Chapter 1

Dynamic Programming

1.1 The Basic Problem

Dynamics and the notion of state

Optimal control is concerned with optimizing of the behavior of dynamical systems, using information that becomes progressively available as the systems evolve. In this course, we will treat a discrete-time version of the problem. Moreover in this chapter and the first part of the course, we will also assume that the problem terminates at a *specified* finite time, to get what is often called a *finite horizon* optimal control problem. We denote the horizon of the problem by a given integer N. The dynamic system is characterized by its *state* at time $k = 0, 1, \ldots, N$, denoted by x_k [1]. The evolution of the state is subject to control and disturbances, and we will encounter two ways of modeling these dynamics. In the first type of models, the state evolves according to the difference equation

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, \ldots, N-1. \tag{1.1}$$

The state x_k is an element of the set X_k, the *state space* at time k. The *control* u_k is required to belong to the set U_k at time k, and the *process disturbance* w_k (also called plant disturbance, or process noise) to the space W_k. In this course, we will assume a random disturbance model, where w_k is characterized by a probability distribution $P_{w_k}(\cdot|x_k, u_k)$ that may depend on the state x_k and control u_k. However, conditioned on x_k, u_k, the disturbance w_k is assumed independent of the past $w_{0:k-1}, x_{0:k-1}, u_{0:k-1}$

$$P(W_k|X_{0:k}, U_{0:k}, W_{0:k-1}) = P(W_k|X_k, U_k), \forall k.$$

There are situations where a worst-case or adversarial model of the disturbances is more adapted (which leads to minimax control and game theory), and we

[1] In this course we use exclusively state-space models. For input-output models however, see chapter **??**[not written].

might briefly touch upon those. Finally, the initial state x_0 can also be random and characterized by a probability distribution ν, i.e., $P(x_0 \in A) = \nu(A)$ for all (Borel measurable) subset of X_0.

Notation. We use the notation $v_{i:j}$ to denote the sequence $v_i, v_{i+1}, \ldots, v_j$.

The notion of state is of *fundamental importance.* By *definition*, given the state x_k at time k and the sequence of controls $u_k, \ldots, u_{N-1}$, the future trajectory of the system $x_{k+1}, \ldots, x_N$ should be completely characterized. In the case of random disturbances w_k, this means that we can compute the probability that any of these trajectories occur. If we know x_k, we do not need to remember the past state, control or disturbance values prior to time k to compute this probability distribution. In particular, given the state x_k and the control u_k, the disturbance w_k must be independent of $w_{0:k-1} = w_0, \ldots, w_{k-1}$. If not, then x_k is not a proper state vector and its size must be increased to add the necessary memory of the past.

Example 1.1.1 (on the proper choice of a state vector). In [Ber07, section 1.4], Bertsekas discusses a notion of "enlarged state" or "augmented state", in particular for problems with time lags and correlated disturbance noise. This notion is confusing and the confusion only comes from the fact that the notion of state is never defined. Here is an example from [Ber07, section 1.4] which highlights the importance of not being fooled by the notation. Suppose we consider a system which evolves according to the recursion

$$x_0 \text{ given};\; x_1 = f_0(x_0, u_0, w_0);\; x_{k+1} = f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), k \geq 2,$$

with the same assumption that w_k is independent of $w_{0:k-1}$ given x_k and u_k. We see that the recursion is not quite of the form (1.1) because of the presence of an additional time delay. Now suppose we know x_k for some $k \geq 2$ say. Is the (probabilistic) future evolution of the system characterized once the sequence of inputs $u_k, u_{k+1}, \ldots$ is given? In order to compute the probability distribution of x_{k+1}, we need to know x_{k-1} and u_{k-1} in addition to x_k and u_k. So the conclusion is that x_k is not a proper state for the system. What would constitute a state then? Following the idea of our previous counterexample, it is reasonable to think that $\tilde{x}_k = [x_k, x_{k-1}, u_{k-1}]^T$ would constitute a state. This is easy to see if we write a recursion of the form (1.1) for $\tilde{x}_k$

$$\tilde{x}_{k+1} = \begin{bmatrix} x^1_{k+1} \\ x^2_{k+1} \\ x^3_{k+1} \end{bmatrix} = \begin{bmatrix} f_k(x^1_k, x^2_k, u_k, x^3_k, w_k) \\ x^1_k \\ u_k \end{bmatrix} = \tilde{f}_k(\tilde{x}_k, u_k, w_k).$$

Clearly, w_k is also conditionally independent of $w_{0:k-1}$ given $\tilde{x}_k, u_k$, since the later vectors contain more information that x_k, u_k. We conclude that $\tilde{x}_k$ constitutes a proper state for the system.

