Modeling for dynamic programming

Prof. Paolo Brandimarte
Dip. di Scienze Matematiche – Politecnico di Torino
e-mail: `paolo.brandimarte@polito.it`
URL: `https://staff.polito.it/paolo.brandimarte`

This version: May 7, 2025

References

These slides are taken from my book:

P. Brandimarte. *From Shortest Paths to Reinforcement Learning: A MATLAB-Based Introduction to Dynamic Programming*. Springer, 2021.

Other references:

- W.B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (2nd ed.). Wiley, 2011.

- G.J. van Ryzin, K.T. Talluri. *An Introduction to Revenue Management*. In INFORMS TutORials in Operations Research, 2005, `https://pubsonline.informs.org/doi/10.1287/educ.1053.0019`.

The DP principle is a flexible concept for the decomposition of a multistage decision problem into a sequence of single-stage problems, by balancing the immediate contribution and the expected contributions from future decisions.

This hinges on the value function defined by the DP recursion

$$V_t(\mathbf{s}_t) = \underset{\mathbf{x}_t \in \mathcal{X}(\mathbf{s}_t)}{\text{opt}} \left\{ f_t\big(\mathbf{s}_t, \mathbf{x}_t\big) + \gamma \mathsf{E}\big[V_{t+1}\big(\mathbf{s}_{t+1}\big) \mid \mathbf{s}_t, \mathbf{x}_t\big] \right\}. \tag{1}$$

Equation (1) is only one possible form of DP recursion.

We may adopt more specific forms, e.g., when risk factors are discrete random variables and the expectation boils down to a sum.

A more radical rephrasing is sometimes adopted, where we swap expectation and optimization.

This may occur because of the information structure of the problem at hand, or it may be the result of some manipulation aimed at avoiding the solution of a difficult stochastic optimization subproblem.

One well-known case occurs in Q-learning, a form of reinforcement learning where the state value function $V(\mathbf{s})$ is replaced by Q-factors $Q(\mathbf{s}, \mathbf{x})$ representing the value of state–action pairs.

More generally, we will see how the swap of optimization and expectation can be obtained by introducing the concept of post-decision states.

Also finding a suitable description of the system state may require careful modeling. For instance, a redefinition of the state space may be needed to eliminate path-dependencies and to obtain an augmented and equivalent Markovian model, in order to make the problem amenable to a DP approach.

We should also note that, although it is natural to think of states and control decisions as scalars or vectors, they may consist of different objects, like sets.

The bottom line is that modeling skills come hand in hand with algorithmic knowledge, when tackling a decision problem by DP. These skills can only be honed by experiencing with a diverse array of problems.

The term **Markov decision process** (**MDP**) is reserved to problems featuring discrete state and action spaces.

The term **action** is often adopted to refer to control decisions in this context. Since states and actions are discrete, they can be enumerated, i.e., associated with integer numbers.

Here, we only consider *finite* MDPs. Hence, even though states may correspond to vectors of a multidimensional space, we will refer to states by a notation like

$$i, j \in \mathcal{S} = \{1, \ldots, N\},$$

where N is the size of the state space.

By the same token, we may use a notation like a or a_t to refer to actions, as well as $\mathcal{A}_t(i)$ or $\mathcal{A}(i)$ to denote the set of feasible actions at state $i \in \mathcal{S}$ at time t. We will also denote the whole set of possible actions by $\mathcal{A}$.

The underlying system is essentially a discrete-time and finite **Markov chain**, where it is customary to represent dynamics by a matrix collecting transition probabilities between pairs of states.

In MDPs we use transition probabilities too, but, since we deal with a (partially) controlled Markov chain, the transition probabilities depend on the selected action:

$$\pi_{t+1}(i, a, j)$$

is the probability of a transition from state i to state j during time interval $t + 1$, after choosing action a at time t.

Let us consider again the toy stochastic inventory control problem, where demand may assume values within the set $\{0, 1, 2\}$, with probabilities 0.1, 0.7, and 0.2, respectively.

