Using the Foundations and Trends® LaTeX Class

Instructions for Creating an FnT Article

Suggested Citation: Alet Heezemans and Mike Casey (2018), "Using the Foundations and Trends® LaTeX Class", : Vol. xx, No. xx, pp 1–18. DOI: 10.1561/XXXXXXXXX.

Alet Heezemans
now publishers, Inc.
alet.heezemans@nowpublishers.com

Mike Casey
now publishers, Inc.
mike.casey@nowpublishers.com

now
the essence of knowledge
Boston — Delft

Contents

Using the Foundations and Trends® LaTeX Class

Alet Heezemans[1] and Mike Casey[2]

[1]*now publishers, Inc.; alet.heezemans@nowpublishers.com*
[2]*now publishers, Inc.; mike.casey@nowpublishers.com*

ABSTRACT

This document describes how to prepare a Foundations and Trends® article in LaTeX . The accompanying LaTeX source file FnTarticle.tex (that produces this output) is an example of such a file.

Alet Heezemans and Mike Casey (2018), "Using the Foundations and Trends® LaTeX Class", : Vol. xx, No. xx, pp 1–18. DOI: 10.1561/XXXXXXXXX.

1

Introduction to Dynamic Programming

1.1 Dynamic Programming Setting

Consider a *generic* optimization problem

$$\min_{u \in U} g(u)$$

where u is the decision variable, $g(u)$ is the cost function, and U is the constraint set. There are several categories of problems:

- Discrete (i.e., U is finite), or continuous
- Linear (i.e., g is lienar and U is polyhedral) or nonlinear
- Stochastic or deterministic: For stochastic problems, the cost involves a stochastic parameter ω, which is averaged:

  $$g(u) = \mathbb{E}_{\omega}\{G(u, w)\}$$

  where ω is a random parameter.

The dynamic programming (dp) is able to deal with complex stochastic problems where ω becomes available in stages, and the decisions are also made in stages based on the information.

1.1.1 Basic Structure of Stochastic DP

The DP usually invoves the following elements:

- Discrete-time system:
$$x_{k+1} = f_k(x_k, u_k, \omega_k), \quad k = 0, 1, \ldots, N-1$$
  where
  - k denotes the stage / discrete time
  - x_k denotes the state for stage k, which *summarizes past information* that is relevant for future decision
  - u_k denotes the control variable, i.e., a decision to be selected at time k from a given set. When making the decision, the state x_k is known.
  - w_k denotes the *random parameter / disturbance / noise* at stage k.
  - N is the *Horizon* or the number of times that the control is applied.
- A cost function that is *addictive* over time:
$$\mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, \omega_k) \right\}$$

Remark 1.1. We may use the transition matrix as the alternative *discrete-system* description, i.e., suppose the transition matrix $P(x_{k+1} \mid x_k, u_k)$ is available, then
$$x_{k+1} = \omega_k, \text{ with } P(\omega_k \mid x_k, u_k) = P(x_{k+1} \mid x_k, u_k)$$

Several assumptions are made for common dp problem:

- The set of possible values for control u_k depend at most on x_k but not on *prior* x or u
- The probability distribution of ω_k does not depend on past values $\omega_{k-1}, \ldots, \omega_0$, i.e., $\omega_k \sim P(\cdot; x_k, u_k)$, but may depend on x_k and u_k, since otherwise past values of ω would be useful for future optimization by applying *state augmentation.*

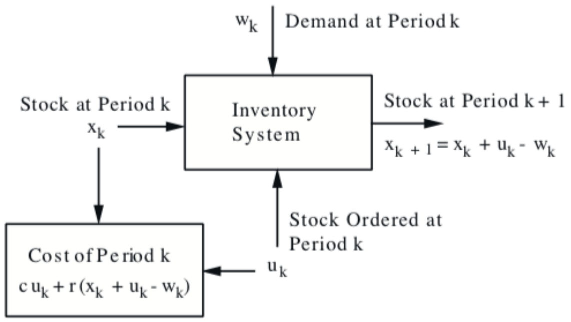

Figure 1.1: Inventory Model

1.1.2 Inventory Control Example

The inventory model is summarized in the figure below: In this case, the discrete-time system is

$$x_{k+1} = f_k(x_k, u_k, \omega_k) = x_k + u_k - \omega_k$$

with the cost function that is*additive* over time:

$$\mathbb{E}\left\{g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, \omega_k)\right\} = \mathbb{E}\left\{\sum_{k=0}^{N-1}(c \cdot u_k + r \cdot (x_k + u_k - \omega_k))\right\}$$

Our goal is to optimize the cost function over *policies*, and the policies $u_k = \mu_k(x_k)$ maps states into controls. The discrete-time system for this problem is deterministic, but the cost function is a expected value for a stochastic rv.

1.1.3 Deterministic Finite-State Problem

Consider a deterministic dp problem. We aim to find the optimal sequence of four operations A, B, C, D, where A must *precede* B and C must *precede* D. Assume the startup cost to be S_A, S_C, and the setup transition cost from operation m to operation n is C_{mn}.

We may list all possible scheduling in the figure below:

The discrete-time system for this problem is deterministic, and the cost function is deterministic as well.

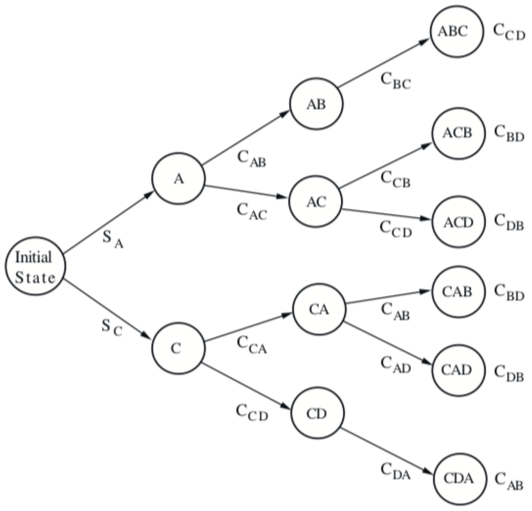

Figure 1.2: Scheduling Problem

1.1.4 Stochastic Finite-State Problem

The goal is to find the two-game chess match strategy.

- The *timid* play draws with probability $p_d > 0$ and loses with probability $1 - p_d$
- The *bold* play wins with probability $p_w < \frac{1}{2}$ and loses with probability $1 - p_w$.

We may list all possible strategies result in the figure (1.3). The discrete-time system for this problem is stochastic, and the cost function is stochastic as well.

1.1.5 Formal Setting of DP

The formal setting of DP is as follows:

- The system transition satisfies

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

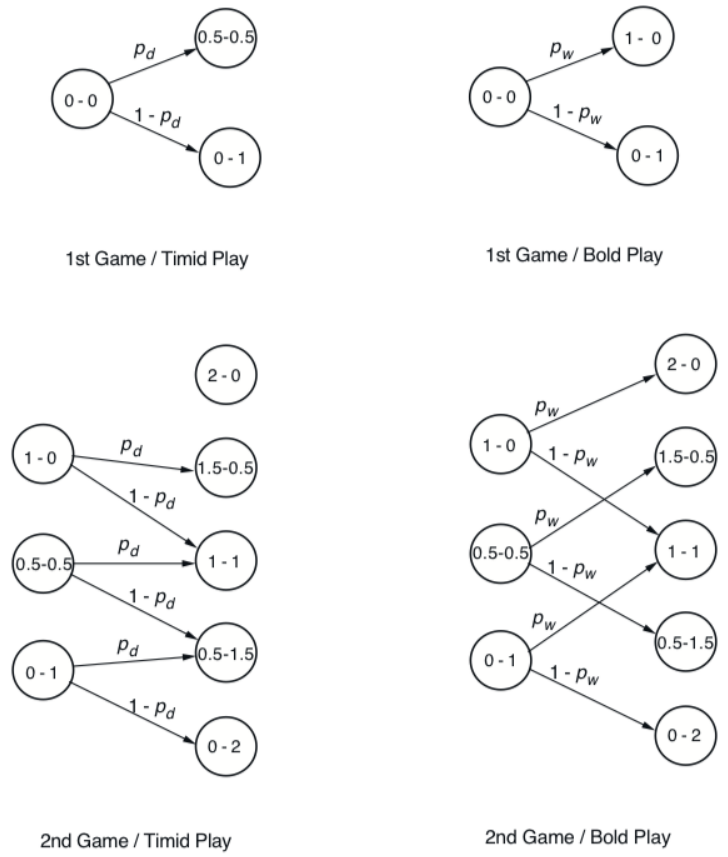

Figure 1.3: Two-Game Chess Match Strategies

- The control constraints only depends on x_k, i.e., $u_k \in U_k(x_k)$
- The probability distribution of ω_k is $P_k(\cdot \mid x_k, u_k)$
- The policy is a collection of functions $\pi = \{\mu_0, \dots, \mu_{N-1}\}$, where μ_k maps states x_k into controls $u_k = \mu_k(x_k) \in U_k(x_k)$ for all x_k
- The expected cost of π starting at x_0 is given by
$$J_\pi(x_0) = \mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), \omega_k) \right\}$$
The goal is to minimize the cost function:
$$J^*(x_0) = \min_\pi J_\pi(x_0),$$
where the optimal policy π^* satisfies
$$J_{\pi^*}(x_0) = J^*(x_0).$$
Note that π^* is independent of x_0.

1.1.6 Significance of Feedback

The closed-loop policies refer to that the controller can adapt to the unexpected values of the state, while the open-loop policies cannot get access to this information.

- For deterministic problems, the *open-loop policies* is as good as the *closed-loop policies*
- For stochastic problems, the reduction in the cost gained by closed-loop is called the *value of information.*

For example. consider the chess match problem,

1. The probability of win using optimal open-loop policy is
$$p_w^2 + p_w(1-p_w) \cdot \max(2p_w, p_d)$$
2. The probability of win using optimal closed-loop policy is
$$p_w^2(2-p_2) + p_w(1-p_w) \cdot p_d$$
3. Therefore, we see that the value of information is sometimes positive:
$$\text{Value of Information } = p_w(1-p_w)\min(p_w, p_d - p_w)$$

Variants of DP In the second-half of this couse, we may consider several variants of DP problems:

- Continuous-time problems
- Imperfect state information problems
- Infinite horizon problems
- Suboptimal Control

1.2 Dynamic Programming Solving

1.2.1 Principle of Optimality

Suppose $\pi^* = \{\mu_0^*, \dots, \mu_{N-1}^*\}$ be optimal policy. Consider the *tail subproblem*, i.e., we are at x_i at time i and wish to minimize the *cost-to-go* from time i to time N:

$$\mathbb{E}\left\{g_N(x_N) + \sum_{k=i}^{N-1} g_k(x_k, \mu_k(x_k), \omega_k)\right\}$$

Consider the *tail policy* $\{\mu_i^*, \mu_{i+1}^*, \dots, \mu_{N-1}^*\}$.

Theorem 1.1 (Principle of Optimality)**.** The tail policy is optimal for the tail subproblem, i.e., the optimization of the future does not depend on what we did in the past.

Remark 1.2. Consider an auto travel analogy. Suppose that the fastest route from A to C passes the place B. The principle of optimality highlights that *the B to C portion of the route is also the fastest route of the trip that starts from B and ends at C*.

The DP aims to solve all tail subproblems of N horizons. The DP algorithm is based on the idea as follows: It proceeds sequentially, by solving all the tail subproblems of a given time length, based on the solution of the tail subproblems of shorter time length.

1.2.2 DP algorithm for scheduling problem

Consider the scheduling problem discussed in subsection (1.1.3). The startup cost and transition cost is summarized in the figure (1.4).

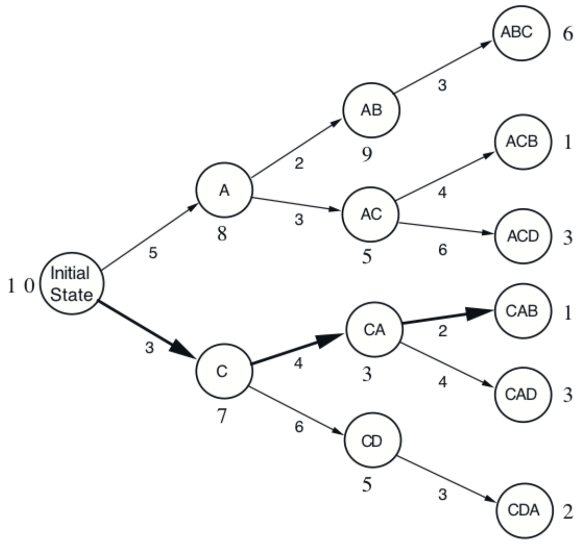

Figure 1.4: Scheduling Problem Cost Description

Tail Problem of Length 1: The subproblems involving one operations are trival to solve. State ABC,ACB,ACD,CAB,CAD,CDA correspond to the tail cost $6, 1, 3, 1, 3, 2$.

Tail Problem of Length 2: Solving the subproblems involving two operations is based on the tail problem of length 1. State AB, AC, CA, CD corresponds to the tail cost $9, \min(4+1, 6+3) = 5, \min(2+1, 4+3) = 3, 5$.

Tail Problem of Length 3: Solving the subproblems involving three operations is based on the tail problem of length 2. State A, C corresponds to the tail cost $\min(2 + 9, 3 + 5) = 8, \min(4 + 3, 6 + 5) = 7$.

Tail Problem of Length 4: Solving the subproblems involving four operations is based on the tail problem of length 3. The initial state corresponds to the tail cost $\min(5 + 8, 3 + 7) = 10$.

After computing the optimal cost, we construct the optimal solution by starting at the initial node and proceeding forward, where each time we choose the operation that starts the *optimal* schedual for the corresponding *tail subproblem*. Therefore, at each state-time pair, we need to record both the optimal cost-to-go and the optimal decision.

1.2.3 Stochastic Inventory Example

Consider the inventory problem in subsection (1.1.2). The tail subproblems of length 1 aims to solve

$$J_{N-1}(x_{N-1}) = \min_{u_{N-1}\geq 0} \mathbb{E}_{\omega_{N-1}}\{c\cdot u_{N-1}+r\cdot(x_{N-1}+u_{N-1}-\omega_{N-1})\},\ \forall x_{N-1}$$

The tail subproblems of length $N-k$ aims to solve

$$J_k(x_k) = \min_{u_k\geq 0} \mathbb{E}_{\omega_k}\{c\cdot u_k + r\cdot(x_k+u_k-\omega_k) + J_{k+1}(x_k+u_k-\omega_k)\},\ \forall x_k$$

Here $J_0(x_0)$ is the optimal cost with initial state x_0.

1.2.4 Formal Dynamic Programming Algorithm

The dynamic programming aims to find the policy by starting with

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

and go backwards in time from period $N-1$ to period 0:

$$J_k(x_k) = \min_{u_k\in U_k(x_k)} \mathbb{E}_{\omega_k}\left\{g_k(x_k,u_k,\omega_k) + J_{k+1}(f_k(x_k,u_k,\omega_k))\right\},\ k=0,\dots,N-1. \tag{1.1}$$

In such case, $J_0(x_0)$ is generated at the last step, and is equal to the optimal cost $J^*(x_0)$. Furthermore, if $u_k^* = \mu_k^*(x_k)$ minimizes the RHS of (1.1) for each x_k and k, then $\pi^* = \{\mu_0^*,\dots,\mu_{N-1}^*\}$ is optimal.

Proof. Proof by induction that $J_k(x_k)$ is equal to $J_k^*(x_k)$, which is defined as the optimal cost of the tail subproblem that starts tat time k at state x_k. □

1.2.5 Linear-Quadratic Analytic Example

Consider a certain material is passed through a sequence of two ovens shown in Figure (1.5).

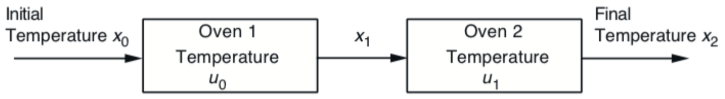

Figure 1.5: Linear-Quadratic Example

- The system is given by
$$x_{k+1} = (1-a)x_k + au_k, \quad k = 0, 1,$$
where a is a given scalar from the interval $(0, 1)$.

- Define the cost function
$$r(x_2 - T)^2 + u_0^2 + u_1^2$$
where r is a given positive scalar.

- We may apply the dp algorithm as follows:
$$\begin{aligned} J_2(x_2) &= r(x_2 - T)^2 \\ J_1(x_1) &= \min_{u_1}\left[u_1^2 + r((1-a)x_1 + au_1 - T)^2\right] \\ J_0(x_0) &= \min_{u_0}\left[u_0^2 + J_1((1-a)x_0 + au_0)\right] \end{aligned}$$

1.2.6 State Augmentation

When the assumptions of the basic problem are violated, then we need to consider reformulating the state. The DP algorithm still applies, but the problem gets bigger.

Example Consider the system becomes
$$x_{k+1} = f_k(x_k, x_{k-1}, u_k, \omega_k)$$
Introducing the additional state variable $y_k = x_{k-1}$ and view $\tilde{x_k} = (x_k, y_k)$. Therefore, the new system takes the form
$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, y_k, u_k, \omega_k) \\ x_k \end{pmatrix}$$

The DP algorithm applies to the reformulated problem:

$$J_k(x_k, x_{k-1}) = \min_{u_k \in U_k(x_k)} \mathbb{E}_{\omega_k} \left\{ g_k(x_k, u_k, \omega_k) + J_{k+1}(f_k(x_k, x_{k-1}, u_k, \omega_k), x_k) \right\}$$

2

Dynamic Programming for Deterministic System

2.1 Deterministic Finite-State Problem

Consider a shortest path problem shown in Figure (2.1). We can convert the problem into a dynamic programming system as follows:

- The states correspond to the nodes in each stage.
- The controls correspond to the acrs in each stage.
- The control sequences for this open-loop system correspond to

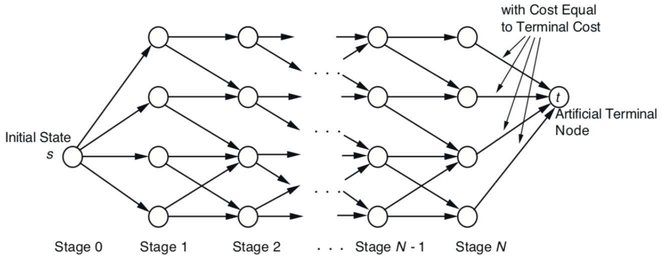

Figure 2.1: Illustration of shortest path problem

paths from initial state to the terminal states.

- Let a_{ij}^k denote the cost of transition (length of the arc) from the state $i \in S_k$ to state $j \in S_{k+1}$ at time k
- The cost of the control sequence is the cost of the corresponding path (total length of the path)

DP algorithms The backward algorithm applies as follows:

$$J_N(i) = a_{ii}^N, \quad i \in S_N$$
$$J_k(i) = \min_{j \in S_{k+1}} [a_{ij}^k + J_{k+1}(j)], \quad i \in S_k, k = N-1, N-2, \dots, 0.$$

Here the length of the shortest path from s to t equals to the $J_0(s)$.

Note that the optimal path $s \to t$ is also an optimal path $t \to s$ for a *reversed* shortest path problem, where the direction of each arc is reversed with its length unchanged.

Therefore, we can apply the backward DP algorithm for the reversed problem, which is so called the *forward* DP algorithm:

$$\tilde{J}_N(j) = a_{sj}^0, \quad j \in S_1,$$
$$\tilde{J}_k(j) = \min_{i \in S_{N-k}} \left[a_{ij}^{N-k} + \tilde{J}_{k+1}(i)\right], \quad j \in S_{N-k+1}$$

The optimal cost for the whole problem is $\tilde{J}_0(t) = \min_{i \in S_N} \left[a_{it}^N + \tilde{J}_1(i)\right]$. The interpretation for this forward DP algorithm is that we can view $\tilde{J}_k(j)$ as the *optimal cost-to-arrive* to state j from initial state s.

Remark 2.1. There is no *forward* DP algorithm for *stochastic problems*. This is becasue that we cannot restrict ourselves to open-loop sequences for the stochastic problems. In other words, due to the uncertainty, the concept of *optimal-cost-to-arrive* at the state x_k does not make sense, i.e., it's impossible to guarantee (with probability 1) that any given state can be reached.

Fortunately, even for stochastic problems, the concept of *optimal cost-to-go* from any state x_k makes clear sense.