Remark. The choice of a good state representation can play a crucial role in the ability of solving a problem. It is important to spend time thinking about this in the earlier part of the course and in the exercises.

The second way of modeling the system of interest consists in representing it directly as a *controlled Markov chain*. Namely, we specify directly for each time k and each value of the control $u \in \mathsf{U}_k$ at time k a *transition kernel* $P_k^u(\cdot,\cdot) : (\mathsf{X}_k, \mathcal{X}_{k+1}) \to [0,1]$, where $\mathcal{X}_{k+1}$ is the Borel σ-algebra of X_{k+1}. This means that if $X_k = x$ and $U_k = u$ at time k, the probability that $X_{k+1} \in A$ at time $k+1$, with A a (Borel measurable) subset of X_{k+1}, is given by $P_k^u(x, A)$. The fact that we have a controlled Markov chain is then captured by the property

$$P(X_{k+1} \in A | X_k, X_{k-1}, \dots X_0, U_k, \dots, U_0) = P_k^{U_k}(X_k, A).$$

We also specify a probability distribution ν for the initial state, i.e.

$$\nu(A) = P(X_0 \in A), \; \forall A \in \mathcal{X}_0.$$

What you remember from this is that the sequence of states $X_0, X_1, \dots$ evolves as an (inhomogeneous) Markov chain once the control values $u_0, u_1, \dots$ have been fixed.

Remark. Often we will work with models where the state spaces X_k are countable (finite of infinite), for all k. In this case, the transition kernels are specified by the data

$$P_k^u(x, x') = P(X_{k+1} = x' | X_k = x, u_k = u).$$

For each k and u, we can then think of P_k^u as a matrix, although infinite if the state spaces are. In particular when the state spaces are finite, then for a given k and u, we can think of P_k^u as a matrix. Later on, we will consider homogeneous or stationary models, where the transition kernels do not depend on k. Other notations for $P^u(x, x')$ will then be used occasionally and can be found in the references, e.g. $P_{x,u,x'}, P_u(x, x'), p_{x,x'}(u)$, etc. Markov chains on finite (and sometimes countably infinite) state spaces are often represented as a graph with one node per state and edges labeled by transition probabilities. For controlled Markov chains, this gives different graphs for different choices of controls.

You can probably guess that there is a relationship between the the Markov property in the controlled Markov chain model and the definition of state that we considered with the model based on the difference equation (1.1). The main difference consisted in specifying the distribution of the disturbance w_k in the first model or directly the transition kernel of the state x_k in the second model. Now suppose that we start with a state space model of the form (1.1), and we wish to transform it into a controlled Markov chain model. We simply use in the third equality below the conditional independence of w_k and the past given

x_k, u_k

$$\begin{aligned} P(X_{k+1} \in A | X_{0:k}, U_{0:k}) &= E[1\{X_{k+1} \in A\} | X_{0:k}, U_{0:k}] \\ &= E[1\{f_k(X_k, U_k, W_k) \in A\} | X_{0:k}, U_{0:k}] \\ &= E[1\{f_k(X_k, U_k, W_k) \in A\} | X_k, U_k] \\ &= \int 1\{f_k(X_k, U_k, W_k) \in A\} \, dP(W_k | X_k, U_k) \\ &= P_k^{U_k}(X_k, A). \end{aligned}$$

Hence the Markov property is verified and the transition kernel can be computed from the knowledge of the conditional distribution of W_k given X_k, U_k. Conversely, we can transform a controlled Markov chain model into a recursive model of the form (1.1). Just define the random disturbance W_k on the space X_{k+1} and set

$$X_{k+1} = W_k$$

with $P(W_k \in A | X_k, U_k) = P_k^{U_k}(X_k, A), \ \forall A \in \mathsf{X}_{k+1}$. Variations of this artificial representations are actually useful for simulation. Suppose we have a homogeneous Markov chain $\{X_n\}_{n \geq 0}$ on $\mathbb{N}$ (let's assume no control for clarity) with transition matrix P. Then starting with X_k, we can generate X_{k+1} by using the recursive formula

$$X_{k+1} = j \text{ if } W_k \in \left[\sum_{i=0}^{j-1} P_{X_k, i}, \ \sum_{i=0}^{j} P_{X_k, i} \right],$$

where $\{W_k\}_{k \geq 0}$ is i.i.d. uniform on $[0, 1]$. In conclusion, the two types of models are equivalent, although for practical reasons one might be preferred to the other in specific situations.