Since there is a constraint on the maximum inventory level, $I_t \leq 2$, the state space is

$$\mathcal{S} = \{0, 1, 2\},$$

since inventory may take values 0, 1, and 2. By the same token, the feasible action sets are

$$\mathcal{A}(0) = \{0, 1, 2\},\ \mathcal{A}(1) = \{0, 1\},\ \mathcal{A}(2) = \{0\}.$$

The transition probability matrices $\mathbf{\Pi}(a)$, where each element $\pi_{ij}(a)$ contains the transition probability $\pi(i, a, j)$, are as follows:

$$\mathbf{\Pi}(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0.9 & 0.1 & 0 \\ 0.2 & 0.7 & 0.1 \end{bmatrix}, \quad \mathbf{\Pi}(1) = \begin{bmatrix} 0.9 & 0.1 & 0 \\ 0.2 & 0.7 & 0.1 \\ \cdot & \cdot & \cdot \end{bmatrix}, \quad \mathbf{\Pi}(2) = \begin{bmatrix} 0.2 & 0.7 & 0.1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}.$$

Here, the dot $\cdot$ is used for rows corresponding to states in which an action is not feasible. For instance, the action $a = 2$ is only feasible at state $i = 0$, because of the constraint on maximum inventory.

In the MDP setting, the value function at any time instant t boils down to a finite-dimensional vector with components $V_t(i)$, and its satisfies

$$V_t(i) = \underset{a \in \mathcal{A}(i)}{\text{opt}} \left\{ f_t(i,a) + \gamma \sum_{j \in \mathcal{S}} \pi_{t+1}(i,a,j) V_{t+1}\big(j\big) \right\}, \qquad i \in \mathcal{S}. \tag{2}$$

In the discounted infinite-horizon case, we have

$$V(i) = \underset{a \in \mathcal{A}(i)}{\text{opt}} \left\{ f(i,a) + \gamma \sum_{j \in \mathcal{S}} \pi(i,a,j) V\big(j\big) \right\}, \qquad i \in \mathcal{S}. \tag{3}$$

If the immediate contribution is stochastic, possibly because it depends on the next state, we may denote it as $h(i,a,j)$ and rewrite Eq. (3) as

$$V(i) = \underset{a \in \mathcal{A}(i)}{\text{opt}} \sum_{j \in \mathcal{S}} \pi(i,a,j) \Big\{ h(i,a,j) + \gamma V\big(j\big) \Big\}, \qquad i \in \mathcal{S}. \tag{4}$$

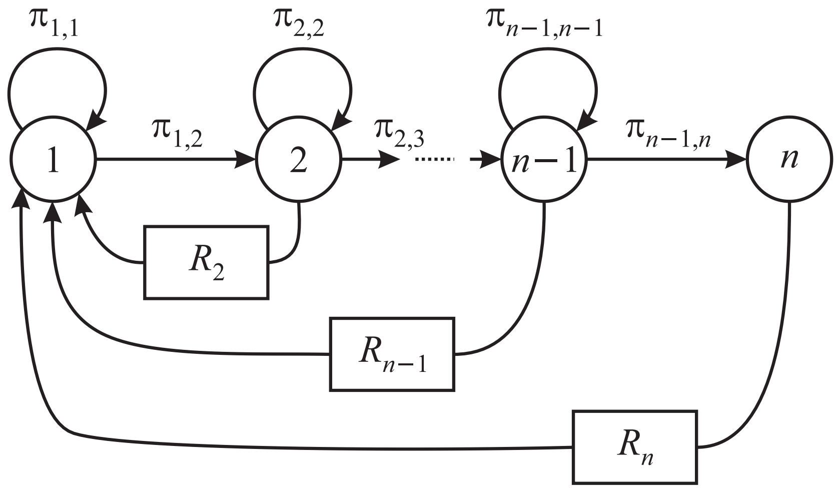
In principle, we may write

$$f(i,a) = \sum_{j \in \mathcal{S}} \pi(i,a,j) h(i,a,j),$$

but this may be not practical, either because N is finite but huge, or because transition probabilities are not known.

To illustrate finite MDPs and controlled Markov chains let us consider a simple case, where states $k = 1, 2, \ldots, n$ are arranged in a sort of linear chain.

The state can only move to the right, but it can be reset, resulting in the collection of a reward and a system restart.



When we are at state 1, we will stay there with probability π_{11}, or we will move to the next state 2 with probability π_{12}. In state 1, there is no decision to make.