2.2 Generic Stochastic Path Problems

Consider first a general deterministic shortest path($\boldsymbol{s}$) problems as follows:

- Let $\{1, \dots, N, t\}$ denote the nodes of a graph, where t is the destination.
- The a_{ij} denotes the cost of moving from node i to node j
- Our goal is to find a shortest (minimum cost) path from each node i to node t
- Assumption: *All cycles have non-negative* length, i.e., the optimal path *need not take more than* N *moves.*
- We can reformulate this problem as one where we require *exactly* N moves by allowing *degerate moves* from a node i to itself with cost $a_{ii} = 0$. Moreover, we let

$$J_k(i) = \textit{optimal cost of getting from } i \textit{ to } t \textit{ in } N-k \textit{ moves}$$
$$J_0(i) = \textit{cost of the optimal path from } i \textit{ to } t$$

- Therefore, we apply the DP algorithm as follows:

$$J_k(i) = \min\nolimits_{j=1,\dots,N}\left[a_{ij} + J_{k+1}(j)\right], \quad k = 0, 1, \dots, N-2,$$

with $J_{N-1}(i) = a_{it}, i = 1, \dots, N$.

For example, consider the shortest paths problem shown in Fig (2.2). Assume the destination node is the node 5. The state i means the current position is in the node i. In such case, note that

$$\begin{aligned} J_{N-1}(i) &= a_{it}, \quad i = 1, \dots, N \\ J_k(i) &= \min_{j=1,\dots,N}\left[a_{ij} + J_{k+1}(j)\right], \quad k = 0, 1, \dots, N-2 \end{aligned}$$

The solution for this problem is shown in Fig. (2.2b), where the arrow indicate the optimal move to node 5 from the position at current stage.

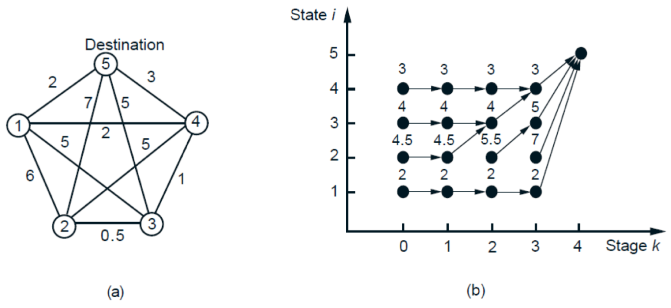

Figure 2.2: One Example of shortest paths problems

2.2.1 HMM setting and solving

Now consider the *Hidden Markov Models*. Suppose that the Markove chain is with transition probability p_{ij}, but the state transitions are hidden from our view. For each transition, we obtain an *observation* instead. Let $r(z;i,j)$ denote the probability of observation taking value z given the transition $i \to j$. The *trajectory estimation problem* is that, given the observation sequence $Z_N = \{z_1,\dots,z_N\}$, what's the *most likely* state transition sequence $\hat{X}_N := \{\hat{x}_0,\hat{x}_1,\dots,\hat{x}_N\}$, i.e., the one equals to

$$\arg\min_{\text{all } X_N} p(X_N \mid Z_N) = \frac{p(X_N, Z_N)}{p(Z_N)}$$

We can rewrite the probability $p(X_N, Z_N)$ as follows:

$$\begin{aligned}
p(X_N, Z_N) &:= p(x_{0:N}, z_{1:N}) \\
&= \pi_{x_0} \cdot p(x_{1:N}, z_{1:N} \mid x_0) \\
&= \pi_{x_0} p(x_1, z_1 \mid x_0) p(x_{2:N}, z_{2:N} \mid x_{0:1}, z_1) \\
&= \pi_{x_0} p_{x_0 \to x_1} r(z_1; x_0, x_1) p(x_{2:N}, z_{2:N} \mid x_{0:1}, z_1)
\end{aligned}$$

Following the similar idea, we compute

$$\begin{aligned}
p(x_{2:N}, z_{2:N} \mid x_{0:1}, z_1) &= p(x_2, z_2 \mid x_{0:1}, z_1) p(x_{3:N}, z_{3:N} \mid x_{0:2}, z_{1:2}) \\
&= p_{x_1 \to x_2} r(z_2; x_1, x_2) p(x_{3:N}, z_{3:N} \mid x_{0:2}, z_{1:2})
\end{aligned}$$

Therefore, we derive

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

Thus the problem is equivalent to

$$\text{minimize} \quad -\ln(\pi_{x_0}) - \textstyle\sum_{k=1}^{N} \ln\left(p_{x_{k-1}\to x_k} r(z_k; x_{k-1}, x_k)\right)$$

over all possible sequences $\{x_0, x_1, \ldots, x_N\}$. Therefore, due to the structure of our objective function, we can apply a distributed algorithm (e.g., dynamic programming) to solve it.

2.3 General Shortest Path Algorithms

Next, we discuss some non-DP shortest path algorithms. Note that they all can *only* be used to solve deterministic finite-state problems. The advantage is that they avoid computing the optimal cost-to-go for *every* state. This is essential for problems with *huge* state spaces. (see examples in combinatorial optimization)

2.3.1 Label Correcting Methods

The idea of this kind of algorithm is to *progressively* discover shorter paths from the origin to *any other* node i, and to maintain the length of the shortest path found *so far* in a variable d_i (called the *label* of i).

- If a shorter path to i is found, then d_i is reduced, and the algorithm checks if the labels d_j can be reduced, where j is the children of i, i.e., whether labels d_j's can be reduced by setting them into $d_i + a_{ij}$.

- This algorithm makes use of a list of nodes called *OPEN* (or *candidate list*). This list contains nodes that are currently active such that they are candidates for further examination by the algorithm, and possible inclusion in the shortest path.

  Initially, OPEN contains only s. Each node that has entered the OPEN at least once (except s) is assigned a *parent* (refers to some node). The parent nodes are not necessary for the computation of shortest distance, but for tracing the shortest path to the origin after the algorithm terminates.

Notations

- d_i denotes the *label* of i, i.e., the length of the shortest path from s to i found so far. Initially $d_s = 0, d_i = \infty, \forall i \neq s$
- UPPER: The label d_t to the destination
- OPEN list: contains nodes that are currently active in the sense that they are candidates for further examination (initially OPEN $= \{s\}$)

The label correcting methods work as follows:

Algorithm 1 The Label Correcting Methods

```
1: (Node Removal): Remove a node $i$ from OPEN and for each child $j$ of $i$, do step 2
2: (Node Insertion Test): If $d_i + a_{ij} < \min\{d_j, \text{UPPER}\}$, set $d_j = d_j + a_{ij}$, and set $i$ to be the parent of $j$. In additiion, if $j \neq t$, place $j$ into the OPEN if $j \notin$ OPEN; otherwise set UPPER to the new value $d_i + a_{it}$ of $d_t$
3: (Termination Test): If OPEN is empty, terminate, else go to step 1.
```

Here we show how to apply the label correcting algorithm into the travelling salesman problem shown below.

Iteation Number	Node Exiting OPEN	OPEN after Iteration	Upper
0	–	1	∞
1	1	2, 7, 10	
2	2	3, 5, 7, 10	∞
3	3	4, 5, 7, 10	∞
4	4	5, 7, 10	43
5	5	6, 7, 10	43
6	6	7, 10	13
7	7	8, 10	13
8	8	9, 10	13
9	9	10	13
10	10	$\emptyset$	13

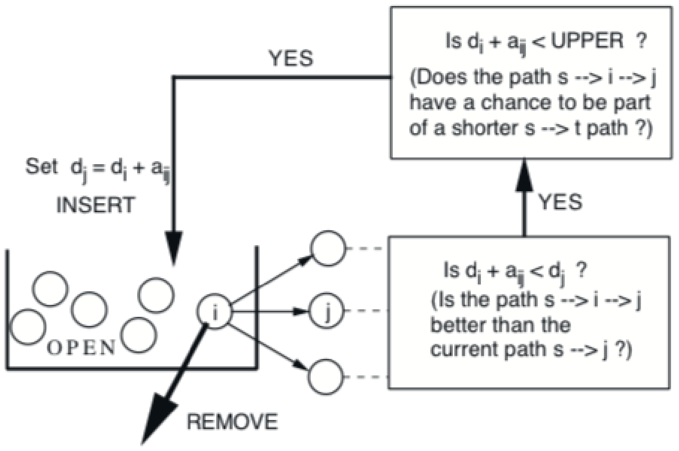
YES
Is $d_i + a_{ij} <$ UPPER ?
(Does the path s --> i --> j have a chance to be part of a shorter s --> t path ?)
Set $d_j = d_i + a_{ij}$
INSERT
YES
OPEN
i
j
Is $d_i + a_{ij} < d_j$?
(Is the path s --> i --> j better than the current path s --> j ?)
REMOVE

Figure 2.3: Diagram Illustration of Label Correcting Methods

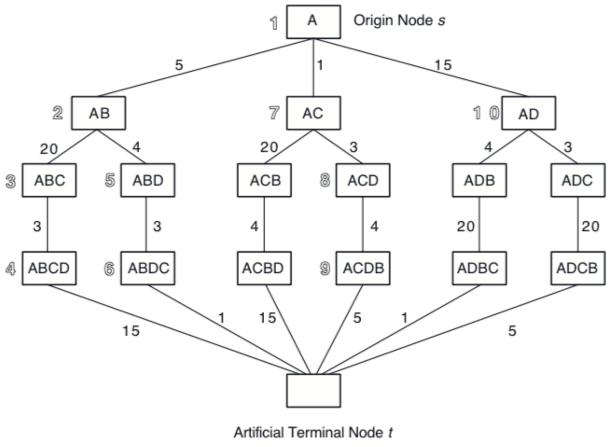

Figure 2.4: One example for travelling salesman problem

Note that some nodes never entered OPEN.

Finally, we give a verification for why label correcting methods work.

Proposition 2.1. If there exists at least one path from the origin to the destination, the label correcting algorithm terminates with UPPER equal to the shortest distance from the origin to the destination.

Proof.

1. Each time a node j enters OPEN, its label is decreased and becomes equal to the length of some path from i to j

2. The number of possible distinct path lengths is finite, and therefore the number of timesa node can enter is finite, and the algorithm terminates.

3. Let $\{(s, j_1, \dots, j_k, t), d^*\}$ be the pair of shortest path and the corresponding shortest distance. If UPPER $> d^*$ at the termination, UPPER will also be larger than the length of all parths $(s, j_1, \dots, j_m), , = 1, \dots, k$ throguhout the algorithm, i.e., the node j_k will never enter the OPEN list with d_{j_k} equal to the shortest distance from s to j_k.

   Similarly, node j_{k-1} will never enter to the OPEN list with $d_{j_{k-1}}$ equal to the shortest distance from s to j_{k-1}. Continue to j_1 by induction will obtain a contradiction.

□

3

Dynamic Programming with perfect information

3.1 Linear-Quadratic Problems

Now consider the case where the future observations is linear w.r.t. current observation and controller:

$$x_{k+1} = A_k x_k + B_k u_k + \omega_k$$

The goal is to minimize the quadratic cost

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

Assumptions:

1. $Q_k \succeq 0$ and $R_k \succ 0$;
2. the noise ω_k are *independent* with *zero mean*.

We may apply the DP algorithm to solve it:

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

We *claim* several key facts:

Theorem 3.1. 1. $J_k(x_k)$ is *quadratic*

2. Optimal policy $\{\mu_0^*, \dots, \mu_{N-1}^*\}$ is *linear*, i.e.,

$$\mu_k^*(x_k) = L_k x_k, \quad k = 0, \dots, N-1$$

Therefore, we may apply the similar treatment to solve for $J_k(x_k)$ with $k = N-1, N-2, \dots, 1$.

Proof. By induction it is easy to verify

$$\mu_k^*(x_k) = L_k x_k, \quad J_k(x_k) = x_k^{\mathrm{T}} K_k x_k + \text{constant}$$

where

$$\begin{aligned} L_k &= -(B_k^{\mathrm{T}} K_{k+1} B_k + R_k)^{-1} B_k^{\mathrm{T}} K_{k+1} A_k, \ k = 0, \dots, N-1 \\ K_N &= Q_N \\ K_k &= A_k^{\mathrm{T}}(K_{k+1} - K_{k+1} B_k (B_k^{\mathrm{T}} K_{k+1} B_k + R_k)^{-1} B_k^{\mathrm{T}} K_{k+1}) A_k + Q_k \end{aligned}$$

The equations above are called the *discrete-time Riccatic equation.*

Similar as DP, it starts at the terminal time N and proceeds backwards. □

Asymptotic Behavior

Theorem 3.2. Consider the *Riccatic equation*

$$K_{k+1} = A^{\mathrm{T}}(K_k - K_k B(B^{\mathrm{T}} K_k B + R)^{-1} B^{\mathrm{T}} K_k) A + Q, \quad k = 0, 1, 2, \dots$$

with the initial K_0 be arbitrary semidefinite metrix. Suppose that (A, B) is controllable, and (A, C) is observable. As a result:

1. The Riccati equation converges:

$$\lim_{k \to \infty} K_k = K,$$

where K is the unique solution of the *algebraic Riccatic equation*

$$K = A^{\mathrm{T}}(K - KB(B^{\mathrm{T}} K B + R)^{-1} B^{\mathrm{T}} K) A + Q$$

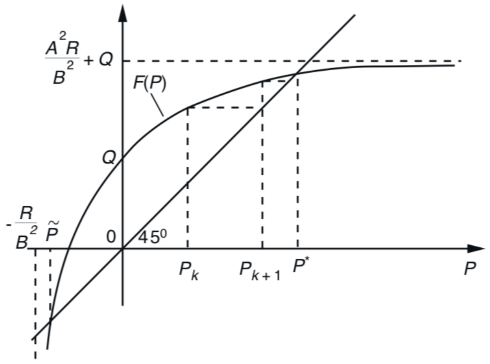

Figure 3.1: Graphical Proof For Scalar Systems

2. The corresponding steady-state controller

$$\mu^*(x) = Lx, \quad \text{where } L = -(B^{\mathrm{T}}KB + R)^{-1}B^{\mathrm{T}}KA$$

is stable, i.e., the matrix $(A + BL)$ for the closed loop system

$$x_{k+1} = (A + BL)x_k + \omega_k$$

satisfies $\lim_{k\to\infty}(A + BL)^k = 0$.

Proof for scalar systems. Consider the scalar stationary system with $A \neq 0, B \neq 0, Q > 0, R > 0$. The Riccatic equation becomes

$$P_{k+1} = A^2\left(P_k - \frac{B^2P_k^2}{B^2P_k + R}\right) + Q$$

Suppose that $P_{k+1} = F(P_k)$, where

$$F(P) = \frac{A^2RP}{B^2P + R} + Q$$

In Fig (3.1) we find that $F(P)$ has two fixed points P^* and $\tilde{P}$. Note that $F(P)$ is increasing on $(-R/B^2, \infty)$. Therefore, the Riccati iteration coverges to P^* starting everywhere in the inerval $(\tilde{P}, \infty)$ as shown in the figure. □

Random System Matrices Suppose that $\{A_0, B_0\}, \dots, \{A_{N-1}, B_{N-1}\}$ are not known but *independent random matrices* that are also independent of the ω_k. The DP algorithm is given by:

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

Following the similar proof in Theorem (3.1), we imply

$\mu^* k(x_k) = L_k x_k$, where $L_k = -(R_k + \mathbb{E}\{B_k^{\mathrm{T}} K_{k+1} B_k\})^{-1} \mathbb{E}\{B_k^{\mathrm{T}} K_{k+1} A_k\}$

and the cost-to-go function $J_k(x_k) = x_k^{\mathrm{T}} K_k x_k + \text{constant}$, with

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

However, there are several drawbacks for random matrices systems:

1. Certainty Equivalence may not hold (what is certainty equivalence?) Formally speaking, the optimal controller may not be the same as for the corresponding deterministic problem, when the randomness is replaced by its expectation.

2. Riccati equation may not converge to a steady-state.

Consider the scalar stationary system such that for the update for K_{k+1}, the random matrices A_k, B_k are replaced by their expectation. Therefore, the update for K_{k+1} can be written as

$$\begin{cases}
K_{k+1} = \tilde{F}(K_k) \\
\tilde{F}(K) = \dfrac{\mathbb{E}\{A^2\} B K}{\mathbb{E}\{B^2\} K + R} + Q + \dfrac{T K^2}{\mathbb{E}\{B^2\} K + R} \\
T = \mathbb{E}\{A^2\}\mathbb{E}\{B^2\} - (\mathbb{E}\{A\})^2 (\mathbb{E}\{B\})^2
\end{cases}$$

When $T = 0$, the problem reduces to the case where A, B are not random; and we can obtain the well-studied asymptotic behavior; the convergece also happens as T is small. However, for large enough T, as shown in Fig (3.2), the graph of F do not intersect with the 45-degree line on the positive interval, i.e., the Riccati equation diverges to infinity.

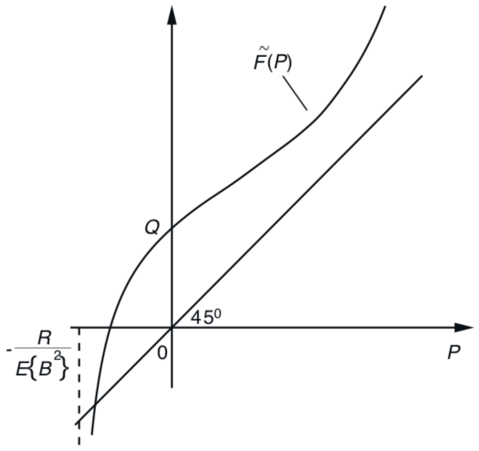

Figure 3.2: Graphical Illustration for a scalar stationary system

3.2 Inventory Control

- State: inventory level x_k
- Control: number of orders u_k
- Disturbence: demand ω_k
- System dynamics: $x_{k+1} = x_k + u_k - \omega_k$, $k = 0, 1, \ldots, N-1$
- Stage cost:

$$c \cdot u_k + H(x_k + u_k)$$

where

$$H(x_k + u_k) := \mathbb{E}_{\omega_k}\left[p \cdot (x_k + u_k - \omega_k)^- + h \cdot (x_k + u_k - \omega_k)^+\right]$$

The goal is to minimize the total cost (question: do we need to add the expectation symbol?)

$$\mathbb{E}\left\{\sum_{k=0}^{N-1}(cu_k + H(x_k + u_k))\right\}$$

The dynamic programming algorithm applies as follows:

$$\begin{aligned} J_N(x_N) &= 0 \\ J_k(x_k) &= \min_{u_k \geq 0} \left[c \cdot u_k + H(x_k + u_k) + \mathbb{E}_{\omega_k} J_{k+1}(x_{k+1})\right] \end{aligned}$$

Optimal Policy Regaring $y_k = x_k + u_k$ as the decision variable, it suffices to solve

$$\begin{array}{ll} \min_{y_k \geq x_k} & G_k(y_k) - cx_k, \\ \text{where} & G_k(y) = cy + H(y) + \mathbb{E}_\omega \left[J_{k+1}(y - \omega)\right] \end{array}$$

Proposition 3.1. If G_k is convex, and $\lim_{|x|\to\infty} G_k(x) \to \infty$, we have the optimal policy

$$\mu_k^*(x_k) = \begin{cases} S_k - x_k, & \text{if } x_k < S_k \\ 0, & \text{if } x_k \geq S_k \end{cases}$$

where S_k is the minimizer of $G_k(y)$.

Now we show the assumption for proposition (3.1) mostly holds. We need to add the assumption $c < p$, since otherwise it would never be optimal to order new iterms at the last period.

Proposition 3.2. Assume that $c < p$, then J_k is convex for all k and

$$\lim_{|x|\to\infty} J_k(x) \to \infty, \quad k = 0, 1, \ldots, N-1$$

Proof. 1. It's easy to see that $J_N \equiv 0$ is convex.