Controls and Measurements

The controls $u_0, \ldots, u_{N-1}$ are parameters available to the decision-maker or controller to influence the behavior of the system. The first order of business is to determine *the information on which the choice of control u_k at time k can be based.* It is often the case that the full state x_k of the system cannot be measured directly. Instead we have measurements $y_0, y_1, \ldots$ are of the form

$$y_0 = g_0(x_0, v_0), \quad y_k = g_k(x_k, u_{k-1}, v_k), \quad k = 1, 2, \ldots N-1, \tag{1.2}$$

where v_k is an *observation noise* and belongs to a given space V_k. In this course, the observation noise is also assumed to be random, with a given probability distribution

$$P_{v_k}(\cdot | x_k, \ldots x_0, u_{k-1}, \ldots, u_0, w_{k-1}, \ldots, w_0, v_{k-1}, \ldots v_0),$$

which could in general depend on the current state and past states, controls, and disturbances. An alternative equivalent model, at least for the case where

v_k is conditionally independent of the past given x_k, u_{k-1} is to specify directly the conditional distribution of the observation Y_{k+1} given X_k and U_{k-1}, i.e., specify the transition kernels $P_k^{U_{k-1}}(\cdot,\cdot) : \mathsf{X}_k \times \mathcal{Y}_k \to [0,1]$. In the case of a countable state space, we specify the matrices with elements $P_k^u(x,y)$, the probability of observaing $y \in \mathsf{Y}_k$ when the state is x and the previous control was u. The discussion is analog to our discussion of the model of the dynamics.

The information available to the controller at time k to compute the control u_k is called the *information vector* or *information state* at time k, denoted $\mathcal{I}_k$. When we consider problems with imperfect state information in chapter **??**, we will limit ourselves to the situation where the control vector u_k at time k is allowed to depend on *all* the available past and present information, i.e., the past and current measurements as well as the past controls:

$$\mathcal{I}_0 = y_0, \quad \mathcal{I}_k = (y_0, \dots, y_k, u_0, \dots, u_{k-1}),\ k = 1, \dots, N-1.$$

Note that we always require the control law to be *causal*, i.e., u_k is not allowed to depend on information available after time k (the policy π is then also called a *nonanticipative policy* in the operations research/queueing theory literature). Hence u_k is a function of I_k, denoted $u_k = \mu_k(\mathcal{I}_k)$. In practice, this might require too much memory, or communication bandwidth in distributed control problems, where this requires the sensors to communicate all the measurements to the controller. But departure from this assumption has been an active research area for at least the last four decades, and quickly leads to very difficult issues. The influence of the choice of information state I_k can be very subtle and the difficulty of the control problem depends crucially on it. We will leave the discussion of the information state at this relatively abstract and perhaps vague level for now, and come back to it at several times during the course, starting with chapter **??**. In the first lectures, we assume that we can measure the state perfectly, i.e., $y_k = x_k$ and so without loss of generality, we can take $\mathcal{I}_k = \{x_k\}$ (why?) and we look for a control law of the form $u_k = \mu_k(x_k)$. We call these problems *control problems with perfect information.*

It is important to note that for each k, we look for a *function* $\mu_k(\cdot)$ which defines a control input u_k for each value of the information vector $\mathcal{I}_k$ (or x_k in the case of perfect information). The *policy* or *control law* $\pi = \{\mu_0, \dots, \mu_{N-1}\}$ is then called a *closed-loop policy*, or a *feedback policy.* This point is essentially the difference between optimal control and optimization. In chapter **??** on deterministic control problems, we encounter the concept of *open-loop policy*, where the controls $u_0, \dots, u_{N-1}$ are chosen by the controller once and for all at time $t = 0$, without taking into account any future measurement. In an ideal world with no uncertainty and perfect models, such policies are equivalent to closed-loop policies and there is a large literature devoted to the design such laws (e.g., most of the motion planning literature in robotics, or even most papers on air traffic control!). The reason is that the design of an open-loop policy is a finite-dimensional optimization problem (more on this in chapter **??**), which is familiar to most engineers, whereas optimizing over closed-loop policies is an infinite-dimensional problem. However, using an open-loop policy for a control problem is like taking your car in the morning and trying to drive to work with your eyes closed, assuming you have perfectly memorized the itinerary. For most purpose, you'll probably be better off designing a reasonable even suboptimal closed-loop policy...

Remark. More generally, the choice of control can be *randomized*, that is, u_k is chosen according to a probability distribution depending on I_k and k, although in this course we will usually not need this extra generality. It becomes necessary to consider such randomized control laws when dealing with constrained Markov decision processes for example [Alt99], or minimax control criteria.

The next example illustrates the ideas seen so far, and their equivalent for continuous-time systems. Do not worry too much about the technical details associated with continuous-time noise models, we won't deal with them in the course. It also shows one way of obtaining a discrete-time linear state space model from a continuous-time one by sampling (the so-called *step-invariant transformation* [CF96]).