In each "interior" state $k = 2, \ldots, n-1$, we have a choice between two actions: wait and reset.

- If we wait and do nothing, there is no reward and we remain in state k with probability π_{kk}, or we move to the next state in the chain with probability $\pi_{k,k+1}$.
- Alternatively, we may exercise an option to collect an immediate reward R_k and reset the state to 1.

Since the transition is deterministic when we choose the reset action, we streamline the notation and use π_{ij} as a shorthand for $\pi(i, \text{wait}, j)$, $i, j \in \mathcal{S}$.

With reference to Eq. (3), the immediate contributions are

$$f(k, \text{wait}) = 0, \quad f(k, \text{reset}) = R_k.$$

We assume that the rewards are increasing along the chain, i.e., $R_k < R_{k+1}$, so there is a tradeoff between collecting the immediate reward and waiting for better opportunities. If we are patient enough and reach the last state n, we may immediately earn the largest reward R_n and move back to state 1.

This problem bears a definite similarity with optimal stopping problems. A possible interpretation is that we own an asset, whose price process is stochastic, and we have to choose when it is optimal to sell the asset. In general, the price need not be monotonically increasing.

The DP equations to find the value $V(i)$ of each state $i \in \mathcal{S}$ are as follows:

$$\begin{aligned}
V(1) &= \gamma\pi_{1,1}V(1) + \gamma\pi_{1,2}V(2) \\
V(k) &= \max\left\{R_k + \gamma V(1),\ \gamma\pi_{k,k}V(k) + \gamma\pi_{k,k+1}V(k+1)\right\}, \qquad k = 2, \ldots, n-1 \\
V(n) &= R_n + \gamma V(1).
\end{aligned}$$

Policy evaluation and Q-factors

The previous example might suggest that finite MDPs are a rather simple business to deal with.

The value function is just a vector and, in the case of a finite horizon, we have an explicit relationship between value functions at different stages. If the set of feasible actions is not too large, the optimization step is trivially solved by enumeration.

In the infinite-horizon case, we have to find state values by solving a system of equations, which are not linear (piecewise linear, in fact). However, they can be solved by rather simple iterative methods.

However, the MDP business may not be so simple, because of the sheer size of the state space. Furthermore, even finding the whole set of transition probabilities $\pi(i, a, j)$ may be quite difficult, if not impossible, when the curse of modeling strikes.

One ingredient to circumvent these difficulties is Monte Carlo sampling. Another ingredient is to rewrite the DP recursion in a different way, based on Q-**factors**. Let us do this for the infinite horizon case.

Informally, a Q-factor $Q(i, a)$ measures the value of taking action a when in state i. More precisely, this should be defined with reference to the policy that will be applied in the next steps.

Let us consider a feasible stationary policy μ, mapping each state i into action $a = \mu(i) \in \mathcal{A}(i)$. The state value function when following policy μ can be found by solving a system of linear equations:

$$V_\mu(i) = f\big(i, \mu(i)\big) + \gamma \sum_{j \in \mathcal{S}} \pi(i, \mu(i), j) \cdot V_\mu\big(j\big), \qquad i \in \mathcal{S}. \tag{5}$$

We will discuss Eq. (5) in more depth later.

For now, it suffices to say that it is fundamental for numerical methods, based on policy iteration, in which we first assess the value of a candidate stationary policy μ, and then we try to improve it.

Also note the difference with respect to Eq. (3), which involves the optimal choice of action a. Here, there is no optimization involved, as the action is prescribed by the selected policy μ.

We define the Q-factor for the stationary policy μ as the following mapping $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$:

$$Q_\mu(i, a) \doteq f(i, a) + \gamma \sum_{j \in \mathcal{S}} \pi(i, a, j) \cdot V_\mu\big(j\big). \tag{6}$$

The idea is to apply action a at the current state i, and then follow policy μ.