2. Since $c < p$ and $H'(y) \to -p$ as $y \to -\infty$, we imply that $G_{N-1} := cy + H(y)$ has a derivative that becomes negative as $y \to -\infty$ and becomes positive when $y \to \infty$ (see Fig (3.3)). Therefore,

$$\lim_{|y|\to\infty} G_{N-1}(y) = \infty.$$

As have shoen,

$$\mu_{N-1}^*(x_{N-1}) = \begin{cases} S_{N-1} - x_{N-1}, & x_{N-1} < S_{N-1} \\ 0, & x_{N-1} \geq S_{N-1} \end{cases}$$

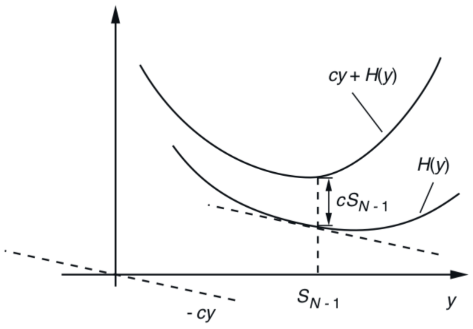

Figure 3.3: Graphical Illustration for the structure of cost-to-go function

Or equivalently, $\mu_{N-1}^*(x_{N-1}) = (S_{N-1} - x_{N-1})^+$ (question: can we write in this form?). Substituting μ_{N-1}^* into J_{N-1}, we imply

$$J_{N-1}(x_{N-1}) = \begin{cases} c \cdot (S_{N-1} - x_{N-1}) + H(S_{N-1}), & \text{if } x_{N-1} < S_{N-1} \\ H(x_{N-1}), & \text{if } x_{N-1} \geq S_{N-1} \end{cases}$$

which is clearly a convex function (question: why?)

Therefore, given the convexity of J_N, we are able to show the convexity of J_{N-1}. Furthermore,

$$\lim_{|y|\to\infty} J_{N-1}(y) = \infty.$$

3. Similarly, given that J_{k+1} is convex and $\lim_{|y|\to\infty} J_{k+1}(y) = \infty$, $\lim_{|y|\to\infty} G_k(y) = \infty$, for $k = N-2, \dots, 0$, we imply

$$J_k(x_k) = \begin{cases} c \cdot (S_k - x_k) + H(S_k) + \mathbb{E}\{J_{k+1}(S_k - \omega_k)\}, & \text{if } x_k < S_k \\ H(x_k) + \mathbb{E}\{J_{k+1}(x_k - \omega_k)\}, & \text{if } x_k \geq S_k \end{cases}$$

where S_k is the unconstrained minimizer of G_k. Moreover, we imply J_k is convex, $\lim_{|y|\to\infty} J_k(y) = \infty$ and $\lim_{|y|\to\infty} G_{k-1}(y) = \infty$.

$\square$

Therefore, we reduce the policy-based optimization into the pattern of finding the baseline S_k, i.e., a traditional parametric optimization problem.

Proposition 3.3. If the disturbance ω_k is i.i.d., then we imply the stationary optimal policy, i.e., the policy baseline S_k^* becomes constant.

Now suppose further that

$$c(u_k) = \begin{cases} K + c \cdot u_k, & u_k > 0 \\ 0, & u_k = 0 \end{cases}$$

Now the question is that do we obtain the pattern with baseline?

$$\begin{aligned} J_k(x_k) &= \min[\min_{u_k>0}[K + cu_k + H(x_k + u_k)] + \mathbb{E}_{\omega_k} J_{k+1}(x_{k+1}) \\ &\quad, 0 + H(x_k) + \mathbb{E} J_{k+1}(x_k)] \\ &= \min\left[\min_{y_k \ge x_j} (K + G_k(y_k)), G_k(x_k)\right] - cx_k \end{aligned}$$

where $f(x_k) := \min_{y_k \ge x_j}(K + G_k(y_k))$ has its shape. Then we need to compare the two curves $f(x_k)$ and $G_k(x_k)$:

$$u_k^* = \begin{cases} S_k - x_k, & \text{if } x_k \le S_k \\ 0, & \text{if } x_k > S_k \end{cases}$$

- If the inventory is very low (compare the baseline s_k), order $u_k^* = S_k - x_k$ to make the inventory into S_k
- If not that low, not order anything.

Remark 3.1. Unfortunately, $J_{k+1}(x_k)$ is convex does not necessarily imply $J_k(x_k)$ is convex. There is a concept K-convexity in literature. Under this K-convexity setting, we can show that the control u_k^* still obtain the similar pattern but with two baselines, i.e., the decision can be made by first computing the baseline (s, S), and then obtaining the (s, S) strategy.

The key for reducing computational complexity is to find patterns for decision variable u_k^*.

3.3 Optimal Stopping Problem

There are two possible controls for a stopping problem:

1. Stop, i.e., incur a one-time stopping cost, and move to a *cost-free*, *absorbing* stop state.
2. Continue, using $x_{k+1} = f_k(x_k, \omega_k)$ and incurring the *cost-per-stage*

Therefore, eacy policy should consist of a partition of a set of states x_k into two regions:

1. Stop region: where we stop?
2. Continue region, where do we continue?

The natural question is that can we obtain an optimal policy that is similar to the pattern of the *baseline* studied in inventory control problem?

Asset Selling Suppose an industry wants to sell its iterms. This industry receives a random price ω_k for $k = 1, \dots, N-1$, which are assumed to be i.i.d. with known distribution. This industry may accept ω_k and invest the money at a fixed rate of interest r; or reject ω_k and wait for ω_{k+1}. It must accept the last offer ω_{N-1}

We may formulate this problem as follows:

- State: $x_k = \omega_k$
- Decision variable: u^1 for sell and u^2 for wait.
- Stage reward:
$$g(x_k, u_k, \omega_k) = \begin{cases} x_k(1+r)^{N-1}, & \text{if } u_k = u^1 \\ 0, & \text{if } u_k = u^2 \end{cases}$$
- Therefore, the DP algorithm is by solving the optimization problem below recursively:
$$J_N(x_N) = \begin{cases} x_N & \text{if } x_N \neq T \\ 0, & \text{if } x_N = T \end{cases}$$
$$J_k(x_k) = \begin{cases} \max\left[(1+r)^{N-k}x_k, \mathbb{E}\{J_{k+1}(\omega_k)\}\right] & \text{if } x_k \neq T \\ 0, & \text{if } x_k = T \end{cases}$$

Specifically, note that for k-th stage, if $x_k(1+r)^{N-k} \geq \mathbb{E}_{\omega_k}[J_{k+1}(\omega_k)]$, then the industry should decide to sell; otherwise decide to wait.

Define the variable (baseline)

$$\alpha_k = \frac{\mathbb{E}J_{k+1}(\omega)}{(1+r)^{N-k}},$$

then the optimal policy should be:

$$\begin{array}{ll} \text{accept the offer } x_k & \text{if } x_k > \alpha_k \\ \text{reject the offer } x_k & \text{if } x_k < \alpha_k \end{array}$$

In particular, the high baseline corresponds to the high standard for offering the price.

Future Analysis

Theorem 3.3. Based on the assumption of i.i.d. ω_k, the baseline $\alpha_k \geq \alpha_{k+1}$ for all k.

Proof. First introduce the functions

$$V_k(x_k) = \frac{J_k(x_k)}{(1+r)^{N-k}}, \qquad x_k \neq T$$

The DP algorithm reduces into

$$\begin{aligned} V_N(x_N) &= x_N \\ V_k(x_k) &= \max\left[x_k, (1+r)^{-1}\mathbb{E}_\omega\{V_{k+1}(\omega)\}\right] \end{aligned}$$

Note that $\alpha_k = \mathbb{E}\{V_{k+1}(\omega)\}/(1+r)$. It's clear that

$$V_{N-1}(x) \geq V_N(x), \quad \forall x \geq 0$$

Following the similar procedure, we derive $V_k(x) \geq V_{k+1}(x)$ for all x and k. Since $\alpha_k = \mathbb{E}\{V_{k+1}(\omega)\}/(1+r)$, we obtain $\alpha_k \geq \alpha_{k+1}$, as desired. □

Remark 3.2. We can also show that $\alpha_k \to \bar{\alpha}$ as $k \to \infty$, where $\bar{\alpha}$ is a constant satisfying

$$(1+r)\bar{\alpha} = P(\bar{\alpha})\bar{\alpha} + \int_{\bar{\alpha}}^{\infty} \omega \, \mathrm{d}P(\omega)$$

Therefore, for an infinite horizon, the optimal policy is stationary.

Now we show that $\{\alpha_k\}$ is monotonically decreasing.

Define

$$V_k(x_k) = \frac{J_k(x_k)}{(1+r)^{N-k}} \implies V_k(x_k) = \max\left(x_k, \frac{\mathbb{E}V_{k+1}(\omega)}{1+r}\right)$$

and $V_N(x_N) = x_N$.

Note that we show $V_{k+1}(x) \leq V_k(x)$ first:

$$\begin{aligned} V_N(x) &= x \\ V_{N-1}(x) &= \max(x, \mathbb{E}V_N(\omega)) \end{aligned}$$

It's clear that $V_N(x) \leq V_{N-1}(x)$.

$$\begin{aligned} V_K(x) &= \max(x, \frac{\mathbb{E}V_{k+1}(\omega)}{1+r}) \\ V_{k-1}(x) &= \max(x, \frac{\mathbb{E}V_k(\omega)}{1+r}) \end{aligned}$$

It's clear that $V_k(x) \leq V_{k-1}(x)$ for any k. Therefore, $\{\alpha_k\}$ is monotonically decreasing.

Note that the key for this proof is the i.i.d. of ω_k.

- Pattern
- Trends
- Convergence: If there are inifinte horizons, what is the bottleneck for the policy.

In this case,

$$\begin{aligned} \alpha_k = \frac{\mathbb{E}V_{k+1}(\omega)}{1+r} &= \frac{1}{1+r}\left[\int_0^{\alpha_{k+1}} \alpha_{k+1} dP(\omega) + \int_{\alpha_{k+1}}^{\infty} \omega dP(\omega)\right] \\ &= \frac{P(\alpha_{k+1})}{1+r}\alpha_{k=1} + \frac{1}{1+r}\int_{\alpha_{k+1}}^{\infty} \omega dP(\omega) \end{aligned} \tag{3.1}$$

where the $P(\alpha_{k+1})$ is the cdf, and α_k is bounded.

Start from $\alpha_N = 0$, derive $\alpha_{N-1}, \ldots, \alpha_0$ sequentially.

When $N \to \infty$, note that computing the equation above is based on the derivation of α_{N+1}. We can imply the convergence of α, and we can compute α_{N+1} based on (3.1).

4

Dynamic Programming with imperfect information

Motivation We have assumed so far that the controller has access to the exact value of the current state, which may be unrealistic in pratice. We model the situation that at each stage the controller may receive some observations about the value of the current state, which may be corrupted by stochastic uncertainty.

4.1 Imperfect State Problem Setting

- Observation Space: Z_k, where

$$z_0 = h_0(x_0, v_0), \quad z_k = h_k(x_k, u_{k-1}, v_k), \quad k = 1, \dots, N-1$$

- Observation disturbance: $v_k \in V_k$, characterized by

$$P_{v_k}(\cdot \mid x_{k:0}, u_{k-1:0}, \omega_{k-1:0}),$$

  i.e., depends on the whole history of the sytem up to the period $k-1$.

- The initial state x_0 may be unknown, but the distribution P_{x_0} is known.

- The probability distribution for disturbance of underlying dynamics, say ω_k is known, denoted as, $P_{\omega_k}(\cdot \mid x_k, u_k)$, which depends explicitly on x_k and u_k but not the prior disturbances $\omega_0, \ldots, \omega_k, v_0, \ldots, v_{k-1}$.
- Control: $u_k \in U_k$. We assume that U_k does not depend on x_k (why?)
- Information vector: the information available to the controller at stage k, to make its decision u_k, denoted as

$$\begin{aligned} I_0 &= z_0, \\ I_k &= (z_{0:k}, u_{0:k-1}),\ k = 1, \ldots, N-1 \end{aligned}$$

- The goal is to derive the policy

$$\pi = (\mu_0(I_0), \ldots, \mu_{N-1}(I_{N-1}))$$

where $\mu_k(\cdot) : I_k \to I_k$ for $k = 0, \ldots, N-1$, to minimize the cost function

$$J_\pi = \mathbb{E}_{x_0, \omega_k, v_k, k=0,\ldots,N-1} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(I_k), \omega_k) \right\}$$

subject to

1. the underlying dynamics equation

$$x_{k+1} = f_k(x_k, \mu_k(I_k), \omega_k),\ k = 0, 1, \ldots, N-1 \tag{4.1}$$

2. the measurement equation

$$\begin{aligned} z_0 &= h_0(x_0, v_0) \\ z_k &= h_k(x_k, \mu_{k-1}(I_{k-1}), v_k),\ k = 1, \ldots, N-1 \end{aligned} \tag{4.2}$$

4.2 Reformulation as a perfect state information system

- Dynamics:

$$\begin{aligned} I_0 &= z_0 \\ I_{k+1} &= (I_k, z_{k+1}, u_k), k = 0, 1, \ldots, N-2 \end{aligned}$$

- State: $I_k := (z_{0:k}, u_{0:k-1})$

- Control: $u_k \in U_k$

- Disturbance: we view z_{k+1} as the noise, and its distribution only depends on the state I_k and control u_k.

- Stage Cost: Since we have

$$\mathbb{E}\{g_k(x_k, u_k, \omega_k)\} = \mathbb{E}\left\{\mathbb{E}_{x_k,\omega_k}\{g_k(x_k, u_k, \omega_k) \mid I_k, u_k\}\right\}$$

  we imply the stage cost can be reformulated as a function of the state I_k and the control u_k:

$$\tilde{g}_k(I_k, u_k) = \mathbb{E}_{x_k,\omega_k}\{g_k(x_k, u_k, \omega_k) \mid I_k, u_k\}$$

Therefore, the dynamic programming algorithm applies as follows:

1. New cost function:

$$\mathbb{E}\left\{\tilde{g}_N(I_N) + \sum_{k=0}^{N-1} \tilde{g}_k(I_k, u_k)\right\} \tag{4.3a}$$

2. New cost-to-go function:

$$J_N(I_N) = \tilde{g}_N(I_N) = \mathbb{E}_{x_{N-1},\omega_{N-1}} g_N(f_{N-1}(x_{N-1}, u_{N-1}, \omega_{N-1})) \tag{4.3b}$$

$$J_k(I_k) = \min_{u_k \in U_k} \{\tilde{g}_k(I_k, u_k) + \mathbb{E}[J_{k+1}(I_{k+1}) \mid I_k, u_k]\}, \ k = 0 : N-1 \tag{4.3c}$$

Or more precisely,

$$J_k(I_k) = \min_{u_k \in U_k} \Bigg\{ \mathbb{E}_{x_k, \omega_k, z_{k+1}} g_k(x_k, u_k, \omega_k) + J_{k+1}(I_k, z_{k+1}, u_k) \mid I_k, u_k \Bigg\}, \ k = 0 : N-2 \tag{4.3d}$$

$$J_{N-1}(I_{N-1}) = \min_{u_{N-1} \in U_{N-1}} \Bigg\{ \mathbb{E}_{x_{N-1}, \omega_{N-1}} g_{N-1}(x_{N-1}, u_{N-1}, \omega_{N-1}) + g_N(f_{N-1}(x_{N-1}, u_{N-1}, \omega_{N-1})) \mid I_{N-1}, u_{N-1} \Bigg\} \tag{4.3e}$$

We need the conditional distribution $P(x_k \mid I_k)$ and $P(z_{k+1} \mid I_k)$.

4.3 Examples for Imperfect State Information Programming

4.3.1 Slotted Aloha

- Dynamics:

$$x_{k+1} = x_k + a_k - t_k$$

- Measurement:

$$z_{k+1} = \begin{cases} \text{Success}, & \text{with probability } (1-u_k)^{x_k} \\ \text{collision}, & \text{with probability } x_k u_k (1-u_k)^{x_k - 1} \end{cases}$$

How to model?

4.3.2 Machine Repair Example

- State Transition:

$$\begin{array}{ll} p(P \to P) = \frac{2}{3}, & p(P \to \bar{P}) = \frac{1}{3} \\ p(\bar{P} \to P) = 0, & p(\bar{P} \to \bar{P}) = 1 \end{array}$$

- Inspection:

$$\begin{array}{ll} p(z = G \mid x = P) = \frac{3}{4}, & p(z = B \mid x = P) = \frac{1}{4} \\ p(z = G \mid x = P) = \frac{1}{4}, & p(z = B \mid x = \bar{P}) = \frac{3}{4} \end{array}$$

  Two possible actions can be taken after each inspection:

  - C: continue operation of the machine
  - S: stop the machine and determine its state through accurate diagonastic. If it is in bad state $\bar{P}$, then back to good P

- The cost for decision C is 0; for decision S is 1.

  At each stage if the state is $\bar{P}$ then cost 2; if the state is P then cost 0.

- Total Horizon: 2.

We can model this problem more clear:

- State Space: $\{P, \bar{P}\}$; Control Space: $\{C, S\}$; Measurment Space: $\{G, B\}$

- State Transition:

$$x_{k+1} = \omega_k,\ k = 0, 1$$

  where ω_k is a random variable depending on x_k, u_k:

$$\begin{aligned} P(\omega_k = P \mid (x_k, u_k) = (P, C)) &= 2/3 \\ P(\omega_k = \bar{P} \mid (x_k, u_k) = (P, C)) &= 1/3 \\ P(\omega_k = P \mid (x_k, u_k) = (\bar{P}, C)) &= 0 \\ P(\omega_k = \bar{P} \mid (x_k, u_k) = (\bar{P}, C)) &= 1 \\ P(\omega_k = P \mid (x_k, u_k) = (P, S)) &= 2/3 \\ P(\omega_k = \bar{P} \mid (x_k, u_k) = (P, S)) &= 1/3 \\ P(\omega_k = P \mid (x_k, u_k) = (\bar{P}, S)) &= 2/3 \\ P(\omega_k = \bar{P} \mid (x_k, u_k) = (\bar{P}, S)) &= 1/3 \end{aligned}$$

- Initial State Distribution:

$$P(x_0 = P) = 2/3,\ P(x_0 = \bar{P}) = 1/3$$

- Measurment Equation:

$$z_k = v_k,\ k = 0, 1,$$

where v_k is a random variable depending on x_k:

$$\begin{aligned} P(v_k = G \mid x_k = P) &= 3/4 \\ P(v_k = B \mid x_k = P) &= 1/4 \\ P(v_k = G \mid x_k = \bar{P}) &= 1/4 \\ P(v_k = B \mid x_k = \bar{P}) &= 3/4 \end{aligned}$$

- Total Cost:

$$g(x_0, u_0) + g(x_1, u_1)$$

where

$$g(P, C) = 0,\ g(P, S) = 1,\ g(\bar{P}, C) = 2,\ g(\bar{P}, S) = 1$$

We can apply the DP to solve this problem:

- Construct the information vector from stage 0 to stage $N - 1 = 1$:

$$I_0 = z_0, \quad I_1 = (z_0, z_1, u_0)$$

- Seek the controllers $\mu_0(I_0)$ and $\mu_1(I_1)$ to minimize the cost function

$$\begin{aligned} &\mathbb{E}_{x_0, \omega_{0:1}, v_{0:1}}\left\{ g_0(x_0, \mu_0(I_0)) + g_1(x_1, \mu_1(I_1)) \right\} \\ = &\mathbb{E}_{x_0, \omega_{0:1}, v_{0:1}}\left\{ g_0(x_0, \mu_0(z_0)) + g_1(x_1, \mu_1(z_0, z_1, \mu_0(z_0))) \right\} \end{aligned}$$

- The cost function is

$$\begin{aligned} &\mathbb{E}_{x_0, \omega_0, \omega_1, v_0, v_1}\left\{ g(x_0, u_0) + g(x_1, u_1) \right\} \\ &\qquad = \mathbb{E}_{x_0, \omega_0, \omega_1, v_0, v_1}\left\{ g(x_0, \mu_0(z_0)) + g(x_1, \mu_1(z_0, z_1, \mu_0(z_0))) \right\} \end{aligned}$$

with

$$g(P, C) = 0, \quad g(P, S) = 1, \quad g(\bar{P}, C) = 2, \quad g(\bar{P}, S) = 1$$

- Thererfore, we construct the cost-to-go function:

$$
\begin{aligned}
J_2(I_2) &= 0,\\
J_1(I_1) &= \min_{u_1 \in \{C,S\}} \left\{ \mathbb{E}_{x_1} g_1(x_1, u_1) \middle| I_1 \right\}\\
&= \min \left\{ \mathbb{E}_{x_1} g_1(x_1, C) \middle| I_1, \mathbb{E}_{x_1} g_1(x_1, S) \middle| I_1 \right\}\\
&= \min \left\{ \mathbb{E}_{x_1} g_1(x_1, C) \middle| I_1, 1 \right\}\\
&= \min \left\{ 2 \cdot P(x_1 = \bar{P} \mid I_1), 1 \right\}\\
J_0(I_0) &= \min_{u_0 \in \{C,S\}} \left\{ \mathbb{E}_{x_0} g_0(x_0, u_0) \mid I_0 + \mathbb{E}_{z_1}\{J_1(z_0, z_1, u_0) \mid I_0, u_0\} \right\}\\
&= \min_{u_0 \in \{C,S\}} \left\{ \mathbb{E}_{x_0} g_0(x_0, u_0) \mid I_0 + \mathbb{E}_{z_1}\{J_1(z_0, z_1, u_0) \mid I_0, u_0\} \right\}\\
&= \min \left\{ 2 \cdot P(x_0 = \bar{P} \mid I_0) + \mathbb{E}_{z_1}\{J_1(I_0, z_1, C) \mid I_0, C\}, \right.\\
&\left. 1 + \mathbb{E}_{z_1}\{J_1(I_0, z_1, S) \mid I_0, S\} \right\}
\end{aligned}
$$

4.4 Sufficient Statistics

The drawback for the DP algorithm above is that the information vector grows with k. Fortunately, sometimes we can find equivalent representations of the information contained in the observations that provide a significant reduction of the data to remember.