Example 1.1.2 (discretization of a continuous-time stochastic linear time-invariant system). A vehicle moves on a one-dimensional line, with its position $\xi \in \mathbb{R}$ evolving in continuous time according the differential equation

$$\ddot{\xi}(t) = u(t) + w(t),$$

where $w(t)$ is a zero mean white Gaussian noise with power spectral density $\hat{w}$. Hence $E[w(t)] = 0$, and the autocorrelation of the process is $E[w(t)w(\tau)] = \hat{w}\,\delta(t-\tau)$, where $\delta(\cdot)$ is the Dirac delta function [2]. The deterministic part of the equation corresponds to Newton's law. That is, there is a force $u(t)$ available for the controller to modify the acceleration of the vehicle. However, the acceleration is also subject to a perturbation $w(t)$ modeled as a random noise. What is the state of this system? Ignoring the stochastic component $w(t)$ at least for the moment, and by analogy with our definition in the discrete-time domain, elementary notions of differential equations tell us that the information necessary at time t to characterize the future evolution of the system consists of both the position and velocity of the vehicle, that is, the state is the two-dimensional vector $x = [\xi, \dot{\xi}]^T$. We can rewrite the dynamics as

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w. \tag{1.3}$$

More generally, the state-space representation of a *continuous-time linear Gaussian time-invariant system* can be written as

$$\dot{x}(t) = Ax(t) + Bu(t) + Qw(t), \tag{1.4}$$

[2]we will not spend time in this course on building rigorous foundations in stochastic calculus, because most of the time we will work directly with a discrete-time model. When needed, we will use continuous-time "white noise" freely, as does most of the engineering literature, even though it is usually more convenient from a mathematical point of view to work with integrals of the noise. For a gentle introduction to some of the mathematics involved in a more rigorous presentation, see for example [Oks07].

where x is the state vector of dimension n_x, u is the input vector of dimension n_u, w is a zero mean white Gaussian *process noise* vector of dimension n_w and power spectral density matrix $E[w(t)w(\tau)^T] = W\,\delta(t-\tau)$, and A, B, Q, W are known matrices of appropriate dimensions. The matrix A is called the system matrix, B is the input gain and Q is the noise gain. The output of the system (i.e., the available measurements) is assumed to be a vector of dimension n_y of the form

$$y(t) = Cx(t) + Du(t) + Rv(t). \tag{1.5}$$

Here v is a zero mean white Gaussian *measurement noise* vector of dimension n_v and power spectral density matrix $E[v(tv(\tau)] = V\,\delta(t-\tau)$, and C is called the measurement matrix. We assume that $RV^{1/2}$ is invertible. The general solution of (1.4) can be written explicitely as

$$x(t) = e^{A(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{A(t-\tau)}[Bu(\tau) + Qw(\tau)]d\tau. \tag{1.6}$$

The fact that $w(t)$ is a white noise translates into the fact that $x(t)$ satisfies the properties necessary to represent the state of the system. In this probabilistic model, it means that the distribution of $x(t)$ at some time t conditioned on its values up to an earlier time t' depends only on the last value $x(t')$:

$$P(x(t)|x_{[-\infty,t']}, u_{[t',t]}) = P(x(t)|x(t'), u_{[t',t]}), \text{ for } t' < t.$$

In probabilistic terms, $x(t)$ is a *Markov process* when the system is driven by white noise (this would not necessarily be the case if w were not white, because states prior to t' could be used to predict $w_{[t',t]}$ and $x(t)$ in some way).

Now suppose that we sample the system (1.4) periodically with period T. It is assumed that between samples, the input $u(t)$ is kept constant

$$u(t) = u(kT),\ \forall\, kT \le t < (k+1)T.$$

Let us write $x(k) := x(kT)$, and similarly for the other signals. Then from (1.6) we deduce the following linear time invariant difference equation[3] for $x(k)$

$$x(k+1) = A_d\, x(k) + B_d\, u(k) + \tilde{w}(k), \tag{1.7}$$

where

$$A_d := e^{AT},\ B_d := e^{AT}\left(\int_0^T e^{-A\,s}ds\right)B,$$
$$\tilde{w}(k) = \int_{kT}^{(k+1)T} e^{A((k+1)T-\tau)}Qw(\tau)d\tau.$$

[3]this discretization step is exact for linear systems with such piecewise constant inputs, that is, $x(k)$ represents exactly the value of the signal $x(t)$ at the sampling times, with no approximation involved so far. It is one of the basic techniques in digital control system design, worth remembering.