We may also define the optimal Q-factors by plugging the optimal value function into Eq. (6):

$$Q(i, a) = f(i, a) + \gamma \sum_{j \in \mathcal{S}} \pi(i, a, j) V\big(j\big), \qquad i \in \mathcal{S}, a \in \mathcal{A}(i). \tag{7}$$

Then,

$$V(j) \equiv \operatorname*{opt}_{a \in \mathcal{A}(j)} Q(j,a), \qquad j \in \mathcal{S}.$$

Hence, we may rewrite the DP recursion in terms of Q-factors:

$$Q(i,a) = f(i,a) + \gamma \sum_{j \in \mathcal{S}} \pi(i,a,j) \left[\operatorname*{opt}_{\tilde{a} \in \mathcal{A}(j)} Q(j,\tilde{a}) \right], \qquad i \in \mathcal{S},\ a \in \mathcal{A}(i). \tag{8}$$

If we compare Eqs. (3) and (8), we may notice that there are both good and bad news:

- The bad news is that, instead of a state value function $V(i)$, now we have state–action value functions $Q(i,a)$. Unless we are dealing with a small sized problem, it seems that we have just made the curse of dimensionality even worse.

- The good news is that now we have swapped expectation and optimization.

We will appreciate the advantage of swapping expectation and optimization later, especially when dealing with continuous problems: it is often easier to solve many simple (deterministic) optimization problems than a single large (stochastic) one.

Furthermore, we may learn the Q-factors by statistical sampling, which is useful in both large-scale problems and problems with an unknown underlying dynamics. This paves the way to model-free DP.

Last, but not least, when dimensionality precludes finding the factors $Q(i,a)$ for all state–action pairs, we may devise a compact representation based on an approximation architecture, which may range from a relatively straightforward linear regression to deep neural networks.

Let us consider once more the basic recursive DP equation, as given in Eq. (1):

$$V_t(\mathbf{s}_t) = \operatorname*{opt}_{\mathbf{x}_t \in \mathcal{X}(\mathbf{s}_t)} \Big\{ f_t(\mathbf{s}_t, \mathbf{x}_t) + \gamma \mathsf{E}\big[V_{t+1}\big(\mathbf{s}_{t+1}\big) \mid \mathbf{s}_t, \mathbf{x}_t\big] \Big\},$$

The underlying assumptions are:

1. We first observe the state $\mathbf{s}_t$ at time instant t.

2. Then, we make a feasible decision $\mathbf{x}_t \in \mathcal{X}(\mathbf{s}_t)$.

3. Then, we observe an immediate contribution $f_t(\mathbf{s}_t, \mathbf{x}_t)$.

4. Finally, we move to a new state $\mathbf{s}_{t+1}$, which depends on realized risk factors $\boldsymbol{\xi}_{t+1}$ during the subsequent time interval, according to a probability distribution that might depend on the current state $\mathbf{s}_t$ and the selected decision $\mathbf{x}_t$.

In order to find the value $V_t(\cdot)$ of each state $\mathbf{s}_t$, based on knowledge of $V_{t+1}(\cdot)$, we should solve a stochastic optimization problem that may involve a quite challenging expectation. However, in the case of an MDP, we have seen that Q-factors allow us to swap optimization and expectation. Actually, we may *have* to swap them in order to reflect the *information structure*, as shown in the following example.

Under demand uncertainty, we should wonder about the precise nature of the demand process $\big(d_t\big)_{\{t \geq 1\}}$.

In a B2C (business-to-consumer) setting, like a retail store, we may not have any advance information about demand in the next time interval. However, in a B2B (business-to-business) setting, we probably receive *formal* orders from customers over time. In the limit, we may have perfect information about demand in the next time interval, even though uncertainty may be relevant for the not-so-distant future.

If, at time instant t, we have perfect knowledge about demand d_{t+1}, the immediate cost contribution is actually deterministic:

$$f_t\big(I_t, x_t, d_{t+1}\big) = hI_t + cx_t + \phi \cdot \delta(x_t) + q \cdot \max\big\{0, d_{t+1} - I_t - x_t\big\},$$

involving an inventory holding charge h, variable and fixed ordering costs c and ϕ, and a lost sales penalty q. Note that if there is a large fixed ordering charge, we might prefer to leave a small amount of demand unsatisfied.

