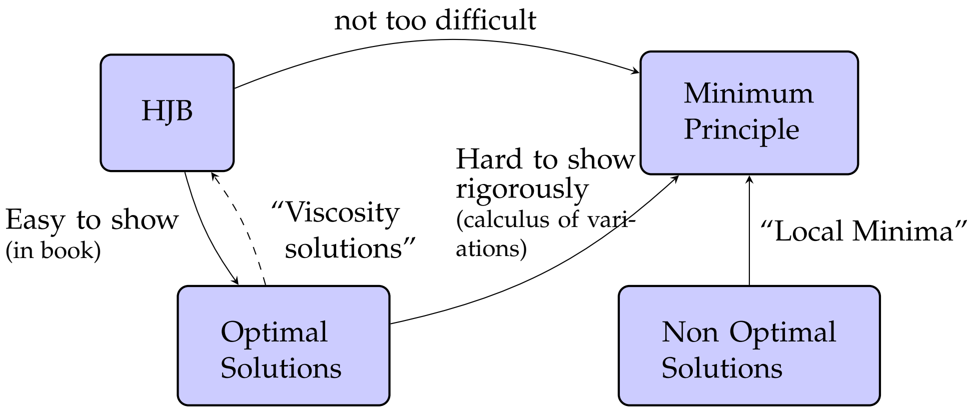


Figure 3.4: Different approaches to find a solution.

3.3 Extensions

(Drop $x^*(t)$, J^* notation for simplycity)

3.3.1 Fixed Terminal State

Case where $x(T)$ is given. Clear that there is no need for a terminal cost. Recall co-state $p(t) = \dfrac{\partial J(t, x(t))}{\partial x}$.
$p(T) = \lim_{t \to T} \dfrac{\partial J(t, x(t))}{\partial x}$, but we can't use $h(t)$, the terminal cost, to constrain $p(T)$. Don't need constraints on p!

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \qquad x(0) = x_0, \quad x(T) = x_T & \qquad 2n \text{ ODEs} \\ \dot{p}(t) &= -\frac{\partial H(x(t), u(t), p(t))}{\partial x} & \qquad 2n \text{ boundary conditions} \end{aligned}$$

Example 14:

$$\begin{aligned} \dot{x}(t) &= u(t) \quad x(0) = 0, \quad x(1) = 1 \\ g(x, u) &= \frac{1}{2}(x^2 + u^2) \\ \text{cost} &= \frac{1}{2}\int_0^1 (x^2(t) + u^2(t))\, \mathrm{d}t \end{aligned}$$

Hamiltonian $H(x, u, p) = \frac{1}{2}(x^2 + u^2) + p\,u$
We get

$$\begin{aligned}
\dot{x}(t) &= u(t) \\
\dot{p}(t) &= -x(t) \\
u(t) &= \arg\min_u \frac{1}{2}(x^2(t) + u^2(t)) + p(t)\,u(t)
\end{aligned}$$

therefore $u(t) = -p(t)$

$$\therefore \dot{x}(t) = -p(t), \quad \dot{p}(t) = -x(t), \quad \ddot{x}(t) = x(t)$$

$$x(t) = A\cosh(t) + B\sinh(t)$$

$$x(0) = 0 \Rightarrow A = 0, \quad x(1) = 1 \Rightarrow B = \frac{1}{\sinh(1)}$$

$$x(t) = \frac{\sinh(t)}{\sinh(1)} = \frac{e^t - e^{-t}}{e^1 - e^{-1}}$$

Exercise: show that Hamiltonian is constant along this trajectory.

3.3.2 Free initial state, with cost

$x(0)$ is *not* fixed, but have *initial* cost $l(x(0))$. Can show that the resuling condition is $p(0) = -\dfrac{\partial l(x(0))}{\partial x}$.

Example 15:

$$\begin{aligned}
\dot{x}(t) &= u(t) \qquad x(1) = 1, \quad x(0) \text{ is free} \\
g(x, u) &= \frac{1}{2}(x^2 + u^2) \\
l(x) &= 0 \qquad \text{no cost (given)}
\end{aligned}$$

Apply Minimum Principle, as before

$$\begin{aligned}
\dot{x}(t) &= u(t) & \ddot{x} &= x(t) \\
\dot{p}(t) &= -x(t) & & \\
u(t) &= -p(t) & x(t) &= A\cosh(t) + B\sinh(t)
\end{aligned}$$

$$\dot{x}(0) = u(0) = -p(0) = 0 \quad \therefore B = 0$$

$$x(t) = \frac{\cosh(t)}{\cosh(1)} = \frac{e^t + e^{-t}}{e^1 + e^{-1}}$$

$$x(0) \approx 0.65$$

Free Terminal Time Result: Hamiltonian $= 0$ on optimal trajectory.
Gain extra degree of freedom in choosing T, we lose a degree of freedom because $H \equiv 0$.