Definition 4.1 (Sufficient Statistic). The sufficient statistic is a function $S_k(I_k)$, which may be (greatly) smaller than the dimension of I_k, but summarize all essential content of I_k for estimation or control purpose.

Formally, consider a random variable X with discrete values and probability distribution P_θ depending on an unknown parameter

θ. A statistic S is sufficient for θ if

$$P_\theta(X = x \mid S = t) = f(x,t), \forall x \in \mathcal{X}$$

Theorem 4.1 (Fisher Factorization Theorem)**.** Given a family of distrbutions $\{P_\theta\}$ such that we have

$$\mathrm{d}P_\theta(x) = f(\theta, x)\,\mathrm{d}\mu(x), \text{ for some fixed measure } \mu.$$

Suppose that X is a discrete random variable and $f(\theta, x)$ denotes pmf (the probability for $\{X = x\}$), and μ is the counting measure. Then S is a sufficient statistics for θ if we have

$$f(\theta, x) = G(\theta, S(x))h(x),$$

for some functions G and h.

Example 4.1. Consider the order statistics $S(X_1, \dots, X_n) = (X_{(1)}, \dots, X_{(n)})$, where $X_{(i)}$ denotes the i-th smallest value in the sample. Note that T is sufficient if X_i's are i.i.d., since the joint pdf can be factorized as

$$f(x_1, \dots, x_n) = \prod_{i=1}^{n} f(x_i) = \prod_{i=1}^{n} f(x_{(i)})$$

Application of sufficient statistics in DP Since our observations form the information vector I_k, we are looking for a stiatistics $S_k(I_k)$ for each k such that (4.3c) can be reformuated as

$$\min_{u_k \in U_k} H_k(S_k(I_k), u_k)$$

for some function H_k. The optimal control policy is given as a function of $S_k(I_k)$:

$$\mu_k^*(I_k) = \bar{\mu}_k(S_k(I_k))$$

the cost-to-go function is given by:

$$\begin{aligned} J_k(I_k) = \min_{u_k \in U_k(I_k) = \{C,S\}} &\{g(P, u_k) \cdot p(x_k = P \mid I_k, u_k) \\ &+ g(\bar{P}, u_k) \cdot p(x_k = \bar{P} \mid I_k, u_k) \\ &+ \mathbb{E}_{z_{k+1}}\{J_{k+1}(I_k, u_k, z_{k+1}) \mid I_k, u_k\}\}, \ k = 0, 1 \end{aligned}$$

and $J_2(I_2) = 0$.

1. Stage 1:
$$J_1(I_1) = \min\{2 \cdot p(x_1 = \bar{P} \mid I_1), 1\}$$
Note that
$$\begin{aligned} p(x_1 = \bar{P} \mid I_1) &= p(x_1 = \bar{P} \mid z_0, z_1, u_0) \\ &= \frac{p(x_1, z_1 \mid z_0, u_0)}{p(z_1 \mid z_0, u_0)} \\ &= \frac{p(z_1 \mid x_1, z_0, u_0) p(x_1 \mid z_0, u_0)}{p(z_1 \mid z_0, u_0)} \\ &= \frac{p(z_1 \mid x_1) p(x_1 \mid z_0, u_0)}{p(z_1 \mid z_0, u_0)} \end{aligned}$$

2. Stage 0:
$$\begin{aligned} J_0(I_0) = \min[&2p(x_0 = \bar{P} \mid I_0, C) + \mathbb{E}_{z_1}\{J_1(I_0, z_1, C) \mid I_0, C\}, \\ &1 + \mathbb{E}_{z_1}\{J_1(I_0, z_1, S) \mid I_0, S\}] \end{aligned}$$

4.4.1 Conditional State Distribution serves as a Sufficient Statistic

In this section we asume that the probability for observation disturbance v_{k+1} depends explicitly only on the immediately preceding state, control, and system disturbance x_k, u_k, ω_k.

Theorem 4.2. For any k and I_k, we have
$$J_k(I_k) = \min_{u_k \in U_k} H_k(P_{x_k|I_k}, u_k) = \bar{J}_k(P_{x_k|I_k})$$
for some appropriate functions H_k and $\bar{J}_k$.

Remark 4.1. As a result, the optimal control can be computed as
$$\mu_k(I_k) = \bar{\mu}_k(P_{x_k|I_k}),\ k = 0, 1, \ldots, N-1$$
which provides a decomposition of the controller in two parts:

1. Estimator: uses the measurement z_k and control u_{k-1} to generate probability distribution $P_{x_k|I_k}$
2. Actuator: generates a control u_k to the system, which is a function of the probability distribution $P_{x_k|I_k}$

4.4.2 The conditional state distribution recursion

This section aims to justify the recursion

$$P_{x_{k+1}|I_{k+1}} = \Phi_k(P_{x_k|I_k}, u_k, z_{k+1})$$

Propagation Step We compute the conditional probability for X_{k+1} according to our knowledge of the control input u_k and information vector I_k:

$$\begin{aligned} P(X_{k+1} \mid I_k, u_k) &= \int P(X_{k+1} \mid I_k, u_k, x_k) \cdot \mathrm{d}P(x_k \mid I_k, u_k) \\ &= \int P(X_{k+1} \mid u_k, x_k)\, \mathrm{d}P(x_k \mid I_k) \end{aligned}$$

Here $P(x_k \mid I_k, u_k) = P(x_k \mid I_k)$ since the control u_k is a function of I_k; the distribution $P(X_k \mid I_k)$ is assumed known from the previous step; and $P(X_{k+1} \mid u_k, x_k)$ can be computed directly from the distribution of the system disturbance ω_k.

Update Step We take the new measurement z_{k+1} into account:

$$\begin{aligned} P(X_{k+1} \mid I_{k+1}) &= P(X_{k+1} \mid I_k, u_k, z_{k+1}) \\ &= \mathcal{Z} P(z_{k+1} \mid X_{k+1}, I_k, u_k) \cdot P(X_{k+1} \mid I_k, u_k) \\ &= \mathcal{Z} P(z_{k+1} \mid X_{k+1}, u_k) \cdot P(X_{k+1} \mid I_k, u_k) \end{aligned}$$

where $\mathcal{Z}$ denotes the normalization factor

$$\mathcal{Z} = \left(\int P(z_{k+1} \mid x_{k+1}, u_k)\, \mathrm{d}P(x_{k+1} \mid I_k, u_k) \right)^{-1},$$

and the distribution $P(z_{k+1} \mid X_{k+1}, u_k)$ can be computed directly from the distribution of measurement noise v_{k+1}.

Perfect State Information Reduction In particular, the new DP formulation is given as:

$$\bar{J}_k(P_{x_k|I_k}) = \min_{u_k \in U_k} \left[\mathbb{E}_{x_k,\omega_k,z_{k+1}} \left\{ g_k(x_k,u_k,\omega_k) + \bar{J}_{k+1}(\Phi_k(P_{x_k|I_k},u_k,z_{k+1})) \middle| I_k, u_k \right\} \right], \quad k = 0,\dots,N-2$$

$$\bar{J}_{N-1}(P_{x_{N-1}|I_{N-1}}) = \min_{u_k \in U_k} \left[\mathbb{E}_{x_{N-1},\omega_{N-1}} \left\{ g_{N-1}(x_{N-1},u_{N-1},\omega_{N-1}) + g_N(f_{N-1}(x_{N-1},u_{N-1},\omega_{N-1})) \middle| I_{N-1}, u_{N-1} \right\} \right]$$

The optimal cost is given by:

$$J^* = \mathbb{E}_{z_0}\{\bar{J}_0(P_{x_0|z_0})\}$$

4.5 Linear Systems and Quadratic Cost

- Dynamics of the system:

$$x_{k+1} = A_k x_k + B_k u_k + \omega_k,\ k = 0,\dots,N-1$$

where the initial state x_0 is random with known distribution.

- Measurement Equation:

$$z_k = C_k x_k + v_k,\ k = 0,\dots,N-1$$

where $z_k \in \mathbb{R}^s, x_k \in \mathbb{R}^n, C_k \in \mathbb{R}^{s\times n}$.

- The noise in the dynamics, say ω_k, are assumed to be independent, and independent of x_0, with zero mean and finite covariance
- The observation noise v_k are independent, and independent from ω_k and x_0 as well, with known distribution.
- The total cost is quadratic:

$$\mathbb{E}\left\{ x_N^{\mathrm{T}} Q_N x_N + \sum_{k=0}^{N-1} (x_k^{\mathrm{T}} Q_k x_k + u_k^{\mathrm{T}} R_k u_k) \right\} \tag{4.4}$$

From the DP equation (4.3e), we imply

$$\begin{aligned} J_{N-1}(x_{N-1}) = \min_{u_{N-1}\in U_{N-1}} \Big[& \mathbb{E}_{x_{N-1},\omega_{N-1}} x_{N-1}^{\mathrm{T}} Q_{N-1} x_{N-1} + u_{N-1}^{\mathrm{T}} R_{N-1} u_{N-1} \\ & + (A_{N-1}x_{N-1} + B_{N-1}u_{N-1} + \omega_{N-1})^{\mathrm{T}} \\ & \cdot Q_N (A_{N-1}x_{N-1} + B_{N-1}u_{N-1} + \omega_{N-1}) \Big] \qquad (4.5a) \\ & = \mathbb{E}_{x_{N-1}} \left\{ x_{N-1}^{\mathrm{T}} (A_{N-1}^{\mathrm{T}} Q_N A_{N-1} + Q_{N-1}) x_{N-1} \,\middle|\, I_{N-1} \right\} \\ & + \mathbb{E}_{\omega_{N-1}} \{ \omega_{N-1}^{\mathrm{T}} Q_N \omega_{N-1} \} \\ & + \min_{u_{N-1}\in U_{N-1}} \Big\{ u_{N-1}^{\mathrm{T}} (B_{N-1}^{\mathrm{T}} Q_N B_{N-1} + R_{N-1}) u_{N-1} \\ & + 2\mathbb{E}\{x_{N-1} \mid I_{N-1}\}^{\mathrm{T}} A_{N-1}^{\mathrm{T}} Q_N B_{N-1} u_{N-1} \Big\} \qquad (4.5b) \end{aligned}$$

Therefore, the optimal policy for the last stage is given by:

$$\begin{aligned} u_{N-1}^{*} &= -M_{N-1}^{-1} \ell_{N-1} \\ \text{where } M_{N-1} &= B_{N-1}^{\mathrm{T}} Q_N B_{N-1} + R_{N-1} \\ \ell_{N-1} &= B_{N-1}^{\mathrm{T}} Q_N A_{N-1} \mathbb{E}\{x_{N-1} \mid I_{N-1}\} \end{aligned}$$

Theorem 4.3. For all k, we have

$$J_k^{*}(I_k) = \mathbb{E}_{x_k}\{x_k^{\mathrm{T}} K_k x_k \mid I_k\} + \sum_{j=k+1}^{N-1} \mathbb{E}_{x_j}\{e_j^{\mathrm{T}} P_j e_j \mid I_k\} + \sum_{j=k+1}^{N-1} \mathbb{E}_{\omega_j}\{\omega_j^{\mathrm{T}} K_{j+1} \omega_j\} \qquad (4.6a)$$

where

$$e_k := x_k - \mathbb{E}_{x_k}\{x_k \mid I_k\} \qquad (4.6b)$$

and matrices K_k, P_k are given recursively by the Riccati Equation.

Note that u_k does not influence the future error terms $e_{k+1}, \dots, e_{N-1}$ at all! This is because of the linearity of the system and observation equations, and that the noises are independent of the control inputs. However, in general this situation may not necessarily hold.

Lemma 4.4. For all k, there is a function M_k such that

$$e_k = M_k(x_0, \omega_{0:k-1}, v_{0:k})$$

independentof the policy being used.

Proof. Consider two systems, one driven by some control inputs, and the other with zero control inputs:

$$\begin{cases} x_{k+1} = A_k x_k + B_k u_k + \omega_k \\ z_k = C_k x_k + v_k \end{cases} \qquad \begin{cases} \tilde{x}_{k+1} = A_k \tilde{x}_k + \omega_k \\ \tilde{z}_k = C_k \tilde{x}_k + v_k \end{cases}$$

Assume that $x_0 = \tilde{x}_0$, and denote the information vector for two systems as

$$I_k = \{z_{0:k}, u_{0:k-1}\}, \quad \tilde{I}_k = \{\tilde{z}_{0:k}\}$$

Define the transition matrix

$$\Phi(k, \ell) = \begin{cases} A_{k-1} \cdots A_\ell, \ k > \ell \geq 0 \\ I, \ k = \ell \end{cases}$$

which follows that

$$x_k = \Phi(k,0)x_0 + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1) B_\ell u_\ell + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1)\omega_\ell$$

$$\tilde{x}_k = \Phi(k,0)x_0 + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1)\omega_\ell$$

Moreover,

$$\begin{aligned} \mathbb{E}\{x_k \mid I_k\} &= \Phi(k,0)\mathbb{E}\{x_0 \mid I_k\} + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1) B_\ell \mathbb{E}\{u_\ell \mid I_k\} \\ &\quad + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1)\mathbb{E}\{\omega_\ell \mid I_k\} \\ &= \Phi(k,0)\mathbb{E}\{x_0 \mid I_k\} + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1) B_\ell u_\ell \\ &\quad + \sum_{\ell=0}^{k-1} \Phi(k, \ell+1)\mathbb{E}\{\omega_\ell \mid I_k\} \end{aligned}$$

Therefore, we immediately get

$$\begin{aligned} x_k - \mathbb{E}\{x_k \mid I_k\} &= \Phi(k,0)(x_0 - \mathbb{E}\{x_0 \mid I_k\}) + \sum_{\ell=0}^{k-1} \Phi(k,\ell+1)(\omega_\ell - \mathbb{E}\{\omega_\ell \mid I_k\}) \\ &= \tilde{x}_k - \mathbb{E}\{\tilde{x}_k \mid I_k\} \end{aligned}$$

It suffices to show that $\mathbb{E}\{\tilde{x}_k \mid I_k\} = \mathbb{E}\{\tilde{x}_k \mid \tilde{I}_k\}$, or more precisely, $\text{Information}(I_k) = \text{Information}(\tilde{I}_k)$. Note that

$$\tilde{z}_k = z_k - \sum_{\ell=0}^{k-1} \Phi(k,\ell+1) C_\ell B_\ell u_\ell,$$

and therefore this is true.

□

Now we prove the Theorem (4.3). Applying (4.3d) and Lemma (4.4), we have

$$\begin{aligned} J_k(I_k) = \mathbb{E}_{x_k}\{x_k^{\mathrm{T}} Q_k x_k \mid I_k\} + \sum_{j=k+1}^{N-1} \mathbb{E}_{\omega_j}\{\omega_j^{\mathrm{T}} K_{j+1} \omega_j\} + \sum_{j=k+1}^{N-1} \mathbb{E}\{e_j^{\mathrm{T}} P_j e_j \mid I_k\} \\ + \min_{u_k \in U_k} \left\{ u_k^{\mathrm{T}} R_k u_k + \mathbb{E}\{x_{k+1}^{\mathrm{T}} K_{k+1} x_{k+1} \mid I_k\} \right\} \end{aligned} \tag{4.7}$$

Now simplify the minimization term:

$$\begin{aligned} &u_k^{\mathrm{T}} R_k u_k + \mathbb{E}\{(A_k x_k + B_k u_k + \omega_k)^{\mathrm{T}} K_{k+1} (A_k x_k + B_k u_k + \omega_k) \mid I_k\} \\ &= u_k^{\mathrm{T}} (R_k + B_k^{\mathrm{T}} K_{k+1} B_k) u_k + 2 u_k^{\mathrm{T}} B_k^{\mathrm{T}} K_{k+1} A_k \mathbb{E}\{x_k \mid I_k\} \\ &+ \mathbb{E}\{x_k^{\mathrm{T}} A_k^{\mathrm{T}} K_{k+1}^{\mathrm{T}} A_k x_k \mid I_k\} + \mathbb{E}_{\omega_k}\{\omega_k^{\mathrm{T}} K_{k+1} \omega_k\} \end{aligned}$$

Therefore we imply the optimal policy for each stage:

$$\begin{aligned} \mu_k^*(I_k) &= L_k \mathbb{E}\{x_k \mid I_k\} \\ \text{with } L_k &= -(R_k + B_k^{\mathrm{T}} K_{k+1} B_k)^{-1} B_k^{\mathrm{T}} K_{k+1} A_k \end{aligned}$$

where K_k, P_k are given recursively by the Riccati Equation.

Substituting $\mu_k^*(I_k)$ into $J_k(I_k)$, we imply

$$J_k(I_k) = \mathbb{E}_{x_k}\{x_k^{\mathrm{T}}(Q_k + A_k^{\mathrm{T}}K_{k+1}A_k)x_k \mid I_k\} + \sum_{j=k+1}^{N-1} \mathbb{E}_{\omega_j}\{\omega_j^{\mathrm{T}}K_{j+1}\omega_j\}$$
$$+ \sum_{j=k+1}^{N-1} \mathbb{E}\{e_j^{\mathrm{T}}P_je_j \mid I_k\} - \mathbb{E}\{x_k \mid I_k\}^{\mathrm{T}}P_k\mathbb{E}\{x_k \mid I_k\} + \sum_{j=k+1}^{N-1} \mathbb{E}\{e_j^{\mathrm{T}}P_je_j \mid I_k\}$$

Also, applying the identity

$$\mathbb{E}\{x_k^{\mathrm{T}}Mx_k \mid I_k\} = \mathbb{E}\{x_k \mid I_k\}^{\mathrm{T}}M\mathbb{E}\{x_k \mid I_k\} + \mathbb{E}\{e_k^{\mathrm{T}}Me_k \mid I_k\} \quad (4.8)$$

with $M = P_k$ into the forth term, we obtain the desired result.

Remark 4.2. The optimal control problem indeed decomposes into two successive parts:

1. Estimator: estimate the conditional mean $\mathbb{E}\{x_k \mid I_k\}$, i.e., the estimate $\hat{x_k}$ of x_k, given I_k, which minimizes $\mathbb{E}\{\|x_k - \hat{x}_k\|^2 \mid I_k\}$
2. Actuator: $u_k = L_k\mathbb{E}\{x_k \mid I_k\}$

Steady State Policy Consider the case where the system and measurement equations and the disturbance statistics are stationary. When the horizon tends to infinite, the optimal controller tends to the steady state policy

$$\mu^*(I_k) = L\hat{x}_k$$

where

$$L = -(R + B^{\mathrm{T}}KBB)^{-1}B^{\mathrm{T}}KA,$$

where K is the unique positive semi-definite symmetric equation of the algebraic Riccatic equation

$$K = A^{\mathrm{T}}(K - KB(R + B^{\mathrm{T}}KB)^{-1}B^{\mathrm{T}}K)A + Q$$

4.6 Finite State Space Problems

The imperfect state problem is analytically solvable in two main cases. The first case is the linear quadratic controller discussed before, and the other concerns models with a finite state space X. In both cases, we should have an efficient way for computing the conditional distribution $P_{x_k|I_k}$.

Several assumptions are made for general finite-state space case:

- Observation and control space are also finite
- The state transition provided a fixed control input is stationary
- The observation noise depends only on the current state and the proceding control.