In order to write the DP recursion, we should pay attention to the precise relationships among information flow, decisions, and system dynamics. Given the current inventory state I_t, first we observe demand, *then* we make a decision. Hence, the DP recursion in this case reads:

$$V_t(I_t) = \mathsf{E}_t\left[\min_{x_t \geq 0}\left\{f_t\big(I_t, x_t, d_{t+1}\big) + V_{t+1}(I_{t+1})\right\}\right].$$

The previous example shows a case in which the form of the DP recursion is

$$V_t(\mathbf{s}_t) = \mathrm{E}_t\left[\operatorname*{opt}_{\mathbf{x}_t \in \mathcal{X}(\mathbf{s}_t)} \left\{ f_t(\mathbf{x}_t, \mathbf{s}_t) + \gamma V_{t+1}\big(\mathbf{s}_{t+1}\big) \right\} \right]. \tag{9}$$

The potential advantages of this form are:

- The inside optimization problem is deterministic.
- The outside expectation may be estimated by statistical sampling.

Since these are relevant advantages, we may try to rephrase the standard DP recursion in this form, even though it may not be the natural one.

Sometimes, this can be accomplished by rewriting the system dynamics based on post-decision state variables.

In the familiar view of system dynamics, we consider transitions from state $\mathbf{s}_t$ to state $\mathbf{s}_{t+1}$, depending on the selected decision $\mathbf{x}_t$ and the realized risk factor $\boldsymbol{\xi}_{t+1}$. Sometimes, it is convenient to introduce an intermediate state, observed after the decision $\mathbf{x}_t$ is made, but *before* the risk factor $\boldsymbol{\xi}_{t+1}$ is realized. Such a state is referred to as **post-decision** state, and it may be denoted by $\mathbf{s}_t^x$.

Example. In a lot-sizing problem, we may introduce a post-decision state, the inventory level I_t^x after making the ordering decision at time instant t, but before observing and meeting demand during the time interval $t+1$:

$$I_t^x = I_t + x_t.$$

Hence, we break the transition to the next state into two steps, the second one being

$$I_{t+1} = I_t^x - d_{t+1}.$$

The introduction of post-decision states changes the dynamics from the standard sequence

$$(\mathbf{s}_0, \mathbf{x}_0, \boldsymbol{\xi}_1; \mathbf{s}_1, \mathbf{x}_1, \boldsymbol{\xi}_2; \ldots; \mathbf{s}_{t-1}, \mathbf{x}_{t-1}, \boldsymbol{\xi}_t; \ldots)$$

to

$$\big(\mathbf{s}_0, \mathbf{x}_0, \mathbf{s}_0^x, \boldsymbol{\xi}_1; \mathbf{s}_1, \mathbf{x}_1, \mathbf{s}_1^x, \boldsymbol{\xi}_2; \ldots; \mathbf{s}_{t-1}, \mathbf{x}_{t-1}, \mathbf{s}_{t-1}^x, \boldsymbol{\xi}_t; \ldots\big)$$

In other words, we split the transition equation $\mathbf{s}_{t+1} = g_{t+1}\big(\mathbf{s}_t, \mathbf{x}_t, \boldsymbol{\xi}_{t+1}\big)$ into two steps:

$$\mathbf{s}_t^x = g_t^1\big(\mathbf{s}_t, \mathbf{x}_t\big), \tag{10}$$

$$\mathbf{s}_{t+1} = g_{t+1}^2\big(\mathbf{s}_t^x, \boldsymbol{\xi}_{t+1}\big). \tag{11}$$

This is also reflected in terms of value functions, as we may introduce the **value of the post-decision state** at time t, $V_t^x(\mathbf{s}_t^x)$, where superscripts x point out that these quantities refer to post-decision states.

The relationship between the standard value function $V_t(\mathbf{s}_t)$ and $V_t^x(\mathbf{s}_t^x)$ can be expressed as

$$V_t^x(\mathbf{s}_t^x) = \mathsf{E}\big[V_{t+1}(\mathbf{s}_{t+1}) \mid \mathbf{s}_t^x\big]. \tag{12}$$