Time Varying System and Cost What happens if $f = f(x, u, t)$, $g = g(x, u, t)$?
Result: Everything stays the same, except that Hamiltonian is no longer constant along trajectory. Hint: $\dot{t} = 1$ and $\dot{x} = f(x, u, t)$

Singular Problems Motivate via an example:
Tracking problem: $z(t) = 1 - t^2 \quad 0 \le t \le 1$
Minimize $\frac{1}{2}\int_0^1 (x(t) - z(t))^2 \, dt$ subject to $|\dot{x}(t)| \le 1$.
Apply Minimum Principle:

$$\begin{aligned} \dot{x}(t) &= u(t) \qquad |u(t)| \le 1 \quad x(0),\, x(1) \text{ are free} \\ g(x, u, t) &= \frac{1}{2}(x - z(t))^2 \\ H(x, u, t) &= \frac{1}{2}(x - z(t))^2 + p\,u \end{aligned}$$

Co-state equation:

$$\dot{p}(t) = -(x(t) - z(t)) \qquad p(0) = 0,\; p(1) = 0$$

Optimal u:

$$u(t) = \arg\min_{|u| \le 1} H(x(t), u, p(t), t)$$

$$u(t) = \begin{cases} = -1 & \text{if } p(t) > 0 \\ = 1 & \text{if } p(t) < 0 \\ = ? & \text{if } p(t) = 0 \end{cases}$$

Problem is singular if Hamiltonian is not a function of u for a non-trivial time interval.
Try the following:

$$p(0) = 0 \quad \dot{p}(t) = 0 \qquad \text{for } 0 \le t \le T \quad T \text{ to be determined}$$

Then

$$\begin{aligned} p(t) &= 0 & 0 \le t \le T \\ \therefore x(t) &= z(t) & 0 \le t \le T \\ \therefore \dot{x}(t) &= \dot{z}(t) = -2t = u(t) \end{aligned}$$

One guess: pick $T = \frac{1}{2}$.
This can't be the solution: for $t > \frac{1}{2}$:

$$x(t) - z(t) > 0 \quad \therefore \dot{p} < 0 \quad \therefore p(1) < 0$$

can't satisfy constraint.
Explore this instead:
Switch before $T = \frac{1}{2}$.

$$\begin{aligned}
x(t) &= z(t) && 0 \leq t \leq T < \frac{1}{2} \\
\dot{x}(t) &= -1 && T < t \leq 1 \\
\therefore x(t) &= z(T) - (t - T) = 1 - T^2 - t + T \\
\dot{p}(t) &= -(x(t) - z(t)) - (1 - T^2 - t + T - 1 + t^2) && T < t \leq 1 \\
&= T^2 - T - t^2 + t \\
p(1) &= \int_T^1 (T^2 - T - t^2 + t)\,\mathrm{d}t \\
&= T^2 - T - \frac{1}{3} + \frac{1}{2} - T^3 + T^2 + \frac{T^3}{3} - \frac{T^2}{2} = 0
\end{aligned}$$

Simplifies to (multiply by 6):

$$\begin{aligned}
0 &= -4T^3 + 9T^2 - 6T + 1 \\
&= (T-1)(T-1)(1-4T) \\
\therefore T &= 1, T = \frac{1}{4}
\end{aligned}$$

$T = \frac{1}{4}$ satisfies all the constraints and we are done!
Can easily verify that $p(t) > 0$ for $\frac{1}{4} < t < 1$, giving $u(t) = -1$ as required.

3.4 Linear Systems and Quadratic Costs

Look at infite horizon, LTI (linear time invariant) system:

$$\begin{aligned}
x_{k+1} &= Ax_k + Bu_k && k = 0, 1, \ldots \\
\text{cost } &= \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k, && R > 0,\ Q \geq 0, R = R^T, Q = Q^T
\end{aligned}$$

Informally, the cost to go is time invariant: only depends on the state and **not** when we get there.

$$J(x) = \min_u \left(x^T Q x + u^T R u + J(Ax + Bu) \right)$$

Conjecture that optimal cost to go is quadratic in x: $J(x) = x^T K x$, where $K = K^T, K \geq 0$. Then

$$x^T K x = x^T Q x + x^T A^T K A x + \min_u \left(u^T R u + u^T B^T K B u + x^T A^T K B u + u^T B^T K A x \right)$$