Problem Formulation Suppose in each stage we have finite number n of possible states $X_k = X = \{1, \dots, n\}, \forall k$. The distribution of the state x_k, given the observation I_k, is given by a vector of n real numbers

$$p_k = [p_k^1, \dots, p_k^n]^{\mathrm{T}}, \quad \text{where } p_k^i = P(x_k = i \mid I_k)$$

with p_k belonging to the $(n-1)$-simplex

$$p_k \in \Delta_{n-1} = \left\{ [p_k^1, \dots, p_k^n]^{\mathrm{T}} \middle| p^i \geq 0, \forall i, \sum_{i=1}^{n} p^i = 1 \right\}$$

- We assume that p_0 is given, and p_k can be obtained using recursion:

$$p_{k+1} = \Phi(p_k, u_k, z_{k+1}).$$

- Now the terminal cost-go-go function can be written as

$$\bar{J}_N(p_N) = p_N^{\mathrm{T}} G_N,$$

  where $G_N := [g_N^1, \dots, g_N^n]^{\mathrm{T}}$ denotes costs for each possible state at stage N. Similarly, define the n-dimensional vector $g_k(u_k)$ for each control u_k at stage k by

$$\begin{aligned} g_k(u_k) &= [g_k^1(u_k), \dots, g_k^n(u_k)] \\ \text{where} \quad g_k^i(u_k) &= \mathbb{E}_{\omega_k}\{g_k(x_k, u_k, \omega_k) \mid x_k = i\} \end{aligned}$$

  Thus the cost-go-go function for stages $k \leq N-1$ is given by:

$$\bar{J}_k(p_k) = \min_{u_k \in U_k} \left\{ p_k^{\mathrm{T}} g_k(u_k) + \mathbb{E}_{z_{k+1}} \left[\bar{J}_{k+1}(\Phi_k(p_k, u_k, z_{k+1})) \middle| p_k, u_k \right] \right\}.$$

- For a given control u, define

$$p_{ij}(u) = P(x_{k+1} = j \mid x_k = i, u_k = u)$$

  As a result, the conditional distribution of z_{k+1} given p_k and u_k can be computed as

$$p(z_{k+1} = z \mid p_k, u_k) = \sum_{i=1}^{n} p_k^i \sum_{j=1}^{n} p_{ij}(u_k) P(z_{k+1} = z \mid x_{k+1} = j, u_k)$$

Theorem 4.5. The cost-to-go functions $\bar{J}_k$ in the DP algorithm are piecewise linear and concave.

Example 4.2 (Machine Repair)**.** We follow the machine repair example discussed before. Denote

$$p_1 = P(x_1 = \bar{P} \mid I_1), \quad p_0 = P(x_0 = \bar{P} \mid I_0)$$

Then compute

$$p_1 = \Phi_0(p_0, u_0, z_0)$$

and derive $\bar{J}_1(p_1)$ and $\bar{J}_0(p_0)$. Therefore,

$$J^* = \mathbb{E}_{z_0}\{\bar{J}_0(P_{x_0|z_0})\} = \sum_i P(z_0 = z_0^i)\bar{J}_0(P_{x_0|z_0^i})$$

5

Suboptimal Control

Curse of Dimensionality In most cases, the modeling step leads to state-space explosion. Moreover, some problem data is unknown until shortly before the control is needed, thus seriously constraining the amount of time available for the DP computation.

5.1 Certainty Equivalence Control

The certainty equivalent controller (CEC) is a heuristic controller that applies the certainty equivalence and separation principles of linear quadratic control theory, even if the assumption does not formally hold for the problem. Given a finite horizon problem with horizon N, and possibly imperfect state information, the heuristic is given for computing the cost-go-go function at stage k as follows:

1. Given the information vector I_k, compute the state estimate $\bar{x}_k(I_k)$, e.g., the conditional mean estimate $\bar{x}_k(I_k) := \mathbb{E}\{x_k \mid I_k\}$

2. Fix the process disturbances $\{w_i\}_{i \geq k}$ at some typical deterministic value, e.g., $\bar{\omega}_i(x_i, u_i) = \mathbb{E}\{\omega_i \, mid x_i, u_i\}$. If we know that the ω_i are i.i.d. with zero mean, we just fix them to 0.

3. Solve the *deterministic perfect state information* optimal control problem

$$\begin{array}{ll} \min & g_N(x_N) + \sum_{j=k}^{N} g_i(x_i, u_i, \bar{\omega}_i(x_i, u_i)) \\ \text{subject to} & \text{initial condition } x_k = \bar{x}_k(I_k) \\ & \text{constraints } u_i \in U_i,\ i = k : N-1 \\ & \text{dynamics } x_{i+1} = f_i(x_i, u_i, \bar{\omega}_i(x_i, u_i)),\ i = k : N-1 \end{array}$$

4. Use only the first element $\bar{u}_k = \bar{\mu}_k(I_k)$ in the control sequence found. We land in a new state x_{k+1} an and make a new observation z_{k+1}. Now repeat from step 1.

Remark 5.1. The step (1) is unnecessary once we have perfect state information; The deterministic optimization probelm in step (3) must be solved at each stage k; Under some good structures these problems can be solved efficiently such as convexity; the implementation of the CEC requires no storage of the type required for the optimal feedback controller.

An Offline Alternative for CEC We can solve the optimal control problems by solving these problems as a priori off-line. In particular, we use DP to solve the problem

$$\min \quad g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), \bar{\omega}_k(x_k, u_k)) \tag{5.1a}$$

$$\text{subject to } x_{k+1} = f_k(x_k, \mu_k(x_k), \bar{\omega}_k(x_k, u_k)),\ \mu_k(x_k) \in U_k, k \geq 0 \tag{5.1b}$$

w.r.t. a feedback policy, denoted as $\pi^d = \{\mu_0^d, \dots, \mu_{N-1}^d\}$. Then we store the functions $\mu_0^d, \dots, \mu_{N-1}^d$. In step 3, there is no need to solve any optimization problem any more, and we just apply $\bar{\mu}_k(I_k) = \mu_k^d(\bar{x}_k(I_k))$ in step 4.

Remark 5.2. The CEC is optimal for linear quadratic problems, as shown in previous chapter. However, it's possible to construct examples where the CEC performs strictly worse than the optimal open-loop controller.

5.1.1 Variantions of CEC

CEC with Heuristics

w.l.o.g., assume we are doing the perfect state information problem, and we aim to design some heuristic to find a *suboptimal* control sequence for the problem $\{\bar{u}_k, \dots, \bar{u}_{N-1}\}$ to the problem

$$\begin{array}{ll} \min & g_N(x_N) + \sum_{j=k}^{N} g_i(x_i, u_i, \bar{\omega}_i(x_i, u_i)) \\ \text{subject to} & \text{initial condition } x_k \text{ is given} \\ & \text{constraints } u_i \in U_i,\ i = k : N-1 \\ & \text{dynamics } x_{i+1} = f_i(x_i, u_i, \bar{\omega}_i(x_i, u_i)),\ i = k : N-1 \end{array}$$

The idea is to use minimization over the first control u_k and use the heuristic only for the remaining stages $k+1 : N-1$. In particular, we find the control $\bar{u}_k$ at stage k such that

$$\bar{u}_k = \arg \min_{u_k \in U_k(x_k)} g_k(x_k, u_k, \bar{\omega}_k(x_k, u_k)) + H_{k+1}(f_k(x_k, u_k, \bar{\omega}_k(x_k, u_k)))$$

where H_{k+1} is the heuristic cost-go-go function, i.e., $H_{k+1}(x_{k+1})$ is the cost-go-go function starging from a state x_{k+1} using heuristic, by assuming that the future disturbances will be equal to their typical values $\bar{\omega}_i(x_i, u_i), i = k+1 : N-1$. Note that $H_{k+1}(x_{k+1})$ is an approximation of the optimal cost-to-go function $J^*_{k+1}(x_{k+1})$

Partially Stochastic Certainty Equivalent Control

Sometimes we take into account the stochastic nature of disturances, but still treat these problems as perfect state information:

1. First generate the optimal policy $\{\mu_0^p(x_0), \dots, \mu_{N-1}^p(x_{N-1})\}$ for the stochastic perfect state information problem

$$\begin{array}{ll} \min & \mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), \omega_k) \right\} \\ \text{subject to} & \text{constraints } \mu_k(x_k) \in U_k,\ k = 0 : N-1 \\ & \text{dynamics } x_{k+1} = f_k(x_k, u_k, \omega_k),\ k = 0 : N-1 \end{array}$$

2. Then apply the control input at stage k as

$$\bar{\mu}_k(I_k) = \mu_k^p(\bar{x}_k(I_k))$$

Example 5.1 (Aloha). In previous perfect state information chapter we have shown that the optimal policy is

$$\mu_k(x_k) = \min\left\{\frac{1}{x_k}, 1\right\}$$

Therefore, the partially stochastic CEC policy is given by:

$$\bar{\mu}_k(I_k) = \min\left\{\frac{1}{\bar{x}_k(I_k)}, 1\right\}$$

Example 5.2 (Systems with unknown parameters). Sometimes the system parameters are not known exactly or change over time. One alternative is to formulate the stochastic control problem such that the unknown parameters are dealt with directly. In particular, it can be transformed into imperfect state information problem using state augmentation:

1. Suppose that the system equation is of the form

   $$x_{k+1} = f_k(x_k, \theta, u_k, \omega_k)$$

   where θ is a vector of unknown parameters with a given priori probability distribution.

2. We introduce an additional state variable $y_k = \theta$, and obtain a new system equation

   $$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, \theta, u_k, \omega_k) \\ y_k \end{pmatrix}$$

   Or equivalently, $\tilde{x}_{k+1} = \tilde{f}_k(\tilde{x}_k, u_k, \omega_k)$. The initial condition is given by $\tilde{x}_0 = (x_0, \theta)$. The resulting problem fits into our framework.

3. Since θ is unobservable, we are faced with a problem of imperfect state information, even if we know the state x_k exactly. Therefore, typically an optimal solution cannot be found, but suboptimal solution could be found using partially stochastic CEC.

4. Suppose that for a fixed parameter vector θ, we are able to derive the optimal policy

   $$\{\mu_0^*(I_0, \theta), \ldots, \mu_{N-1}^*(I_{N-1}, \theta)\}$$

Then the partially stochastic CEC policy takes the form

$$\bar{\mu}_k(I_k) = \mu_k^*(I_k, \bar{\theta}_k(I_k))$$

where $\bar{\theta}_k(I_k)$ is some estimate of θ, based on the information vector I_k.

5.1.2 Caution, Probing, and Dual Control

1. Caution: fully understand the system dynamics

2. Probing: obtain lower objective cost

Example 5.3 (Control Objective & Parameter Estimation). Consider the perfect state information system setting

$$\begin{aligned} \text{linear scalar system} \quad & x_{k+1} = x_k + bu_k + \omega_k, \ \ k = 0 : N-1 \\ \text{Quadratic Terminal Cost} \quad & \mathbb{E}\{x_N^2\} \\ \text{disturbance distribution} \quad & \omega_k \sim \mathcal{N}(0, \sigma_\omega^2) \end{aligned}$$

except that b is unknown, with a prior probability distribution of b:

$$b \sim \mathcal{N}(\bar{b}, \sigma_b^2)$$

1. When $N = 1$, we may verify that

$$\mathbb{E}\{x_1^2\} = x_0^2 + 2\bar{b}x_0u_0 + (\bar{b}^2 + \sigma_b^2)u_0^2 + \sigma_\omega^2$$

and the minimizer of u_0 is

$$u_0 = -\frac{\bar{b}}{\bar{b}^2 + \sigma_b^2} x_0$$

with the optimal cost

$$\frac{\sigma_b^2}{\bar{b}^2 + \sigma_b^2} x_0^2 + \sigma_\omega^2$$

The optimal control here is cautious in that the optimal $|u_0|$ decreases as the uncertainty in b increases.

2. When $N = 2$, the optimal cost-to-go function at stage 1 is

$$J_1(I_1) = \frac{\sigma_b^2(1)}{(\bar{b}(1))^2 + \sigma_b^2(1)} x_1^2 + \sigma_\omega^2 \tag{5.2}$$

where $I_1 = (x_0, u_0, x_1)$ denotes the information vector, and

$$\bar{b}(1) = \mathbb{E}\{b \mid I_1\}, \quad \sigma_b^2(1) = \mathbb{E}\{(b - \bar{b}(1))^2 \mid I_1\}$$

where $\bar{b}(1)$ and $\sigma_b^2(1)$ can be obtained from the equation $x_1 = x_0 + bu_0 + \omega_0$. The formulat for $\sigma_b^2(1)$ is given by

$$\sigma_b^2(1) = \frac{\sigma_b^2 \sigma_\omega^2}{u_0^2 \sigma_b^2 + \sigma_\omega^2}$$

Therefore, to achieve small error variance$\sigma_b^2(1)$, we should apply a control u_0 that is large in absolute value. However, when $|u_0|$ is large, then $|x_1|$ will also be large, which is not desirable in view of Eq. (5.2). Therefore, there is a tradeoff between the control objective and the parameter estimation.

5.1.3 Two Phase Control and Identifiability

Return to example (5.2). We separate the control process into two phases, a *parameter identification phase* and a *control phase.*

One drawback is that the information gathered during the identification phase is not used to adjust the control law until the begining of the second phase. Furthermore, it's not always easy to determine when to terminate one phase and start the other.

Another drawback is that the control process may make some of the unknown parameters *invisible* to the identification phase, which is best explained by the example below:

Example 5.4. Consider the perfect state information system setting

linear scalar system	$x_{k+1} = ax_k + bu_k + \omega_k,\ k = 0 : N-1$
Quadratic Terminal Cost	$\mathbb{E}\left\{\sum_{k=1}^{N} x_k^2\right\}$
disturbance distribution	$\mathbb{E}\omega_k = 0$

except that the parameters a and b is unknown. For fixed a and b, the control law is

$$\mu_k^*(x_k) = -\frac{a}{b}x_k.$$

Consider the two-phase method. During the first phase the control law

$$\tilde{\mu}_k(x_k) = \gamma x_k := -\frac{\bar{a}}{\bar{b}}x_k \tag{5.3}$$

is used. At the end of the first phase, the control law is changed into

$$\bar{\mu}_k(x_k) = -\frac{\hat{a}}{\hat{b}}x_k, \tag{5.4}$$

where $\hat{a}$ and $\hat{b}$ are the estimates obtained from the identification process. However, with the control law (5.3), the closed-loop system is changed into

$$x_{k+1} = (a + b\gamma)x_k + \omega_k,$$

i.e., the identification process can at best identify the value of $(a + b\gamma)$ but not the values of both a and b, i.e., we cannot identify pairs (a_1, b_1) and (a_2, b_2) once $a_1 + b_1\gamma = a_2 + b_2\gamma$,

1. One way to rescue is to add an additional known input δ_k into the control (5.3):

   $$\tilde{\mu}_k(x_k) = \gamma x_k + \delta_k$$

   Then the closed-loop system becomes

   $$x_{k+1} = (a + b\gamma x_k) + b\delta_k + \omega_k,$$

   and the knowledge for $\{x_k\}$ and x_{k+1} makes possible to identify $(a + b\gamma)$ and b.

2. The second way is to change the structure of the system by introducing a one-unit delay in the control feedback. Therefore, we consider the control of the form

   $$u_k = \hat{\mu}_k(x_{k-1}) = \gamma x_k$$

   and then the closed-loop system becomes

   $$x_{k+1} = ax_k + b\gamma x_{k-1} + \omega_k$$

   Given γ and $\{x_k\}$, it's possible to identify both a and b. However, introducing a control delay makes the system less responsive to control.

5.1.4 Certanity Equivalent Control and Identifiability

The certainty equivalent control approach incorporated the parameter estimation process into the control law as they are generated, and they are treated as if they were the real values.

Consider the sytem $x_{k+1} = f_k(x_k, \theta, u_k, \omega_k)$ again. Suppose that for each possible value of θ, the control law $\pi^*(\theta) = \{\mu^*_{0:N-1}(\cdot, \theta)\}$ is optimal w.r.t. a certain cost $J_\pi(x_0, \theta)$. The suboptimal control used at time k is

$$\hat{\mu}_k(I_k) = \mu^*(x_k, \hat{\theta}_k)$$
$$\text{with} \quad \hat{\theta} \text{ is an estimate of } \theta \text{ based on } I_k$$

Unfortunately, the parameter estimates $\hat{\theta}_k$ may not necessarily converge to the true value θ, i.e., the CEC is not asymptotically optimal. Suppose for simplicity that the system is stationary with a prior transition probabilities $P\{x_{k+1} \mid x_k, u_k, \theta\}$ and that the control law used is stationary:

$$\hat{\mu}_k(I_k) = \mu^*(x_k, \hat{\theta}_k), \quad k = 0, 1, \ldots$$

There are three systems of interest here:

1. The system *believed* to be true:
$$P\{x_{k+1} \mid x_k, \mu^*_k(x_k, \hat{\theta}_k), \hat{\theta}_k\}$$
2. Underlying true closed-loop system:
$$P\{x_{k+1} \mid x_k, \mu^*(x_k, \hat{\theta}_k), \theta\}$$
3. The optimal cost-loop system regarding to the underlying system:
$$P\{x_{k+1} \mid x_k, \mu^*(x_k, \theta), \theta\}$$

It's possible that

1. $\hat{\theta}_k$ does not converge into anything
2. $\hat{\theta}_k$ converges to a parameter $\hat{\theta} \neq \theta$: suppose that some $\hat{\theta} \neq \theta$ we have
$$P\{x_{k+1} \mid x_k, \mu^*(x_k, \hat{\theta}), \hat{\theta}\} = P\{x_{k+1} \mid x_k, \mu^*(x_k, \hat{\theta}), \theta\}$$
Then the identification process locks into a wrong parameter no matter how long information is collected.

5.2 Open-Loop Feedback Control

A variation of CEC, called open-loop feedback control, does not fix the process disturbances, but still solves at eacg stage an open-loop control problem. The computation is more complicated than step 3 in CEC, since it involves the computation of some expected values.

Assumption The observation disturbance v_{k+1} depends explicitly only on the immediate preceding state, control, and the system disturbance x_k, u_k, ω_k. As a result, $P_{x_k|I_k}$ is a sufficient statistic for the imperfect information state problem. Then the OLFC proceeds as follows at stage k: (question: this assumption is needed in CEC?)

1. Given I_k, compute $P_{x_k|I_k}$
2. Compute an optimal *open-loop* control sequence $\{\bar{u}_k, \dots, \bar{u}_{N-1}\}$ for the problem

$$\begin{array}{ll} \min & \mathbb{E}\left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, u_i, \omega_i) \middle| I_k \right\} \\ \text{s.t.} & x_{i+1} = f_i(x_i, u_i, \omega_i),\ u_i \in U_i,\ i \geq k \end{array}$$

As a result, $\bar{\mu}_k(I_k) = \bar{u}_k$, then go to step 1 again.

Remark 5.3. Different between OLFC and partially stochastic CEC? The partially stochastic CEC does not involve the expectation over x_k; the OLFC deals with the expectation over x_k. (right?)

A nice property of the OLFC is that it performs at least as well as an optimal open-loop policy, i.e., a sequence $\{u_0^*, u_1^*, \dots, u_{N-1}^*\}$ that is the optimal solution for

$$\begin{array}{ll} \min & \bar{J}(u_0, \dots, u_{N-1}) = \mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, \omega_k) \right\} \\ \text{subject to} & x_{k+1} = f_k(x_k, u_k, \omega_k),\ u_k \in U_k,\ k = 0, \dots, N-1 \end{array}$$

Proposition 5.1. The cost $J_{\bar{\pi}}$ corresponding to an OLFC satisfies

$$J_{\bar{\pi}} \leq J_0^*,$$

where J_0^* is the cost corresponding to an optimal open-loop policy.

Proof. • Let $\bar{\pi} = \{\bar{\mu}_0, \dots, \bar{\mu}_{N-1}\}$ be the OLFC, with the cost

$$J_{\bar{\pi}} = \mathbb{E}_{z_0}\{\bar{J}_0(I_0)\}, \tag{5.5a}$$

where $\bar{J}_0$ is obtained from the recusive algorithm:

$$\bar{J}_{N-1}(I_{N-1}) = \mathbb{E}_{x_{N-1},\omega_{N-1}}\Bigg\{ g_N(f_{N-1}(x_{N-1}, \bar{\mu}_{N-1}(I_{N-1}), \omega_{N-1})) + g_{N-1}(x_{N-1}, \bar{\mu}_{N-1}, \omega_{N-1}) \Bigg| I_{N-1} \Bigg\} \tag{5.5b}$$

$$\bar{J}_k(I_k) = \mathbb{E}_{x_k,\omega_k,v_{k+1}}\Bigg\{ g_k(x_k, \bar{\mu}_k(I_k), \omega_k) \tag{5.5c}$$

$$+ \bar{J}_{k+1}(I_k, h_{k+1}(f_k(\bar{\mu}_k(I_k), \omega_k), \bar{\mu}_k(I_k), v_{k+1}), \bar{\mu}_k(I_k)) \Bigg| I_k \Bigg\},$$

$$k = 0, \dots, N-1 \tag{5.5d}$$

where $h_k(x_k, u_{k-1}, v_k)$ denotes the measurement equation.