From the assumption that $w(t)$ is Gaussian, zero mean white and stationary (i.e., W independent of t), it follows that $\{\tilde{w}(k)\}_{k\geq 0}$ is i.i.d. Gaussian with mean $E[\tilde{w}(k)] = 0$ and covariance matrix $E[\tilde{w}(j)\tilde{w}(k)^T] = W_d\, \delta_{jk}$ where

$$\begin{aligned} W_d &= \int_{kT}^{(k+1)T} e^{A((k+1)T-\tau)} QWQ^T e^{A^T((k+1)T-\tau)} d\tau \\ &= \int_0^T e^{A(T-u)} QWQ^T e^{A^T(T-u)} du. \end{aligned}$$

Hence after discretization the system at the sampling times is described by (1.7), which is a discrete time controlled random walk (CRW) of the form (1.1). The process $x(k)$ is a Markov chain, evolving over the continuous state space $\mathbb{R}^{n_x}$. The CRW is driven by a discrete time zero mean Gaussian white noise $\tilde{w}$ with covariance matrix Q_d. Note that a first order approximation (as $T \to 0$) of W_d is

$$W_d \approx (QWQ^T)\, T$$

Exercise 1. Rederive the expression of W_d in more details, starting from the definition of the covariance $E[\tilde{w}(j)\tilde{w}(k)^T]$. You can proceed formally, using properties of the Dirac delta, and not worry about the theory.

We now proceed to discretize the output of the sensor, i.e., the measurement signal $y(t)$ which can be used to design the control $u(k)$. Here the mathematical issues involved in dealing with continuous-time white observation noise appear more clearly. We cannot usually work directly with the samples $v(kT)$ of the observation noise, notably in the very important case of white Gaussian noise, because this would not be well defined mathematically. Instead, we *define* the discrete-time measurement equation to be

$$\tilde{y}(k) = Cx(k) + Du(k) + \tilde{v}(k), \tag{1.8}$$

where $\tilde{v}$ is a discrete-time zero-mean white noise with covariance matrix

$$E[\tilde{v}(j)\tilde{v}(k)^T] =: V_d \delta_{jk}.$$

This measurement equation is of the form (1.2) introduced earlier, except for a time indexing convention for u_k which is not important for the development of the theory. It is possible to establish more formal relationships between the continuous-time and discrete-time measurement equations (1.5) and (1.8), however. Typically, we need to introduce an integration step, which we did not have for the process noise because the system dynamics (1.4) already involve a differential equation with w on the right hand side. It turns out that the covariance of the corresponding discrete-time noise in (1.8) should be chosen as

$$V_d := \frac{RVR^T}{T}. \tag{1.9}$$

In (1.8), we used the notation $\tilde{y}(k)$ instead of $y(k)$ because $\tilde{y}(k)$ is not equal to $y(kT)$. Instead, as in the derivation presented below, $\tilde{y}(k)$ can be thought of as the time average of the signal $y(t)$ over the period T

$$\tilde{y}(k) = \frac{1}{T} \int_{kT}^{(k+1)T} y(\tau) d\tau.$$

This is compatible with real-world measurement devices, which cannot sample continuous-time values perfectly but perform a "short-term" integration. Moreover, in contrast to the discretization of the system dynamics, (1.8) *involves an approximation*, namely, it assumes that the state $x(t)$ is constant over the interval $kT \leq t < (k+1)T$. The derivation of (1.8) below is only given for completeness and can be skipped. It serves to justify the choice of the discrete-time covariance matrix (1.9), which is useful to pass from continuous-time to discrete-time filters for example. But for all practical purposes, in this course equation (1.8) can simply be taken as the definition of the discrete-time values returned by a digital sensor.

Derivation of the measurement equation (1.8): with the definition of $\tilde{y}(k)$ above, and approximating $x(t)$ by the constant $x(k)$ over the interval $kT \leq t < (k+1)T$, we see that

$$\tilde{v}(k) = \frac{1}{T} \int_{kT}^{(k+1)T} R\, v(\tau) d\tau. \tag{1.10}$$

Now by definition the Gaussian white noise $v(t)$ is the formal derivative of a Brownian motion B_t, with mean zero and covariance V. In stochastic calculus, equation (1.10) would be written

$$\tilde{v}(k) = \frac{1}{T} \int_{kT}^{(k+1)T} R\, dB_\tau.$$

The property $E[\tilde{v}(k))] = 0$ is then a standard property of the stochastic integral, and the discrete-time covariance matrix is

$$\begin{aligned} E[\tilde{v}(j)\tilde{v}(k)^T] &= \delta_{jk} \frac{1}{T^2} R\, E\left[\left(\int_{kT}^{(k+1)T} dB_\tau\right)\left(\int_{kT}^{(k+1)T} dB_\tau\right)^T\right] R^T \\ &= \delta_{jk} \frac{R}{T^2} \left(\int_{kT}^{(k+1)T} V d\tau\right) R^T \\ &= \delta_{jk} \frac{RVR^T}{T}. \end{aligned}$$

Here the first line (introduction of δ_{jk}) follows from basic properties of the Brownian motion (the independence property of the increments and their zero-mean distribution), and the second equality is called the Itô isometry, see e.g. [Oks07].