Since $R > 0$, $B^T K B \geq 0$, $R + B^T K B > 0$

$$2\left(R + B^T K B\right) u + 2 B^T K A x = 0$$

$$\rightarrow u = -\left(R + B^T K B\right)^{-1} B^T K A$$

Substitute back in:
All terms are of the form $x^T \, (\,\cdot\,)\, x$.
Therefore we must have:

$$K = Q + A^T K A + A^T K B \left(R + B^T K B\right)^{-1} \left(R + B^T K B\right) \left(R + B^T K B\right)^{-1} B^T K A$$
$$- 2 A^T K B \left(R + B^T K B\right)^{-1} B^T K A$$

$$K = A^T \left(K - K B \left(R + B^T K B\right)^{-1} B^T K \right) A + Q, \quad K \geq 0$$

Summary

- Optimal Cost to go $J(x) = x^T K x$
- Optimal feedback strategy $u = Fx,\ F = -\left(R + B^T K B\right)^{-1} B^T K A$

Questions

1. Can we always solve for K?
2. Is closed loop system $x_{k+1} = (A + BF) x_k$ stable?

Example 16:

$$x_{k+1} = 2 x_k + 0 \cdot u_k$$
$$\text{cost } = \sum_{k=0}^{\infty} \left(x_k^2 + u_k^2 \right)$$

$$A = 2, \qquad B = 0, \qquad Q = 1, \qquad R = 1$$

Solve for K:

$$K = 4\,(K - 0) + 1$$
$$\Leftrightarrow -3K = 1$$
$$\rightarrow K = -1/3$$

K **does not** satisfy $K \geq 0$ constraint. No solution to this problem: cost is infite.
Problem with this example is that (A, B) is not stabilizable.

Stabilizable : One can find a matrix F such that $A + BF$ is stable.
$\rho\left(A+BF\right) < 1$, eigenvalues of $A + BF$ have magnitude < 1.

Example 17:

$$\begin{aligned} x_{k+1} &= 0.5x_k + 0 \cdot u_k \\ \text{cost } &= \sum_{k=0}^{\infty}\left(x_k^2 + u_k^2\right) \end{aligned}$$

$$A = 0.5, \qquad B = 0, \qquad Q = 1, \qquad R = 1$$

Solve for K:

$$\begin{aligned} K &= 0.25K + 1 \\ \rightarrow K &= 4/3 \end{aligned}$$

Cost to go = $\frac{4}{3}x_k^2$. $F = 0$.

Example 18:

$$\begin{aligned} x_{k+1} &= 2x_k + u_k \\ \text{cost } &= \sum_{k=0}^{\infty}\left(x_k^2 + u_k^2\right) \end{aligned}$$

$$A = 2, \qquad B = 1, \qquad Q = 1, \qquad R = 1$$

(A, B) is stabilizable. Solve for K:

$$\begin{aligned} K &= 4\left(K - K^2(1+K)\right) + 1 \\ 0 &= K^2 - 4K - 1 \\ \rightarrow K &= \frac{4 \pm \sqrt{20}}{2} = 2 \pm \sqrt{5} \end{aligned}$$

Pick $K = 2 + \sqrt{5} \approx 4.236$ and solve for F:

$$F = -(1+K)^{-1}2K = \frac{-2K}{1+K} \approx -1.618$$

$A + BF = 2 - 1.618 \approx 0.3820$ is stable.
Our optimizing strategy stabilizes system as expected.

Example 19:

$$\begin{aligned} x_{k+1} &= 2x_k + u_k \\ \text{cost } &= \sum_{k=0}^{\infty} \left(u_k^2\right) \end{aligned}$$

$$A = 2, \qquad B = 1, \qquad Q = 0, \qquad R = 1$$

Solve for K:

$$\begin{aligned} K &= 4\left(K - K^2(1+K)^{-1}\right) \\ \rightarrow K &= \{0, 3\} \end{aligned}$$

$K = 0$ is clearly optimal thing to do, but it leads to an unstable system. $K = 3$, however, while not being optimal, leads to $F = -(1+3)^{-1}3 \cdot 2 = -1.5$, $A + BF = 0.5$, stable.
Modify cost to $\sum_{k=0}^{\infty}\left(u_k^2 + \varepsilon x_k^2\right)$, $\varepsilon > 0$, $\varepsilon \ll 1$.

$$A = 2, \qquad B = 1, \qquad Q = \varepsilon, \qquad R = 1$$

Solve for K:

$$K(\varepsilon) = \{3 + \frac{4\varepsilon}{3}, -\frac{\varepsilon}{3}\} \quad \text{(to first order in } \varepsilon\text{)}$$

The $K = 3$ solution is the limiting case as we put arbitrarily small cost to the state which would otherwise $\rightarrow \infty$.