• Define the intermediate function $J_k^c(I_k)$ for $k = 0 : N-1$:

$$J_k^c(I_k) = \min_{u_i \in U_i, i=k,\dots,N-1} \mathbb{E}\Bigg\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, u_i, \omega_i) \Bigg| I_k \Bigg\}$$

Since we can write J_0^* as

$$J_0^* = \min_{u_0,\dots,u_{N-1}} \mathbb{E}_{z_0}\mathbb{E}\{\text{cost} \mid z_0\},$$

and $\mathbb{E}_{z_0}\{J_0^c(z_0)\}$ can be expanded as

$$\mathbb{E}_{z_0}\Bigg\{ \min_{u_0,\dots,u_{N-1}} \mathbb{E}\{\text{cost} \mid z_0\} \Bigg\},$$

and due to the inequality $\mathbb{E}\{\min(\cdot)\} \le \min(\mathbb{E}\{\cdot\})$, we have

$$\mathbb{E}_{z_0}\{J_0^c(z_0)\} \le J_0^*.$$

• Now we will show that

$$\bar{J}_k(I_k) \le J_k^c(I_k), \quad \forall I_k, k \tag{5.6}$$

Then by (5.5a) we imply

$$J_{\bar{\pi}} \leq .$$

We will show that (5.6) by induction. By definition,

$$\bar{J}_{N-1}(I_{N-1}) = J^c_{N-1}(I_{N-1}), \quad \forall I_{N-1}.$$

Assume that

$$\bar{J}_{k+1}(I_{k+1}) \leq J^c_{k+1}(I_{k+1}), \quad \forall I_{k+1} \tag{5.7a}$$

Then we imply

$$\begin{aligned}
\bar{J}_k(I_k) = \mathbb{E}_{x_k,\omega_k,v_{k+1}}\Bigg\{ & g_k(x_k, \bar{\mu}_k(I_k), \omega_k) \\
& + \bar{J}_{k+1}(I_k, h_{k+1}(f_k(x_k, \bar{\mu}(I_k), \omega_k), \bar{\mu}_k(I_k), v_{k+1}), \bar{\mu}_k(I_k)) \Bigg| I_k \Bigg\} \\
& \leq \mathbb{E}_{x_k,\omega_k,v_{k+1}}\Bigg\{ g_k(x_k, \bar{\mu}_k(I_k), \omega_k) \\
& + J^c_{k+1}(I_k, h_{k+1}(f_k(x_k, \bar{\mu}(I_k), \omega_k), \bar{\mu}_k(I_k), v_{k+1}), \bar{\mu}_k(I_k)) \Bigg| I_k \Bigg\} \\
& = \mathbb{E}_{x_k,\omega_k,v_{k+1}}\Bigg\{ \min_{u_{k+1:N-1}} \mathbb{E}\Bigg\{ g_k(x_k, \bar{\mu}_k(I_k), \omega_k) \\
& + \sum_{i=k+1}^{N-1} g_i(x_i, u_i, \omega_i) + g_N(x_N) \Bigg| I_{k+1} \Bigg\} \Bigg| I_k \Bigg\} \\
& \leq \min_{u_{k+1:N-1}} \mathbb{E}\Bigg\{ g_N(x_N) + g_k(x_k, \bar{\mu}_k(I_k), \omega_k) \\
& + \sum_{i=k+1}^{N-1} g_i(x_i, u_i, \omega_i) \Bigg| I_k \Bigg\} \\
& = J^c_k(I_k)
\end{aligned}$$

□

Partial Open-Loop Feedback Control The POLFC uses past measurement to cmompute $P_{x_k|I_k}$, but the control input is computed based on the fact that some of the measurements will be taken in the future, and the remaining measurements will not be taken.

5.3 Limited Lookahead Policies

The limited lookahead polies replies on an *approximation* $\tilde{J}_k(\cdot)$ of the cost-to-go function in the DP algorithm. For the perfect state problem, the *one-step looahead policy* takes at sstage k and the state x_k, the control $\bar{\mu}_k(x_k)$ that minimizes

$$\min_{u_k \in U_k(x_k)} \mathbb{E}\left\{ g_k(x_k, u_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, \omega_k)) \right\} \tag{5.8}$$

where $\tilde{J}_{k+1}$ is the approximation of J_{k+1} in the DP recusion step, with additional $\tilde{J}_N = g_N$. Similarly, we consider a two-step looahead polciy, where $\tilde{J}_{k+1}$ itself is obtained using a one-step lookahead

$$\begin{aligned} \tilde{J}_{k+1}(x_{k+1}) = \min_{u_{k+1} \in U_{k+1}(x_{k+1})} \mathbb{E}\Big\{ & g_{k+1}(x_{k+1}, u_{k+1}, \omega_{k+1}) \\ & + \tilde{J}_{k+2}(f_{k+1}(x_{k+1}, u_{k+1}, \omega_{k+1})) \Big\} \end{aligned}$$

where $\tilde{J}_{k+2}$ is some approximation of the cost-to-go function J_{k+2}. Moreover, the minimization over U_k for $\bar{\mu}_k(x_k)$ can be done approximately.

5.3.1 Performance Bounds for Limited Lookahead Policies

Notations Let $\bar{J}_k(x_k)$ denote the underlying policy evaluation for the limited lookahead policy $\{\bar{\mu}_0, \dots, \bar{\mu}_{N-1}\}$ (right?).

Proposition 5.2. Assume that for all x_k and k, we have

$$\min_{u_k \in \bar{U}_k(x_k)} \mathbb{E}\left\{ g_k(x_k, u_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, \omega_k)) \right\} \leq \tilde{J}_k(x_k) \tag{5.9a}$$

Then the cost-to-go function $\bar{J}_k$ corresponding to a one-step lookahead policy that uses $\tilde{J}_k$ and $\bar{U}_k(x_k)$ satisfy for all x_k and k,

$$\bar{J}_k(x_k) \leq \min_{u_k \in \bar{U}_k(x_k)} \mathbb{E}\left\{ g_k(x_k, u_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, \omega_k)) \right\} \tag{5.9b}$$

Proof. Introduce a new function

$$\hat{J}_k(x_k) = \min_{u_k \in \bar{U}_k(x_k)} \mathbb{E}\left\{ g_k(x_k, u_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, \omega_k)) \right\}, \tag{5.10}$$

with $\hat{J}_N = g_N$. It suffices to show that $\bar{J}_k(x_k) \leq \hat{J}_k(x_k), \forall x_k, k$.

We use the backward induction on k. Since we have $\bar{J}_N(x_N) = \hat{J}_N(x_N) = g_N(x_N)$ for all x_N, assume that $\bar{J}_{k+1}(x_{k+1}) \leq \hat{J}_{k+1}(x_{k+1}), \forall x_{k+1}$, which implies that

$$\begin{aligned}
\bar{J}_k(x_k) &= \mathbb{E}\{g_k(x_k, \bar{\mu}_k, \omega_k) + \bar{J}_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k))\} \\
&\leq \mathbb{E}\{g_k(x_k, \bar{\mu}_k, \omega_k) + \hat{J}_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k))\} \\
&\leq \mathbb{E}\{g_k(x_k, \bar{\mu}_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k))\} \\
&= \hat{J}_k(x_k)
\end{aligned}$$

Furthermore, $\bar{J}_k(x_k) \leq \hat{J}_k(x_k)$. □

Remark 5.4. Note that the cost-go of the one-step lookahead policy is no greater than the lookahead approximation on which it is based. This provides a good upper bound on $\bar{J}_k(x_k)$.

We can verify the critical assumption (5.9a) for few interesting special cases:

Example 5.5 (Rollout Algorithm). Suppose that $\tilde{J}_k(x_k)$ is the cost-to-go for some given (suboptimal) heuristic policy π. The resulting one-step lookahead function *restricted to* π is given by:

$$\tilde{J}_k(x_k) = \mathbb{E}\left\{ g_k(x_k, \mu_k(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \mu_k(x_k), \omega_k)) \right\}$$

which implies

$$\tilde{J}_k(x_k) \geq \min_{u_k \in \bar{U}_k(x_k)} \mathbb{E}\left\{ g_k(x_k, \mu_k(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \mu_k(x_k), \omega_k)) \right\}$$

Therefore, the rollout algorithm performs better than the heuristic on which it is based.

Example 5.6 (Rollout Algorithm with multiple Heuristics)**.** Suppose that $\tilde{J}_k(x_k)$ is the minimum of the cost-to-go functions corresponding to m heurstics:

$$\tilde{J}_k(x_k) = \min\{J_{\pi_1,k}(x_k), \dots, J_{\pi_m,k}(x_k)\}$$

By the DP algorithm,

$$J_{\pi_j,k}(x_k) = \mathbb{E}\left\{g_k(x_k, \mu_{j,k}(x_k), \omega_k) + J_{\pi_j,k+1}(f_k(x_k, \mu_{j,k}(x_k), \omega_k))\right\}$$

which implies

$$\begin{aligned} J_{\pi_j,k}(x_k) &= \mathbb{E}\left\{g_k(x_k, \mu_{j,k}(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \mu_{j,k}(x_k), \omega_k))\right\} \\ &\geq \min_{u_k \in \bar{U}_k(x_k)} \mathbb{E}\left\{g_k(x_k, \mu_{j,k}(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \mu_{j,k}(x_k), \omega_k))\right\} \end{aligned}$$

Taking minimum on the LHS over j gives

$$\tilde{J}_k(x_k) \geq \min_{u_k \in \bar{U}_k(x_k)} \mathbb{E}\left\{g_k(x_k, \mu_{j,k}(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \mu_{j,k}(x_k), \omega_k))\right\}$$

In general, the assumption (5.9a) is not satisfied, but we also obtain an upper bound:

Proposition 5.3. Define $\tilde{J}_k, k = 0, \dots, N$ with $\tilde{J}_N(x_N) = g_N(x_N)$ for all x_N, and let $\pi = \{\bar{\mu}_0, \dots, \bar{\mu}_{N-1}\}$ be policies such that

$$\mathbb{E}\{g_k(x_k, \bar{\mu}_k(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k))\} \leq \tilde{J}_k(x_k) + \delta_k, \ \forall x_k, k \tag{5.12}$$

Then for any x_k and k we have

$$J_{\pi,k}(x_k) \leq \tilde{J}_k(x_k) + \sum_{i=k}^{N-1} \delta_i$$

Proof. We apply the backwards induction on k. In particular, we have $\tilde{J}_N(x_N) = g_N(x_N) = J_{\pi,N}(x_N)$ for any x_N. Assume that

$$J_{\pi,k+1}(x_{k+1}) \leq \tilde{J}_{k+1}(x_{k+1}) + \sum_{i=k+1}^{N-1} \delta_i,$$

then we imply

$$\begin{aligned}
J_{\pi,k} &= \mathbb{E}\left\{ g_k(x_k, \bar{\mu}_k(x_k), \omega_k) + J_{\pi,k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k)) \right\} \\
&\leq \mathbb{E}\left\{ g_k(x_k, \bar{\mu}_k(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k)) \right\} + \sum_{i=k+1}^{N-1} \delta_i \\
&\leq \tilde{J}_k(x_k) + \delta_k + \sum_{i=k+1}^{N-1} \delta_i
\end{aligned}$$

□

5.4 Computational Issues for limited Lookahead

1. Nonlinear programming can be used to compute the One-step lookahead policy or the multi-step version when $U_k(x_k)$ is not a discrete set.

Example 5.7 (Two-state Stochastic Programming). In the first stage we choose u_0 from $U_0 \in \mathbb{R}^m$ with cost $g_(u_0)$. Then we follow an uncertain event represented by a random variable ω taking possible values $\omega^1, \dots, \omega^r$. Once ω^j occurs, we need to choose u_1^j from the set $U_1(u_0, \omega^j) \subseteq \mathbb{R}^m$ at a cost $g_1(u_1^k, \omega^j)$. Therefore, we need to solve

$$\begin{array}{ll}
\min & g_0(u_0) + \sum_{j=1}^r g_1(u_1^j, \omega^j) \\
\text{with} & u_0 \in U_0,\ u_1^j \in U_1(u_0, \omega^j),\ j = 1, \dots, r
\end{array}$$

It can be viewed as a two-stage perfect state information problem, where $x_1 = \omega_0$ denotes the stae equation, ω_0 can take values $\omega^1, \dots, \omega^r$, the cost for first stage is $g_0(u_0)$, the cost for second stage is $g_1(x_1, u_1)$.

This example can be generalized into basic problems for DP with horizon $N = 2$. Suppose ω_0 and ω_1 are random variables taking values $\omega^1, \dots, \omega^r$. The optimal cost is

$$J_0(x_0) = \min_{u_0 \in U_0(x_0)} \left[\sum_{j=1}^{r} p^j \left\{ g_0(x_0, u_0, \omega^j) \right.\right.$$
$$+ \min_{u_1^j \in U_1(f_0(x_0,u_0,\omega^j))} \left[\sum_{i=1}^{r} p^i \left\{ g_1(f_0(x_0, u_0, \omega^j), u_1^j, \omega^i) \right.\right.$$
$$\left.\left.\left.\left. + g_2(f_1(f_0(x_0, u_0, \omega^j), u_1^j, \omega^i)) \right\} \right] \right\} \right]$$

which is equivalent to solving the nonlinear programming problem

$$\text{minimize} \quad \sum_{j=1}^{r} p^j \left\{ g_0(x_0, u_0, \omega^j) + \sum_{i=1}^{r} \left\{ g_1(f_0(x_0, u_0, \omega^j), u_1^j, \omega^i) \right.\right.$$
$$\left.\left. + g_2(f_1(f_0, u_0, \omega^j), u_1^j, \omega^i) \right\} \right\}$$
$$\text{with} \quad u_0 \in U_0(x_0), \; u_1^j \in U_1(f_0(x_0, u_0, \omega^j))$$

2. The choice of approximation functions $\tilde{J}_k$ can be computed in a variety of ways:

   (a) Problem Approximation: Approximate the optimal cost-to-go with some cost derived from a related but simpler problem

   (b) Parametric Cost-to-Go Approximation: Approximate the optimal cost-to-go with a function of a suitable parametric form, e.g., Neuro-Dynamic Programming

   (c) Rollout Approach: Approximate the optimal cost-to-go with the cost of some suboptimal policy, which is calculated either analytically or by simulation.

5.4.1 Rollout Algorithms

One-step Lookahead policy At each stage k and state x_k, we use the control $\bar{\mu}_k(x_k)$ that attains minimum for the expression

$$\begin{array}{ll} \min_{u_k \in U_k(x_k)} & \mathbb{E}\left\{ g_k(x_k, u_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, \omega_k)) \right\} \\ \text{where} & \tilde{J}_N = g_N \\ & \tilde{J}_{k+1} \text{ is the approximation to the true cost-to-go } J_{k+1} \end{array}$$

In the rollout algorithm, $\tilde{J}_{k+1}$ is the cost-to-go of heuristic policy, called the base policy.

Proposition 5.4. The rollout algorithm achieves no worse cost than the base heuristic starting from the same state

Main Difficulty Calculating $\tilde{J}_k(x_k)$ is computationally intensive if the cost-to-go of the base policy cannot be analytically calculated.

- Involve the monte carlo simulation if the problem is stochastic
- Things improve in the deterministic case.

Example 5.8 (Quiz Problem)**.** A person is given N questions; answering correctly question i has probability p_i, reward v_i. Quiz terminates at the first incorrect answer. The problem is to choose the ordering of questions to maximize the total expected reward.

The optimal policy is to use the index policy: answer questions in decreasing order of $p_i v_i/(1-p_i)$

With minor changes in the problem, the index policy will not be optimal:

- A limit on the maximum number of questions that can be answered
- Time windows, sequence-dependent rewards, precedence constraints

Rollout with the index policy as base policy: convenient since at a given state, the index policy and its expected reward can be easily calculated.

Proof for proposition (5.4). It suffices to check the assumption shown in proposition (5.2):

Since $\tilde{J}_k(x_k)$ is the cost-to-go function based on the policy $\{\mu_k(x_k)\}$, we imply

$$\begin{aligned}\tilde{J}_k(x_k) &= \mathbb{E}\left\{g_k(x_k,\mu_k(x_k),\omega_k)+\tilde{J}_{k+1}(f_k(x_k,\mu_k(x_k),\omega_k))\right\}\\ &\ge \min_{u_k\in U_k(x_k)}\mathbb{E}\left\{g_k(x_k,u_k,\omega_k)+\tilde{J}_{k+1}(f_k(x_k,u_k,\omega_k))\right\}\end{aligned}$$

Therefore, applying proposition (5.2) gives

$$\begin{aligned}\bar{J}_k(x_k) &\leq \min_{u_k \in U_k(x_k)} \mathbb{E}\Bigg\{ g_k(x_k, u_k, \omega_k) + \tilde{J}_{k+1}(f_k(x_k, u_k, \omega_k)) \Bigg\} \\ &\leq \mathbb{E}\Bigg\{ g_k(x_k, \mu_k(x_k), \omega_k) + \tilde{J}_{k+1}(f_k(x_k, \mu_k(x_k), \omega_k)) \Bigg\} = \tilde{J}_k(x_k)\end{aligned}$$

□

Alternative Proof for proposition (5.4). Define

$$\begin{aligned}\bar{J}_k(x_k) &: \text{cost to go for the rollout policy} \\ H_k(x_k) &: \text{cost to go for the base policy}\end{aligned}$$

We claim that $\bar{J}_k(x_k) \leq H_k(x_k)$ for all x_k, k: We show this claim by induction. We have $\bar{J}_N(x_N) = H_N(x_N)$ for all x_N. Assume that

$$\bar{J}_{k+1}(x_{k+1}) \leq H_{k+1}(x_{k+1}), \ \forall x_{k+1}$$

It follows that for all x_k,

$$\begin{aligned}\bar{J}_k(x_k) &= \mathbb{E}\Bigg\{ g_k(x_k, \bar{\mu}_k(x_k), \omega_k) + \bar{J}_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k)) \Bigg\} \\ &\leq \mathbb{E}\Bigg\{ g_k(x_k, \bar{\mu}_k(x_k), \omega_k) + H_{k+1}(f_k(x_k, \bar{\mu}_k(x_k), \omega_k)) \Bigg\} \\ &\leq \mathbb{E}\Bigg\{ g_k(x_k, \mu_k(x_k), \omega_k) + H_{k+1}(f_k(x_k, \mu_k(x_k), \omega_k)) \Bigg\} \\ &= H_k(x_k)\end{aligned}$$

□

6

Infinite Horizon Problem

Problem Setting

1. The number of stages is infinite, but
2. The system is stationary, i.e., the system equation, the cost per stage, and the random disturbance does not change from one stage to next.

The objective is to minimize

$$J_\pi(x_0) = \lim_{N\to\infty} \mathbb{E}_{\omega_k, k\geq 0}\left\{\sum_{k=0}^{N-1} \alpha^k g(x_k, \mu_k(x_k), \omega_k)\right\}$$

1. Stochastic Shortest Path problem:

   $$\alpha = 1,$$

   but it is a finite-state system with a *termination* state

2. Discounted Problem with $\alpha < 1$ and bounded cost per stage:

   $$|g(x,\mu,\omega)| \leq C, \quad \alpha < 1$$

3. Average cost per stage problem:

$$\alpha = 1, \quad J_\pi(x_\infty) = \lim_{N\to\infty} \frac{1}{N}\mathbb{E}_\omega \left\{ \sum_{k=0}^{N} g(x_k, \mu_k(x_k), \omega_k) \right\}$$

The infinite horizon problem is challenging for analysis part, and elegant in solution solving and algorithm design.