Objective to Optimize

So far we have discussed the dynamical systems of interest in this course, as described in discrete-time by the state space equations (1.1) and (1.2). The specification of a problem involves choosing an appropriate state, describing the available measurements, the process and measurement disturbances, and determining what the available controls or decisions are and how they influence the dynamics of the system and the measurements. Control theory studies methods for choosing the control policy π in order for the system to achieve a certain behavior. In optimal control we specify this behavior by using an objective function, which depends on the trajectory followed by the system, and which the controller attempts to optimize.

We consider cost functions that are *additive over time.* For a problem with finite horizon N and perfect information, this means that we can write the cost function as

$$J_0(x_0, \pi) = E\left[c_N(x_N) + \sum_{k=0}^{N-1} c_k(x_k, \mu_k(x_k), w_k)\right],$$

where the expectation is with respect to the joint distribution of the random variables involved and $\pi = \{\mu_1, \ldots, \mu_{N-1}\}$ is a control policy. Here the cost incurred at period $0 \leq k \leq N-1$ is written $c_k(x_k, u_k, w_k)$ and is often called the *immediate cost* or the *stage cost* (at stage k). At the last period, we incur a cost $c_N(x_N)$ which is called the terminal cost. The quantity $J_N(x_0, \pi)$ is the expected total cost for the problem when the system starts in state x_0 and the policy π is used. If we had an initial probability distribution ν for x_0, we could consider the average cost $J_0(\pi) = \int J_0(x_0, \pi)\, d\nu(x_0)$. An optimal policy π^* is one that minimizes the total expected cost

$$J_0^*(x_0) := J_0(x_0, \pi^*) = \min_{\pi \in \Pi} J_0(x_0, \pi), \tag{1.11}$$

where Π is the set of admissible policies[4]. We call $J^*(x_0)$ the *optimal cost function* or *optimal value function.* Note that the optimal policy π^* is a priori associated with a fixed initial state x_0. However, by using dynamic programming we will typically be able to find a policy π^* that is simultaneously optimal for all initial states.

Remark. We could have considered maximization instead of minimization in (1.11). Then instead of costs we talk about rewards, and we wish to maximize the total expected reward over the horizon. You can simply change min to max, with trivial changes in the theory.

1.2 The Dynamic Programming Algorithm

The dynamic programming idea is based on a very intuitive idea, the *principle of optimality* (the name was coined by Richard Bellman). This principle simply says the following intuitive fact. Assume that $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ is an optimal policy for the problem (1.11). Assume that using this policy π^*, we reach the state x_i at time i with positive probability. Then it must be that the truncated policy $\{\mu_i^*, \ldots, \mu_{N-1}^*\}$ is optimal for the subproblem

$$\min_{\pi^i = \{\mu_i, \ldots, \mu_{N-1}\}} J_i(x_i, \pi^i) = \min_{\pi^i} E\left\{c_N(x_N) + \sum_{k=i}^{N-1} c_k(x_k, \mu_k(x_k), w_k)\right\},$$

[4]to be more precise, in general the "min" in equation (1.11) should be an "inf" without further hypothesis on the control space. Cases where the minimum is not attained will not arise for most of this course, and we will come back to this point only when necessary.

that is, the optimization problem starting from state x_i at time i and running for the remaining time until period N (this is sometimes called "the tail subproblem", and it has an horizon equal to $N - i$). Otherwise, if there were a stricly better policy $\{\hat{\mu}_i, \dots, \hat{\mu}_{N-1}\}$, then we could do better than π^* by following the policy $\{\mu_0^*, \dots, \mu_{i-1}^*, \hat{\mu}_i, \dots, \hat{\mu}_{N-1}\}$, a contradiction.

The dynamic programming (DP) algorithm is based on this idea. It computes (1.11) by proceedings backwards in time, solving successively all the tail subproblems or increasing horizon length. Namely, the tail subproblem starting at period N and consisting in computing $J_0(x_N)$ is trivial to solve since it does not involve any choice of control:

$$J_N^*(x_N) = c_N(x_N). \tag{1.12}$$

Now suppose that we have solved the tail subproblem which run from period $k+1$ until period N and computed the optimal value function $J_{k+1}^*(x_{k+1})$. We can then compute the optimal function $J_k^*(x_k)$ for the tail subproblem with one more period by solving the one-step optimization problem:

$$J_k^*(x_k) = \min_{u_k \in U_k(x_k)} E_{w_k}\left[g_k(x_k, u_k, w_k) + J_{k+1}^*(f_k(x_k, u_k, w_k)) \middle| x_k, u_k \right], \tag{1.13}$$

where the expectation is taken with respect to the probability distribution of w_k conditioned on the values x_k and u_k. Hence we can compute the optimal value function $J_0^*(x_0)$ for the original problem by proceeding recursively and backwards in time, using the algorithm (1.12), (1.13). Moreover, we obtain the optimal policy as well. In state x_k at time k, by letting $\mu_k^*(x_k) = u_k^*$, where u_k^* minimizes the right-hand side of (1.13), we have that the policy $\pi^* = \{\mu_0^*, \dots, \mu_{N-1}^*\}$ is optimal.