Example 20:

$$\begin{aligned} x_{k+1} &= 0.5x_k + u_k \\ \text{cost } &= \sum_{k=0}^{\infty} \left(u_k^2\right) \end{aligned}$$

$$A = 0.5, \qquad B = 1, \qquad Q = 0, \qquad R = 1$$

Solve for K:

$$K = \{0, -0.75\}$$

Here $K = 0$ makes perfect sense: gives optimal strategy $u = 0$ *and* stable closed loop system.
If $Q = \varepsilon$

$$K(\varepsilon) \rightarrow \{\frac{4\varepsilon}{3}, -0.75 - \frac{\varepsilon}{3}\}$$

$K = 0$ is a well behaved solution for $\varepsilon = 0$.

Need Concept of Detectability Let Q be decomposed as $Q = C^T C$ (can always do this).
Need *detectability assumption*. (A, C) is detectable if $\exists L$ so that $A + LC$ is stable. $C\,x_k \to 0 \Rightarrow x_k \to 0$ and $C\,x_k \to 0 \Leftrightarrow x_k^T Q x_k \to 0$.

3.4.1 Summary

Given

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k && k = 0, 1, \dots \\ \text{cost } &= \sum_{k=0}^{\infty} \left(x^T Q x + u_k^T R u_k \right) && Q \geq 0,\ R \geq 0 \end{aligned}$$

(A, B) is stabilizable, (A, C) is detectable where C is any matrix that satisfies $C^T C = Q$. Then:

1. $\exists$ *unique* solution to D.A.R.E
2. Optimal cost to go is $J(x) = x^T K x$
3. Optimal feedback strategy $u = Fx$
4. Closed loop system is stable.

3.5 General Problem Formulation

- Finite horizon, time varying, disturbances.

$$\begin{aligned} x_{k+1} &= A_k\, x_k + B_k\, u_k + w_k && k = 0, \dots, N-1 \\ \mathrm{E}(w_k) &= 0 && \mathrm{E}(w_k\, w_k^T) \text{ finite} \end{aligned}$$

- Cost:

$$E\left(x_N^T\, Q_N\, x_N + \sum_{k=0}^{N-1} x_k^T\, Q_k\, x_k + u_k^T\, R_k\, u_k \right)$$

$$\begin{aligned} Q_k &= Q_k^T \geq 0 \qquad (\text{EV} \geq 0) \\ R_k &= R_k^T > 0 \end{aligned}$$

Apply DP to solve problem:

$$\begin{aligned} J_N(x_N) &= x_N^T\, Q_N\, x_N \\ J_k(x_k) &= \min_{u_k} \mathrm{E}\left(x_k^T\, Q_k\, x_k + u_k^T\, R_k\, u_k + J_{k+1}(A_k\, x_k + B_k\, u_k + w_k)\right) \end{aligned}$$

Let's do the first step of this recursion. Or equivalently, $N = 1$:

$$J_0(x_0) = \min_{u_0} \mathrm{E}\left(x_0^T\, Q_0\, x_0 + u_0^T\, R_0\, u_0 + (A_0\, x_0 + B_0\, u_0 + w_0)^T\, Q_1\, (\ldots)\right)$$

consider last term:

$$\begin{aligned} &\mathrm{E}\left((A_0\, x_0 + B_0\, u_0)^T\, Q_1\, (A_0\, x_0 + B_0\, u_0) + 2(A_0\, x_0 + B_0\, u_0)^T\, Q_1\, w_0 + w_0^T\, Q_1\, w_0\right) \\ &= (A_0\, x_0 + B_0\, u_0)^T\, Q_1(\ldots) + \mathrm{E}(w_0^T\, Q_1\, w_0) \end{aligned}$$

$$\therefore J_0(x_0) = \min_{u_0}\left(x_0^T\, Q_0\, x_0 + u_0^T\, R_0\, u_0 + (A_0\, x_0 + B_0\, u_0)^T\, Q_1\, (\ldots) + \mathrm{E}(w_0^T\, Q_1\, w_0)\right)$$