Preview The key issue is in the relation between the infinite and finite horizon optimal cost-to-go functions. w.l.o.g., let $\alpha = 1$ and let $J_N(x)$ denote the optimal cost for N-stage problem, generated after N DP iterations, starting from $J_0(x) \equiv 0$:

$$J_{k+1}(x) = \min_{u\in U(x)} \mathbb{E}_\omega\{g(x,u,\omega) + J_k(f(x,u,\omega))\}, \quad \forall x$$

There are typicl results for total cost problems:

- Convergence of DP algorithm (value iteration):

$$J^*(x) = \lim_{N\to\infty} J_N(x), \quad \forall x$$

- Bellman's equation holds for all x:

$$J^*(x) = \min_{u\in U(x)} \mathbb{E}_\omega\{g(x,u,\omega) + J^*(f(x,u,\omega))\}$$

- Optimality condition: if $\mu(x)$ minimizes in the RHS of Bellman equation for each x, then the policy $\{\mu, \mu, \dots\}$ is optimal

Problem Formulation

1. The underlying system is

$$x_{k+1} = \omega_k$$

where ω_k denotes the disturbance

2. Denote $\tilde{g}(i,u,j)$ as the cost for using u at state i and moving to state j, and the stage cost $g(x_k,u_k)$ as

$$g(i,u) = \sum_j p_{ij}(u)\tilde{g}(i,u,j)$$

3. The total cost associated with initial state i and policy $\pi = \{\mu_0, \mu_1, \dots\}$ is given by:

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

   where $\alpha \in (0, 1]$. We will pose some assumptions to guarantee the existence of above limit.

4. For special policies $\pi = \{\mu, \mu, \dots\}$, we use $J_\mu(i)$ in place of $J_\pi(i)$.

   We say that μ is optimal if

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

6.0.1 Stochastic Shortest Path Problems

Assumptions

- Finite-state system: states $1, \dots, n$ and a special cost-free termination state t
- There is no discount, i.e., $\alpha = 1$
- Termination inevitable assumption: There exists integer m such that for every policy and initial state, there is positive probability that the termination state will be reached after no more than m staages, i.e., for all π, we have

$$\rho_\pi = \max_{i=1,\dots,n} P\{x_m \neq t \mid x_0 = i, \pi\} < 1$$

Lemma 6.1. The cost-to-go function $J_\pi(i)$ is bounded by

$$|J_\pi(i)| \leq \frac{m}{1-\rho} \max_{i=1,\dots,n,u\in U(i)} |g(i,u)|$$

where $\rho := \max_\pi \rho_\pi < 1$.

Proof. Note that $\rho_\pi < 1$, since it depends only on the first m components of π, and the number of m-stage policies is finite. We may regard ρ_π as an upper bound on the termination probability over m steps.

Therefore, for any π and any initial stae i,

$$P\{x_{2m} \neq t \mid x_0 = i, \pi\} = P\{x_{2m} \neq t \mid x_m \neq t, x_0 = i, \pi\} \\ \times P\{x_m \neq t \mid x_0 = i, \pi\} \leq \rho^2$$

and similarly,

$$P\{x_{km} \neq t \mid x_0 = i, \pi\} \leq \rho^k, \quad i = 1, \ldots, n$$

Therefore, the expected cost (why expected?) between stage km and stage $(k+1)m - 1$ is bounded by

$$m\rho^k \max_{i=1,\ldots,n, u\in U(i)} |g(i,u)|$$

and further

$$|J_\pi(i)| \leq \sum_{k=0}^{\infty} m\rho^k \max_{i=1,\ldots,n, u\in U(i)} |g(i,u)| = \frac{m}{1-\rho} \max_{i=1,\ldots,n, u\in U(i)} |g(i,u)|$$

□

Proposition 6.1. The following results hold for the stochastic shortest path problems:

1. Given the initial condition $J_0(1), \ldots, J_0(n)$, the sequence $J_k(i)$ generated by the *value iteration*

$$J_{k+1}(i) = \min_{u\in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right] \tag{6.1}$$

   converges to the optimal cost $J^*(i)$ for each i. Here $J_k(i)$ denotes the optimal cost starting from state i of a k-stage problem, with cost per stage given by g, and terminal cost at the end of k stages, given by J_0

2. Bellman equation admits the unique solution, which is the optimal cost-to-go function:

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

3. Policy iteration: Fix any stationary policy μ, the costs $J_\mu(1), \dots, J_\mu(n)$ are the unique solution of the equation

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

Furthermore, given any initial conditions $J_0(1), \dots, J_0(n)$, the sequence $J_k(i)$ generated by the DP iteration

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

converges to the cost $J_\mu(i)$ for each i

4. A stationary policy μ is optimal iff for each state i, $\mu(i)$ attains the minimum in the Bellman's equation.

Proof. The idea is to bound the tail of cost-to-go function

$$\sum_{k=mK}^{\infty} \mathbb{E} g(x_k, \mu_k(x_k))$$

1. For each positive integer K, initial state x_0 and policy $\{\mu_0, \mu_1, \dots\}$, we write

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

The expected cost during the Kth m-stage cycle is bounded by $M\rho^k$, where $M = m \max_{i=1,\dots,n, u\in U(i)} |g(i,u)|$, and therefore

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

Denote J_0 as the terminal cost function and $J_0(t) = 0$, and we imply

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

since the probability that $x_{mK} \neq t$ is bounded by ρ^K for any policy. Then we bound

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

The inequality above holds for any policy μ. The minimization over μ gives

$$\begin{aligned}&- \rho^K \max_{i=1:n} |J_0(i)| + J^*(x_0) - \frac{\rho^K M}{1-\rho} \\ &\leq J_{mK}(x_0) \\ &\leq \rho^K \max_{i=1:n} |J_0(i)| + J^*(x_0) + \frac{\rho^K M}{1-\rho}\end{aligned}$$

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

2. It suffices to show the uniqueness. Suppose that $J(1), \ldots, J(n)$ satisfies the Bellman equation, then the value iteration (6.1) starting from the $J(1), \ldots, J(n)$ gives convergence of $J(1), \ldots, J(n)$, i.e., optimal cost-to-go.

3. Given stationary policy μ, consider the modified SSP, i.e., the control constraint is modified as $\tilde{U}(i) = \{\mu(i)\}$, then $J_\mu(1), \ldots, J_\mu(n)$ uniquely solves the Bellman's equation for this modified problem,

and further more from (a) the global convergence result is also shown.

4. The policy $\mu(i)$ is optimal iff

$$
\begin{aligned}
J^*(i) &= \min_{u\in U(i)} \left[g(i,u) + \sum_{j=1}^n p_{ij}(u) J^*(j) \right] \\
&= g(i,\mu(i)) + \sum_{j=1}^n p_{ij}(\mu(i)) J^*(j), \quad i = 1,\dots,n
\end{aligned}
$$

By part (3), $J_\mu(i) = J^*(i)$ for any i. Conversely, if $J_\mu(i) = J^*(i)$ for any i, then (2) and (3) imply the equation above.

□

Example 6.1. Consider the case where $g(i,u) = 1, i = 1,\dots,n, u \in U(i)$. It corresponds to a problem where the objective is to terminate as fast as possible on average, where the optimal cost-to-go function $J^*(i)$ is the minimum expected time to terminate starting from i. The $J^*(i)$ solves the Bellman equation

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

If there is only one control at each state, $J^*(i)$ represents the mean first passage time from i to t, and these times, denoted as m_i, satisfies

$$
m_i = 1 + \sum_{j=1}^n p_{ij} m_j, \ i = 1,\dots,n
$$

Example 6.2 (Spider and Fly). Let the state denotes the distance between spider and fly. For $i \geq 2$, we have

$$
p_{ii} = p, \quad p_{i,(i-1)} = 1-2p, \quad p_{i,(i-2)} = p, \ i \geq 2,
$$

and

$$
\begin{aligned}
&p_{1,1}(M) = 2p, \quad p_{1,0}(M) = 1-2p \\
&p_{1,2}(\bar{M}) = p, \quad p_{1,1}(\bar{M}) = 1-2p, \quad p_{1,0}(\bar{M}) = p.
\end{aligned}
$$

For state $i \geq 2$, the Bellman equation is

$$J^*(i) = 1 + pJ^*(i) + (1-2p)J^*(i-1) + pJ^*(i-2), \quad i \geq 2 \qquad (6.2a)$$

with $J^*(0) = 0$. For the $i = 1$, the Bellman equation is

$$J^*(1) = 1 + \min\{2pJ^*(1), pJ^*(2) + (1-2p)J^*(1)\} \qquad (6.2b)$$

Firstly we solve $J^*(2)$ in terms of $J^*(1)$:

$$J^*(2) = \frac{1}{1-p} + \frac{(1-2p)J^*(1)}{1-p}$$

and substituting $J^*(2)$ into $J^*(1)$ gives

$$J^*(1) = 1 + \min\left[2pJ^*(1), \frac{p}{1-p} + \frac{(1-2p)J^*(1)}{1-p}\right]$$

Solving this equation suffices to solve cases

$$\begin{cases} J^*(1) = 1 + 2pJ^*(1) \\ 2pJ^*(1) \leq \dfrac{p}{1-p} + \dfrac{(1-2p)J^*(1)}{1-p} \end{cases}$$

and

$$\begin{cases} J^*(1) = 1 + \dfrac{p}{1-p} + \dfrac{(1-2p)J^*(1)}{1-p} \\ 2pJ^*(1) \geq \dfrac{p}{1-p} + \dfrac{(1-2p)J^*(1)}{1-p} \end{cases}$$

and therefore

$$J^*(1) = \begin{cases} 1/(1-2p) & \text{if } p \leq 1/3 \\ 1/p & \text{if } p \geq 1/3 \end{cases}$$

Error Bound for Value Iteration From previous proof we can see that $|J_{mK}(i) - J^*(i)|$ is bounded by a constant multiple of ρ^k. We can strengthen this bound, i.e., for all k and j,

$$J_{k+1}(j) + (N^*(j)-1)c_{k,l} \leq J^*(j) \leq J_{\mu^K}(j) \leq J_{k+1}(k) + (N^*(j)-1)c_{k,u}$$

where μ^k is the policy derived in k-th iteration. Moreover,

- $N^*(j)$ denotes the average number of stages to reach t, starting from j and using some optimal stationary policy

- $N^k(j)$ denotes the average number of stages to reach t, starting from j and using the stationary policy μ^k
- $c_{k,l} = \min_{i=1:n} J_{k+1}(i) - J_k(i)$ and $c_{k,u} = \max_{i=1:n} J_{k+1}(i) - J_k(i)$

=

Policy Iteration Start with a stationary policy μ^0, and generate a sequence of new policies $\mu^1, \mu^2, \ldots$

1. Given the policy μ^k, perform a policy evaluation step, which computes $J_{\mu^k}(i)$ for $i = 1 : n$, as the solution to the system
$$J(i) = g(i, \mu^k(i)) + \sum_{j=1}^{n} p_{ij}(\mu^k(i))J(j), \; i = 1, \ldots, n \tag{6.3a}$$
2. Then perform a policy improvement step, which computes the new policy μ^{k+1} as
$$\mu^{k+1}(i) = \arg\min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_{\mu^k}(j) \right] \tag{6.3b}$$
3. This process is repeated until $J_{\mu^{k+1}}(i) = J_{\mu^k}(i)$ for all i, and terminates with the policy μ^k.

Proposition 6.2. The policy iteration algorithm generates an improving sequence of policies, and terminates with an optimal policy.

Proof. For any k, consider the value-iteration-like sequence generated by the recursion
$$J_{N+1}(i) = g(i, \mu^{k+1}(i)) + \sum_{j=1}^{n} p_{ij}(\mu^{k+1}(i)) J_N(j), \; i = 1 : n, \; N \geq 0,$$

and $J_0(i) = J_{\mu^k}(i)$ for $i = 1 : n$. By (6.3a) and (6.3b),

$$\begin{aligned} J_0(i) &= g(i, \mu^k(i)) + \sum_{j=1}^{n} p_{ij}(\mu^k(i)) J_0(j) \\ &\geq g(i, \mu^{k+1}(i)) + \sum_{j=1}^{n} p_{ij}(\mu^{k+1}(i)) J_0(j) \\ &= J_1(i) \\ &\geq g(i, \mu^{k+1}(i)) + \sum_{j=1}^{n} p_{ij}(\mu^{k+1}(i)) J_1(j) = J_2(i) \\ &\geq J_3(i) \geq \cdots \geq \end{aligned}$$

Since $J_N(i) \to J_{\mu^{k+1}}(i)$, we imply $J_0(i) \geq J_{\mu^{k+1}}(i)$. Therefore, the sequence of generated policies is improving, and since the number of stationary policies is finite, we will obtain a finite of iterations to obtain $J_{\mu^k}(i) = J_{\mu^{k+1}}(i)$.

Therefore, after finite iterations we obtain

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

□

Linear Programming Suppose we use value iteration to generate $J_k = (J_k(1), \ldots, J_k(n))$, starting with an initial condition vector $J_0 = (J_0(1), \ldots, J_0(n))$ s.t.

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

Then we will have $J_k(i) \leq J_{k+1}(i)$ for all k and i. By proposition (6.1) we imply $J_0(i) \leq J^*(i)$. Therefore, J^* is the largest J such that

$$J(i) \leq g(i, u) + \sum_{j=1}^{n} p_{ij}(u) J(j), \ i = 1 : n$$

In particular, $J^*(1), \ldots, J^*(n)$ solves the LP of maximzing $\sum_i J(i)$ w.r.t. the constraint above.

6.1 Discounted Problems

Proposition 6.3. The following holds for the discounted problem:

1. The value iteration algorithm

$$J_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], \; i = 1:n \quad (6.4a)$$

   converges to the optimal cost $J^*(i)$ for $i = 1 : n$, starting from any initial conditions $J_0(1), \ldots, J_0(n)$

2. The optimal costs $J^*(1), \ldots, J^*(n)$ of the discounted problem satisfies the Bellman equation

$$J^*(i) = \min_{u \in U(i)} \left[g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) J^*(j) \right] \quad (6.4b)$$

   and in fact they are the unique solution of this equation.

3. For any stationary policy μ, the costs $J_\mu(1:n)$ are the unique solution to the syste,

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

   Furthermore, given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $J_k(i)$ generated by the DP iteration

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

   converges to the cost $J_\mu(i)$ for each i

4. A stationary policy μ is optimal iff for each state i, the $\mu(i)$ attains the minimum in Bellman's equation.

5. The policy iteration algorithm

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

   generates an improving sequence of policies and terminates with an optimal policy

Example 6.3. The optimal value function J^* is the unique solution to the Bellman's equation

$$J^*(x) = \max \left[x, \frac{\mathbb{E}J^*(\omega)}{1+r} \right]$$

The optimal reward is characterized by the benchmark

$$\bar{\alpha} = \frac{\mathbb{E}J^*(\omega)}{1+r}$$

Example 6.4 (Manufacturering). Bellman equation: for $i = 0 : n-1$,

$$J^*(i) = \min[K+\alpha(1-p)J^*(0)+\alpha p J^*(1), ci+\alpha(1-p)J^*(i)+\alpha p J^*(i+1)]$$

and

$$J^*(n) = K + \alpha(1-p)J^*(0) + \alpha p J^*(1)$$

We can show that $J^*(i)$ is monotonicially increasing in i, and we can argue that if processing a batch of m orders is optimal, then processing a batch of $m+1$ orders is also optimal.

6.2 Average cost per stage problems

Consider the optimization with the average cost per stage starting from a state i:

$$J_\pi(i) = \lim_{N\to\infty} \frac{1}{N} \mathbb{E} \left\{ \sum_{k=0}^{N-1} g(x_k, \mu_k(x_k)) \middle| x_0 = i \right\}$$

For most problems of interest the average cost per stage of a policy and the optimal average cost per stage are independent of the initial state.

Assumption one of the states, say n is such that for some integer $m > 0$, and all initial states and all policies, n is visited with positive propability at least once within the first m stages.

We coonect the average cost problem with SSP. Modify the transition probability by introducing one slack node. Fix the expected cost per stage cost incurred at stage i to be

$$g(i,u) - \lambda^*$$

where λ^* is the optimal average cost per stage starting from the special state n

Proposition 6.4. The following hold for the average cost per stage problem

1. The optimal average cost λ^* is the same for all initial states together with some vector $h^* = \{h^*(1), \dots, h^*(n)\}$ satisfying the Bellman's equation

$$\lambda^* + h^*(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h^*(j) \right], \ i = 1:n \quad (6.5)$$

   Furthermore, if $\mu(i)$ attains the minimum in the above equation for all i, the stationary policy μ is optimal. For all vectors h^* satisfying the equation above, there exists a unique vector for which $h^*(n) = 0$.

2. If a scalar λ and a vector $h = \{h(1), \dots, h(n)\}$ satisfy the Bellman equation, then λ is the average optimal cost per stage for each initial state.

3. Given a stationary policy μ with corresponding average cost per stage λ_μ, then tehre is a unique vectors $h_\mu = \{h_\mu(1), \dots, h_\mu(n)\}$ such that $h_\mu(n) = 0$, and

$$\lambda_\mu + h_\mu(i) = g(i, \mu(i)) + \sum_{j=1}^{n} p_{ij}(\mu(i)) h_\mu(j), \ i = 1:n$$

Example 6.5. The Bellman's equation takes the form

$$\lambda^* + h^*(i) = \min[K + (1-p)h^*(0) + ph^*(1), ci + (1-p)h^*(i) + ph^*(i+1)], \ i = 0 : n-1$$

$$\lambda^* + h^*(n) = K + (1-p)h^*(0) + ph^*(1)$$

6.2.1 Value Iteration

$\alpha = 1$, given $J_0(x)$:

$$J_{k+1}(x) = \min_{u \in U} \mathbb{E}_\omega \{ g(x,u,\omega) + J_k(f(x,u,\omega)) \}$$

Reverse the index for infinite horizon case.

6.2.2 Bellman Equation

$$J^*(x) = \min_{u \in U} \mathbb{E}_\omega [g(x,u,\omega) + J^*(x,u,\omega)]$$

wher $x \in \{1, 2, 3 \dots, n\}$. The solution will be

$$J^*(x) = \begin{bmatrix} J^*(1) & \cdots & J^*(n) \end{bmatrix}$$

This will be the *unique* solution to the bellman equation.

6.2.3 Policy Iteration

6.2.4 Stochastic Shortest Path

1. Special Termination State t:

$$P_{t,t}(u) = 1, \ \forall u$$

2. Cost-free termination cost: $g(t,u) = 0, \forall u$

3. Assumption of inevitable termination: We have the "cost-free" termination state, i.e., $g(t,u) = 0, P_{t,t}(u) = 1, \forall u$.

   There exists some integer M from any initial state $x_0 = i$

   For any admissible policy π,

$$\rho_\pi := \max_i P(x_M \neq t \mid x_0 = i, \pi) < 1$$

For ang given $\pi = (\mu_0, \mu_1, \dots)$, study $J_\pi(x_0)$. Bounded or Not?

$$J_\pi(x_0) = \lim_{N \to \infty} \mathbb{E} \left\{ \alpha^k \sum_{k=0}^{N} g(x_k, \mu_k(x_k)) \right\}, \quad \alpha = 1$$

where $g(x_k, \mu_k(x_k)) = \sum_{j=1}^n g(x_k, \mu_k(x_k), j) P_{x_i,j}(\mu_k(x_k))$

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

Start with any initial function $J_0(i), i = 1, \dots, n$, and $J_0(t) = 0$. As $k \to \infty$, $\lim_{k \to \infty} J_k(x_0) = J^*(x_0)$, which is called the vanishing total cost.

Definition 6.1. Let $J^*(x_0)$ be the best cost-to-go function among all the admissiable policy.

$$J^*(x_0) = \min_{\pi} J_\pi(x_0) := \min_{\mu_0,\dots} \mathbb{E}\left(\sum_{\ell=0}^{\infty} g(x_\ell, \mu(x_\ell))\right)$$

$$J_k(x_0) = \min_{\mu_0,\dots,\mu_{k-1}} \mathbb{E}\left\{\sum_{l=0}^{k-1} g(x_k, \mu_k(x_k)) + J_0(x_k)\right\}$$

1. Give $J_0(i)$ for $i = 1, \dots, n$
2. $$J_{k+i}(i) = \min_{u\in U(i)} \left\{g(x,u) + \sum_{j=1}^{n} P_{ij}(u) + J_k(j)\right\}$$
3. As $k \to \infty$, converges to $J^*(i)$

Measure the distance between $J_k(\cdot)$ and $J_{k+1}(\cdot)$:

$$\|J_k - J_{k+1}\|_{1,2,\infty}$$

6.2.5 Bellman Equation

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

As long as $J^*(\cdot)$ satisfies the n equations above, we insist that J^* is really the optimal solution. Moreover, the solution to the equation above is unique.

$$J_\mu(i) = g(i, \mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) J_\mu(j),\ i = 1, \dots, n$$

Aim at the vector $[J_\mu(1), \dots, J_\mu(n)]$

Just keep implement the operator

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

until $J_{k+1}(i)$ converges.