Remark. Note that before proceeding to compute $J_k^*(x_k)$ by (1.13), we need to compute $J_{k+1}^*(x_{k+1})$ for *all possible values of* x_{k+1}. That is, at step k of the algorithm, we need to compute the "cost-to-go" function J_k^* on the whole state-space for the tail subproblem.

(Informal) Proof of the Validity of the DP algorithm

The DP algorithm is based on the intuitive principle of optimality, yet its validity requires a formal proof (actually for deterministic problems, the necessity of a proof is discutable). Such a proof can be made more or less technical depending on the assumptions made on the disturbance and control spaces, in particular the existence of a minimizing control in (1.13), but does not provide much additional insight. We encourage the reader to look at the introductory discussion of these mathematical issues in [Ber07, p.23 and p.42]. There is one slightly difficult point arising for stochastic problems which we emphasize below. Overall, it turns out that the DP algorithm works as expected in any reasonable setting for finite-horizon problems, which we assume for now, see [BS78].

For any policy $\pi = \{\mu_0, \dots, \mu_{N-1}\}$, define the truncated policy $\pi^i = \{\mu_i, \dots, \mu_{N-1}\}$. For $i = 0, 1, \dots, N-1$, let $J_i^*(x_i)$ be the optimal cost-to-go for the $N-i$ stage problem that starts at x_i at time i and ends at time N:

$$J_i^*(x_i) = \min_{\pi^i} E\left[c_N(x_N) + \sum_{k=i}^{N-1} c_k(x_k, \mu_k(x_k), w_k)\right].$$

For $i = N$, define $J_N^*(x_N) = c_N(x_N)$. We want to show by backward induction that the functions J_i^* are equal to the functions J_i generated by the DP algorithm, so that the validity of the algorithm is proved when we take $i = 0$.

Clearly $J_N^* \equiv J_N \equiv c_N$. Now assume that $J_{i+1}^* \equiv J_{i+1}$ for some index $i+1$, we want to show that $J_i^* \equiv J_i$. We have

$$\begin{aligned} J_i^*(x_i) = &\min_{(\mu_i, \pi^{i+1})} E\left[c_N(x_N) + c_i(x_i, \mu_i(x_i), w_i) + \sum_{k=i+1}^{N-1} c_k(x_k, \mu_k(x_k), w_k)\right] \\ &= \min_{\mu_i} E_{w_i}\Bigg[c_i(x_i, \mu_i(x_i), w_i) \\ &\qquad + \min_{\pi^{i+1}} E\left[c_N(x_N) + \sum_{k=i+1}^{N-1} c_k(x_k, \mu_k(x_k), w_k)\right]\Bigg]. \end{aligned}$$

This step of separating the minimization problem is trivial for deterministic problems, but requires justification for stochastic problems, since in general the min and E operations do not commute. This step turns out to be correct, a fact we will admit, and justify intuitively using the principle of optimality: "the tail portion of an optimal policy is optimal for the tail subproblem". See [Ber07, section 1.5].

Exercise 2. Give an example of random varialbe X such that $E[\min(X, 0)] \neq \min(E[X], 0)$. Is there a inequality relationship between these two quantities that is always valid?

After this step, the proof is easy:

$$\begin{aligned} J_i^*(x_i) &= \min_{\mu_i} E\Big[c_i(x_i, \mu_i(x_i), w_i) + J_{i+1}^*(f_i(x_i, \mu_i(x_i), w_i))\Big] \\ &= \min_{\mu_i} E\Big[c_i(x_i, \mu_i(x_i), w_i) + J_{i+1}(f_i(x_i, \mu_i(x_i), w_i))\Big] \\ &= \min_{u_i \in \mathsf{U}_i(x_i)} E\Big[c_i(x_i, u_i, w_i) + J_{i+1}(f_i(x_i, u_i, w_i))\Big] \\ &= J_i(x_i). \end{aligned}$$

The first equality follows by definition, the second is the induction hypothesis, the third is a trivial statement.