Strategy is the same as if there was no noise (although it does give a different cost). *Certainty equivalence* (works only for some problems, not all).
Solve for minimizing u_0: differentiate , set to 0:

$$\begin{aligned} 2R_0\, u_0 + 2\, B_0^T\, Q_1\, B_0\, u_0 + 2\, B_0^T\, Q_1\, A_0\, x_0 &= 0 \\ u_0 &= -(R_0 + B_0^T\, Q_1\, B_0)^{-1}\, B_0^T\, Q_1\, A_0\, x_0 \\ &=: F_0\, x_0 \end{aligned}$$

Optimal feedback strategy is a linear function of the state.
Substitute and solve for $J_0(x_0)$:

$$\begin{aligned} &x_0^T\, Q_0\, x_0 + u_0^T\, (R_0 + B_0^T\, Q_1\, B_0)\, u_0 + x_0^T\, A_0^T\, Q_1\, A_0\, x_0 + 2\, x_0^T\, A_0^T\, Q_1\, B_0\, u_0 + \mathrm{E}(w_0^T\, Q_1\, w_0) \\ &= x_0^T\, K_0\, x_0 + \mathrm{E}(w_0^T\, Q_1\, w_0) \\ K_0 &= Q_0 + A_0^T(K_1 - K_1\, B_0(R_0 + B_0^T\, K_1\, B_0)^{-1}\, B_0^T\, K_1)\, A_0 \\ K_1 &= Q_1 \end{aligned}$$

Cost at $k = 1$: quadratic in x_1, at $k = 0$: quadratic in x_0 + constant.
Can extend this to any horizon, *Discrete Riccati Equation (DRE)*:

$$\begin{aligned} K_k &= Q_k + A_k^T(K_{k+1} - K_{k+1}\, B_k\, (R_k + B_k^T\, K_{k+1}\, B_k)^{-1}\, B_k\, K_{k+1})\, A_k \\ K_N &= Q_N \end{aligned}$$

Feedback law:

$$u_k = F_k\, x_k \qquad F_k = -(R_k + B_k^T\, K_{k+1}\, B_k)^{-1}\, B_k^T\, K_{k+1}\, A_k$$

Cost:

$$J_k(x_k) = x_k^T K_k x_k + \sum_{j=k}^{N-1} \mathrm{E}(w_j^T K_{j+1} w_j)$$

No noise, time invariant, infinite horizon ($N \to \infty$): recover previous results and DARE. In fact, above iterative method is one way to solve DARE. Iterate backwards until it converges. ([1] has proof of convergence *not* trivial)

Time invariant, infinite horizon *with* noise: Cost goes to infinity. Approach: divide cost by N and let $N \to \infty$, cost $= \mathrm{E}(w^T K w)$.

Example 21: System given:

$$\ddot{z}(t) = u(t)$$

Objective Apply a force to move a mass from any starting point to $z = 0, \dot{z} = 0$. Implement on a computer that can only update information once per second.

1. Discretize problem:

$$\begin{aligned} \dot{z}(t) &= \dot{z}(0) + u(0)t && 0 \le t < 1 \\ z(t) &= z(0) + \dot{z}(0)t + 1/2\,u(0)t^2 && 0 \le t < 1 \end{aligned}$$

   Let $x_1(k) = z(k)$, $x_2(k) = \dot{z}(k)$.

$$x(k+1) = Ax(k) + Bu(k) \qquad k = 0, 1, \ldots$$

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \qquad B = \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}$$

2. Cost $= \sum_{k=0}^{\infty} \left(x_1^2(k) + u^2(k) \right)$. Therefore

$$Q = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \qquad R = 1$$

3. Is system stabilizable? Can we make $A + BF$ stable for some F?
   Yes: $F = \begin{pmatrix} -1 & -1.5 \end{pmatrix}$ makes eigenvalues of $A + BF = 0$

4. Q can be decomposed as follows

$$Q = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \end{pmatrix}$$

Therefore

$$C = \begin{pmatrix} 1 & 0 \end{pmatrix}, \qquad Q = C^T C$$

Is (A, C) detectable?
Yes.
$L = \begin{pmatrix} -2 & -1 \end{pmatrix}$ makes eigenvalues of $A + LC = 0$.

5. Solve *DARE*. Use `MATLAB` command `dare`.

$$K = \begin{pmatrix} 2 & 1 \\ 1 & 1.5 \end{pmatrix}$$

Optimal Feedback matrix:

$$F = \begin{pmatrix} -0.5 & -1.0 \end{pmatrix}$$

6. "Physical" interpretation: spring and a damper. spring has coefficient 0.5, damper has coefficient 1.0.

Bibliography

[1] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume I. Athena Scientific, 3rd edition, 2005.