How to compute the stationary policy:

Proposition 6.5 (Necessary and Sufficient Optimality Condition)**.** The policy μ is optimal if and only if μ satisfies the Bellman equation.

6.2.6 Discounted Problems

- State:{1,...,n}

Introduce the artificial t, and introduce the transition from any state to t.

Value Iteration Start with any $J_0(1), \dots, J_0(n)$. Construct

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

and therefore $J^*(i) = \lim_{k \to \infty} J_k(i)$

Bellman equation: find $J^*(1), \dots, J^*(n)$ such that

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

System performance of stationary policy μ Solve

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

Then start with any $J_0(1), \dots, J_0(n)$, we have

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

then $J_\mu(i) = \lim_{k \to \infty} J_k(i)$.

Policy Iteration Once we get $J_{\mu_k}(i)$ for $i = 1, \dots, n$, improve

$$\mu_{k+1}(i) = \arg \min_{u \in U(i)} \left\{ g(i,u) + \alpha \sum_{j=1}^{n} p_{i,j}(u) J_{\mu_k}(i) \right\}$$

Example 6.6 (Asset Selling)**.**

7

The Distribution and Installation

7.1 Pre-requisites

You will need a working LaTeX installation. We recomend using pdflatex to process the files. You will also need biber.exe installed. This is distributed as part of the latest versions of LiveTex and MikTex. If you have problems, please let us know.

7.2 The Distribution

The distribution contains 2 folders: `nowfnt` and `nowfnttexmf`.

7.2.1 Folder `nowfnt`

This folder contains the following files using a flat stucture required to compile a FnT issue:

- essence_logo.eps
- essence_logo.pdf
- now_logo.eps
- now_logo.pdf

- nowfnt.cls
- nowfnt-biblatex.sty
- NOWFnT-data.tex

It also contains the following folders:

journaldata A set of data files containing the journal-specific data for each journal. There are three files per journal:
<jrnlcode>-editorialboard.tex
<jrnlcode>-journaldata.tex
<jrnlcode>-seriespage.tex
<jrnlcode> is the code given in Appendix A. You will need these three files to compile your article.

SampleArticle This folder contains this document as an example of an article typeset in our class file. The document is called `FnT`article.tex. It also contains this PDF file and the .bib file.

7.2.2 Folder `nowfnttexmf`

This folder contains all the files required in a texmf structure for easy installation.

7.3 Installation

If your LaTeX installation uses a localtexmf folder, you can copy the nowtexmf folder to the localtexmf folder and make it known to your TeX installation. You can now proceed to use the class file as normal.

If you prefer to use the flat files, you will need to copy all the required files each time into the folder in which you are compiling the article. Do not forget to copy the three data files for the specific journal from the folder `j`ournaldata.

You may need to configure your TeX editor to be able to run the programs. If you have problems installing these files in your own system, please contact us. We use Computer Modern fonts for some of the journals. You will need to make sure that these fonts are installed. Refer to your system documentation on how to do this.

8

Quick Start

The now-journal class file is designed is such a way that you should be able to use any commands you normally would. However, do **not** modify any class or style files included in our distribution. If you do so, we will reject your files.

The preamble contains a number of commands for use when making the final versions of your manuscript once it has been accepted and you have been instructed by our production team.

8.1 \documentclass

The options to this command enable you to choose the journal for which you producing content and to indicate the use of biber.

```
\documentclass[<jrnlcode>,biber]{nowfnt}.
```

<jrnlcode> is the pre-defined code identifying each journal. See Appendix A for the appropriate <jrnlcode>.

8.2 \issuesetup

These commands are only used in the final published version. Leave these as the default until our production team instructs you to change them.

8.3 \maintitleauthorlist

This is the authors list for the cover page. Use the name, affilliation and email address. Separate each line in the address by \\.

Separate authors by \and. Do not use verbatim or problematic symbols. _ (underscore) in email address should be entered as `\_`. Pay attention to long email addresses.

If your author list is too long to fit on a single page you can use double column. In this case, precede the \maintitleauthorlist command with the following:

```
\booltrue{authortwocolumn}
```

8.4 \author and \affil

These commands are used to typeset the authors and the afilliations on the abstract page of the article and in the bibliographic data.

\author uses an optional number to match the author with the affiliation. The author name is written <surname>, <firstname>.

\affil uses an optional number to match the author with the author name. The content is <affililiation>; <email address>.

8.5 \addbibresource

Use this to identify the name of the bib file to be used.

9

Style Guidelines and LaTeX Conventions

In this section, we outline guidelines for typesetting and using LaTeX that you should follow when preparing your document

9.1 Abstract

Ensure that the abstract is contained within the

```
\begin{abstract}
```

environment.

9.2 Acknowledgements

Ensure that the acknowledgements are contained within the

```
\begin{acknowledgements}
```

environment.

9.3 References

now publishers uses two main reference styles. One is numeric and the other is author/year. The style for this is pre-defined in the LaTeX distribution and must not be altered. The style used for each journal is given in the table in Appendix A. Consult the sample-now.bib file for an example of different reference types.

The References section is generated by placing the following commands at the end of the file.

```
\backmatter
\printbibliography
```

9.4 Citations

Use standard \cite, \citep and \citet commands to generate citations.

Run biber on your file after compiling the article. This will automatically create the correct style and format for the References.

9.4.1 Example citations

This section cites some sample references for your convenience. These are in author/year format and the output is shown in the References at the end of this document.

Example output when using `citet`: **arvolumenumber** is a citation of reference 1 and Bertsekas (1995) is a citation of reference 2.

Example output when using `citep`: (**beditorvolumenumber**) is a citation of reference 3 and (**inproceedings**) is a citation of reference 4.

9.5 Preface and Other Special Chapters

If you want to include a preface, it should be defined as follows:

```
\chapter*{Preface}
\markboth{\sffamily\slshape Preface}
  {\sffamily\slshape Preface}
```

This ensures that the preface appears correctly in the running headings.

You can follow a similar procedure if you want to include additional unnumbered chapters (*e.g.*, a chapter on notation used in the paper), though all such chapters should precede Chapter 1.

Unnumbered chapters should not include numbered sections. If you want to break your preface into sections, use the starred versions of `section`, `subsection`, *etc.*

9.6 Long Chapter and Section Names

If you have a very long chapter or section name, it may not appear nicely in the table of contents, running heading, document body, or some subset of these. It is possible to have different text appear in all three places if needed using the following code:

```
\chapter[Table of Contents Name]{Body Text Name}
\chaptermark{Running Heading Name}
```

Sections can be handled similarly using the `sectionmark` command instead of `chaptermark`.

For example, the full name should always appear in the table of contents, but may need a manual line break to look good. For the running heading, an abbreviated version of the title should be provided. The appearance of the long title in the body may look fine with LaTeX's default line breaking method or may need a manual line break somewhere, possibly in a different place from the contents listing.

Long titles for the article itself should be left as is, with no manual line breaks introduced. The article title is used automatically in a number of different places by the class file and manual line breaks will interfere with the output. If you have questions about how the title appears in the front matter, please contact us.

9.7 Internet Addresses

The class file includes the `url` package, so you should wrap email and web addresses with `\url{}`. This will also make these links clickable in the PDF.

10

Compiling Your FnT Article

During the first run using the class file, a number of new files will be created that are used to create the book and ebook versions during the final production stage. You can ignore these until preparing the final versions as described in Section 10.3. A complete list of the files produced are given in Appendix B.

10.1 Compiling Your Article Prior to Submission

To compile an article prior to submission proceed as follows:

Step 1: Compile the LaTeX file using pdflatex.

Step 2: Run biber on your file.

Step 3: Compile again using pdfLaTeX. Repeat this step.

Step 4: Inspect the PDF for bad typesetting. The output PDF should be similar to FnTarticle.pdf. Work from the first page when making adjustments to resolve bad line breaks and bad page breaks. Rerun pdfLaTeX on the file to check the output after each change.

10.2 Preparing the Final Versions

If you choose the option to compile the final versions of your PDF for publication, you will receive a set of data from our production team upon final acceptance. With the exception of "lastpage", enter the data into the `\issuesetup` command in the preamble.

lastpage= This is the last page number in the sequential numbering of the journal volume. You will need to enter this once you have compiled the article once (see below).

10.3 Compiling The Final Versions

The final versions should be created once you received all the bibliographic data from our Production Team and you've entered it into the preamble. You will be creating a final online journal version pdf; a printed book version pdf; and an ebook version pdf.

Step 1: Compile the LaTeX file using pdfLaTeX.ArAuthor *et al.*, 2014

Step 2: Run biber on your file.

Step 3: Compile again using pdfLaTeX. Repeat this step.

Step 4: Inspect the PDF for bad typesetting. The output PDF should be similar to FnTarticle.pdf. Work from the first page when making adjustments to resolve bad line breaks and bad page breaks. Rerun pdfLaTeX on the file to check the output after each change.

Step 5: When you are happy with the output make a note of the last page number and enter this in `\issuesetup`.

Step 6: Compile the article again.

Step 7: Open the file <YourFilename>-nowbook.tex. This will generate the printed book version pdf.

Step 8: Compile the LaTeX file using pdflatex.

Step 9: Run biber on your file.

Step 10: Compile again using pdfLaTeX. Repeat this step.

Step 11: Repeat steps 7-10 on the file: <YourFilename>-nowebook.tex. This will generate the ebook version pdf.

Step 12: Repeat steps 7-10 on the file: <YourFilename>-nowplain.tex. This will generate a plain version pdf of your article. If you intend to post your article in an online repository, please use this version.

10.4 Wednesday

Proposition 10.1. G is hamiltionian implies that for any nonempty $S \subseteq V$, $G - S$ has at most $|S|$ components.

Theorem 10.1. G is Hamiltionian if for any vetices v, w such that $(v, w) \notin E$,

$$\deg(v) + \deg(w) \geq n$$

Knapsacle Problem

$$\begin{array}{ll} \max & \sum_{i=1}^{n} v_i u_i \\ \text{such that} & \sum_{i=1}^{n} w_i u_i \leq W \\ & u_i \in \{0,1\},\ i = 1, \dots, n \end{array}$$

This is the binary integer programming problem, which is NP-complete. We cannot find exact solution to this problem. There are 2^n possible solutions, which requires exponential time.

We can find some "pseudo"-polynomial algorithm. Assumption:

1. W is an integer.

State: remaining capacity, define as X_k.
Stage: iterm $1, \dots, n$.

$$J_N(x_N) = \begin{cases} v_N & \text{if } N \leq X_N \\ 0, & \text{if } w_N > X_N \end{cases}$$

$$J_N(x_{N-1}) = \begin{cases} 0 + J_N(X_N), & \text{if } w_{N-1} > X_{N-1}, \text{Here } X_N = \\ \max\{v_{N-1} + J_N(x_N), 0 + J_N(x_N)\}, & \text{if } w_{N-1} \leq X_{N-1} \end{cases}$$

Here why the DP has “pesduo”-polynomial time, and the w_i should be integer?

Let’s list states $\{x_1, \dots, x_N\}$. For each stage k, there would be $W+1$.

Each iteration at most 2 computation, and therefore we face $2(W+1) \cdot N$

10.4.1 Label Correcting Methods

Shortest Path. Back up some information.

The label refers to the intermediate information.

A^*-algorithm; Bellman-Ford Algorithm

Benchmark: MILP, CPLEX, CVX.

10.4.2 DP problems with perfect state information

Linear Quadratic System Let $x_k \in \mathbb{R}^1$ be the state; $u_k \in \mathbb{R}^n$ be the control; $\omega_k \in \mathbb{R}^n$ be the disturbance.

System Dynamics.

$$x_{k+1} = A_k x_k + B_k u_k + \omega_k$$

Stage cost:

$$x_k' Q_k x_k + u_k' R_k u_k$$

Here Q_k is required to be positive-semi-definite symmetric matrix; R_k to be positive-definite symmetric matrix.

$$J_{n-1}(x_{n-1}) = \min_{u_{n-1}} \mathbb{E}_\omega \left\{ x_{n-1}' Q_{n-1} x_{n-1} + u_{n-1}' R_{n-1} u_{n-1} + J_n(x_n) \right\}$$

where

$$J_n(x_n) = (A_{n-1}x_{n-1} + Bu_{n-1} + \omega)' Q_n (A_{n-1}x_{n-1} + Bu_{n-1} + \omega)$$

10.4.3 Linear Quadratic

Dynamics:

$$x_{k+1} = A_k x_k + B_k u_k + \omega_k$$

Stage cost:

$$x_k' Q_k x_k + u_k' R_k u_k, \ Q_k \succeq 0, R_k \succ 0$$

Cost to go function:

$$J_k(x_k) = \min_{u_k} \mathbb{E}_{\omega_k}[x_k' Q_k x_k + u_k' R_k u_k + J_{k+1}(x_{k+1})]$$

The final cost is

$$J_N(x_N) = x_N' Q_N x_N$$

At $N-1$ stage

$$\begin{aligned}&x_{N-1}' Q_{N-1} x_N + u_{N-1}' R_{N-1} u_{N-1}\\&+\mathbb{E}[(A_{N-1}x_{N-1} + B_{N-1}u_{N-1} + \omega_{N-1})' Q_N (A_{N-1}x_{N-1} + B_{N-1}u_{N-1} + \omega_{N-1})]\end{aligned}$$

Optimization model:

$$\frac{\partial}{\partial u}(u'Hu + 2r'u + c) = 2Hu + 2r \implies Hu = -r \implies u^* = -H^{-1}r$$

For $N-1$ stage, we have

$$\begin{aligned}H &= R_{N-1} + B_{N-1}' Q_N B_{N-1}\\ r' &= x_{N-1}' A_{N-1}' Q_N B_N\\ c &= x_{N-1}'(Q_{N-1} + A_{N-1}' Q_N A_{N-1})x_{N-1} + \mathbb{E}_\omega(\omega_{N-1}' Q_N \omega_{N-1})\end{aligned}$$

Therefore,

$$u_{N-1}^* = -(R_{N-1} + B_{N-1}' Q_N B_{N-1})^{-1} B_N Q_N A_{N-1} x_{N-1},$$

which is linaer in terms of x_{N-1}, which is so called linear controller.

Therefore,

$$J_{N-1}(x_{N-1}) = x_{N-1}' K_{N-1} x_{N-1} + \mathbb{E}(\omega_{N-1}' Q_N \omega_{N-1})$$

The linear controller will be tested during mid-term or final.

$$J_0(x_0) = x_0' K_0 x_0 + \sum_{k=0}^{N-1} \mathbb{E}_\omega\{\omega_k' K_{k+1} \omega_k\}$$

$$K_{k-1} = f(K_k) \implies K_{k-1} = K_k, \quad \text{for large } k.$$

Riccati Equation for stationary ststem, i.e., $A_k = A, B_k = B, Q_k = Q, R_k = R$.

$$K = A'(K - KB(B'KB + R)^{-1}B'K)A + Q$$

Therefore as time goes long enough, the cost to go function $J_k(x_k)$ will be a constant plus over stages. such a constant K is called the *optimal stationary controller.*

Stability. For $u^* = Lx$, we imply

$$x_{k+1} = Ax_k + Bu_k + \omega_k = (A + BL)x_k + \omega_k$$

Care about $\lim_{k\to\infty}(A + BL)^k = 0$.

Here

$$L = -(B'KB + R)^{-1}B'KA$$

It suffices to solve

$$P_{k+1} = A^2 \left(P_k - \frac{B^2 P_k^2}{B^2 P_k + R} \right) + Q$$

Then consider the case that A_k, B_k are all random matrices, i.e., independent. $Q_k \succeq 0, R_k \succ 0$.

$$L_k = -\left(R_k + \mathbb{E}(B_k' K_{k+1} B_k)\right)^{-1} \mathbb{E}(B_j' K_{k+1} A_k)$$

Note that

$$P_\infty = \frac{\mathbb{E}A^2 RP}{R + EB^2 P} + \frac{\mathbb{E}A^2 \mathbb{E}B^2 - (\mathbb{E}A)^2 (\mathbb{E}B)^2}{R + \mathbb{E}B^2 P}$$

Certainty Equivalence

State: x_k, inventory level

Control: $u_k \geq 0$, number of orders placed

Disturbance: ω_k: damand

Dynamics:

$$x_{k+1} = (x_k + u_k - \omega_k)$$

Stage cost:

- Ordering cost: $c \cdot u_k$

- Maintaining cost: full backlog.$\mathbb{E}_{\omega_k}(r(x_k + u_k - \omega_k))$.

  where $r(z) = hz^+ + pz^-$ is a convex function.

  We imply the maintaining cost is $H(x_k + u_k)$, which is convex.

$$J_k(x_k) = \min_{u_k \geq 0} \{cu_k + H(x_k + u_k) + \mathbb{E}[J_{k+1}(x_k + u_k - \omega_k)]\}$$

Define $Y = x_k + u_k$, and therefore

$$G(Y) = cY + H(Y) + \mathbb{E}[J(Y - \omega_k)]$$

Therefore, we can always set $Y = x_k + u_k = S_k$, where S_k minimizes $G(Y)$.

Acknowledgements

The authors are grateful to Ulrike Fischer, who designed the style files, and Neal Parikh, who laid the groundwork for these style files.

Appendices

A

Journal Codes

The table below shows the journal codes to be used in `\documentclass`. For Example: `\documentclass[ACC,biber]{nowfnt}`

Journal	<jrnlcode>	**R**ef. Style
Annals of Corporate Governance	ACG	Author/Year
Annals of Science and Technology Policy	ASTP	Author/Year
FnT Accounting	ACC	Author/Year
FnT Comm. and Information Theory	CIT	Numeric
FnT Databases	DBS	Author/Year
FnT Econometrics	ECO	Author/Year
FnT Electronic Design Automation	EDA	Author/Year
FnT Electric Energy Systems	EES	Author/Year
FnT Entrepreneurship	ENT	Author/Year
FnT Finance	FIN	Author/Year
FnT Human-Computer Interaction	HCI	Author/Year
FnT Information Retrieval	INR	Author/Year
FnT Information Systems	ISY	Author/Year
FnT Machine Learning	MAL	Author/Year
FnT Management	MGT	Author/Year
FnT Marketing	MKT	Author/Year
FnT Networking	NET	Numeric

Continues

Journal	<jrnlcode>	**R**ef. Style
FnT Optimization	OPT	Numeric
FnT Programming Languages	PGL	Author/Year
FnT Robotics	ROB	Author/Year
FnT Privacy and Security	SEC	Author/Year
FnT Signal Processing	SIG	Numeric
FnT Systems and Control	SYS	Author/Year
FnT Theoretical Computer Science	TCS	Numeric
FnT Technology, Information and OM	TOM	Author/Year
FnT Web Science	WEB	Author/Year

B

Files Produced During Compilation

The files that are created during compilation are listed below. The additional *.tex files are used during the final production process only. See Section 10.3.

```
<YourFilename>-nowbook.tex
<YourFilename>-nowchapter.tex
<YourFilename>-nowebook.tex
<YourFilename>-nowechapter.tex
<YourFilename>-nowsample.tex
<YourFilename>-nowplain.tex
<YourFilename>.aux
<YourFilename>.bbl
<YourFilename>.bcf
<YourFilename>.blg
<YourFilename>.log
<YourFilename>.out
<YourFilename>.pdf
<YourFilename>.run.xml
<YourFilename>.synctex.gz
<YourFilename>.tex
<YourFilename>.toc
```

References

ArAuthor, A., B. Author, and C. Author (2014). “An article on something interesting (volume only)”. *Journal of interesting Things*. 6: 1–122. I have something special to say about this publication. ISSN: 0899-8256. DOI: 10.5161/202.00000013. URL: http://www.nowpublishers.com/ (accessed on 07/10/2014).

Bertsekas, D. (1995). “Dynamic Programming and Optimal Control”. In: vol. 1.

Sutton, R. S. (1988). “Learning to predict by the methods of temporal differences”. *Machine Learning*. 3: 9–44.

Watkins, C. J. C. H. and P. Dayan (1992). “Q-learning”. In: *Machine Learning*. 279–292.