1.3 Example: Inventory Control [Ber07, p.28]

To illustrate the application of the DP algorithm, we consider a simple optimal inventory control problem. We will revisit this problem in a more general setting later in the course. A shop has a stock x_k of a certain item available at the beginning of the k^{th} period. It can reorder a quantity u_k (immediately delivered) of stock at the beginning of each period, in order to meet a stochastic demand w_k that is realized during that period, and which follows a given probability distribution. The excess demand is lost. Hence the system dynamics follow the equation

$$x_{k+1} = \max(0, x_k + u_k - w_k).$$

Assume that u_k and w_k are nonnegative integers, and that we have a constraint that at most two units of stock can be stored:

$$x_k + u_k \leq 2, \forall k.$$

At each period, we incur a cost of 1 per unit of stock ordered, and a quadratic cost $(x_k+u_k-w_k)^2$ which represents a shortage or holding cost for $(x_k+u_k-w_k)$ negative and positive respectively. Hence the cost per period is

$$c_k(x_k, u_k, w_k) = u_k + (x_k + u_k - w_k)^2.$$

We assume that the terminal cost $c_N(x_N)$ is zero.

Let us take a planning horizon $N = 3$, and compute the optimal value function $J_3^*(x_0)$. Assume that the demand at each period is i.i.d. with probability distribution

$$p(w_k = 0) = 0.1, \;\; p(w_k = 1) = 0.7, \;\; p(w_k = 2) = 0.2.$$

To start the DP algorithm, we have

$$J_3(x_3) = 0,$$

since the terminal cost is 0. Then for $k = 0, 1, 2$, the algorithm takes the form

$$J_k(x_k) = \min_{\substack{0 \leq u_k \leq x_k - 2 \\ u_k = 0,1,2}} E_{w_k} \left\{ u_k + (x_k + u_u - w_k)^2 + J_{k+1}(x_{k+1}) \right\}.$$

Hence the next step is to compute $J_1(x_2)$ for each possible value of the state x_2. For example

$$\begin{aligned} J_2^*(x_2 = 0) &= \min_{u2=0,1,2} E_{w_2} \left\{ u_2 + (u_2 - w_2)^2 \right\} \\ &= \min_{u2=0,1,2} [u_2 + 0.1(u_2)^2 + 0.7(u_2 - 1)^2 + 0.2(u_2 - 2)^2]. \end{aligned}$$

The right hand side evaluates to 1.5 for $u_2 = 0$, 1.3 for $u_2 = 1$ and 3.1 for $u_2 = 2$. Hence $J_2^*(0) = 1.3$ and the optimal control at the beginning of period

2 if the stock is $x_2 = 0$ is to order one unit of additional stock: $\mu_2^*(0) = 1$. Similarly we can compute

$$J_2^*(1) = 0.3,\ \mu_2^*(1) = 0,\ \ J_2^*(2) = 1.1,\ \mu_2^*(2) = 0.$$

At this point, the function $x_2 \to J_2^*(x_2)$ is completely known, and we can proceed to compute the function $x_1 \to J_1^*(x_1)$ and the optimal controls $x_1 \to \mu_1^*(x_1)$. We get

$$J_1^*(0) = 2.5,\ \mu_1^*(0) = 1,\ \ J_1^*(1) = 1.5,\ \mu_1^*(1) = 0,\ \ J_1^*(2) = 1.68,\ \mu_1^*(2) = 0.$$

The last step is then to compute $x_0 \to J_3^*(x_0)$ and the controls $x_0 \to \mu_3^*(x_0)$. We get

$$J_0^*(0) = 3.7,\ \mu_0^*(0) = 1,\ \ J_0^*(1) = 2.7,\ \mu_0^*(1) = 0,\ \ J_0^*(2) = 2.818,\ \mu_0^*(2) = 0.$$

Thus if the initial stock if x_0, we see that by following the optimal policy we can achieve an expected cost of 3.7. The optimal policy always orders one unit if the current stock is zero, and nothing otherwise.

Exercise 3. Do the computations of the previous example in details, following the DP algorithm.

1.4 Practice Problems

Problem 1.4.1. Do all the exercises found in the chapter.

Bibliography

[Alt99] E. Altman. *Constrained Markov Decision Processes.* Chapman & Hall/CRC, 1999.

[Ber07] D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 3rd edition, 2007.

[BS78] D.P. Bertsekas and S.E. Shreve. *Stochastic Optimal Control: The Discrete Time Case.* Academic Press, N.Y., 1978. republished by Athena Scientific, 1996.

[CF96] T. Chen and B.A. Francis. *Optimal Sampled-Data Control Systems.* Springer, 1996.

[Oks07] B. Oksendal. *Stochastic Differential Equations: An Introduction with Applications.* Springer, 6th edition, 2007.