Note that, in moving from $\mathbf{s}_t^x$ to $\mathbf{s}_{t+1}$, no decision is involved. Then, using Eq. (12), we may rewrite the standard DP recursion as a deterministic optimization problem,

$$V_t(\mathbf{s}_t) \;=\; \underset{\mathbf{x}_t \in \mathcal{X}(\mathbf{s}_t)}{\mathsf{opt}} \Big\{ f_t(\mathbf{s}_t, \mathbf{x}_t) + \gamma V_t^x(\mathbf{s}_t^x) \Big\}, \tag{13}$$

where the post-decision state $\mathbf{s}_t^x$ is given by the first transition equation (10). If we move one step backward in time to instant $t-1$ and write Eq. (12) for state $\mathbf{s}_{t-1}^x$, we have

$$V_{t-1}^x(\mathbf{s}_{t-1}^x) = \mathsf{E}\big[V_t(\mathbf{s}_t) \mid \mathbf{s}_{t-1}^x\big].$$

Then, by plugging Eq. (13) into this relationship, we obtain

$$\begin{aligned} V_{t-1}^x(\mathbf{s}_{t-1}^x) &= \mathsf{E}\big[V_t(\mathbf{s}_t) \mid \mathbf{s}_{t-1}^x\big] \\ &= \mathsf{E}\left\{ \underset{\mathbf{x}_t \in \mathcal{X}(\mathbf{s}_t)}{\mathsf{opt}} \Big[f_t(\mathbf{s}_t, \mathbf{x}_t) + \gamma V_t^x(\mathbf{s}_t^x) \Big] \;\Big|\; \mathbf{s}_{t-1}^x \right\}. \end{aligned} \tag{14}$$

Leaving the inconsequential backshift in time aside, we find again a DP recursion featuring a swap between expectation and optimization.

The DP recursion in terms of Q-factors, actually, fits naturally into this framework, since the state–action pair (i, a) may be regarded as a post-decision state, before observing the random transition to the new state.

When dealing with inventory management, on-hand inventory looks like the natural state variable and the obvious basis for any ordering decision.

On the contrary, ordering decisions should not be based on on-hand inventory, but on *available* inventory, which accounts for inventory on-order (items ordered from the supplier, but not yet delivered) and backlog.

The standard state equation

$$I_{t+1} = I_t + x_t - d_{t+1}$$

assumes that what is ordered at time instant t is immediately available to satisfy demand during the following time interval $t+1$.

If the delivery lead time is an integer number LT ≥ 1 of time intervals, we need to introduce state variables keeping track of what was ordered in the past and is still in the transportation pipeline.

Let us denote these state variables by $z_{t,\tau}$, the amount that will be delivered τ time intervals after the current time t, where $\tau = 0, 1, 2, \ldots, \mathrm{LT} - 1$. Hence, $z_{t,0}$ represents what is immediately available at the current time instant t, and it is involved in the state transition equation for on-hand inventory:

$$I_{t+1} = I_t + z_{t,0} - d_{t+1},$$

if we disregard demand uncertainty.

What we order at time instant t, represented by decision variable x_t, will be available LT time intervals after t. Hence, at the next time instant $t+1$, the amount x_t will be $\mathsf{LT}-1$ time intervals from delivery. We may therefore relate the decision x_t to the additional state variable corresponding to $\tau = \mathsf{LT}-1$ as follows:

$$z_{t+1,\mathsf{LT}-1} = x_t.$$

The general transition equation for the additional state variables $z_{t,\tau}$, for $\tau < \mathsf{LT}-1$, boils down to a simple time shift:

$$z_{t+1,\tau} = z_{t,\tau+1}, \qquad \tau = 0, 1, \ldots, \mathsf{LT}-2. \tag{15}$$

By a similar token, when dealing with perishable items we need to describe inventory at each age, by introducing an array of state variables $I_{t,\tau}$ that represent the amount of on-hand inventory at time t with an age of τ time periods.

The state transitions are not quite trivial to write down (see the book), and depend on the kind of inventory issuing, FIFO (first-in-first-out) or LIFO (last-in-first-out).

Revenue management consists of a series of models and techniques to maximize the revenue obtained by selling perishable resources. The idea was introduced (under the name of yield management) within the airline industry.

There are two basic approaches to revenue management: quantity-based and price-based. In the first case, resource availability is restricted according to some policy in order to maximize revenue. In the second one, prices are adjusted dynamically, and we talk of dynamic pricing.

We outline simple DP models for quantity-based revenue management and assume that the marginal cost of each unit of resource is zero or negligible, so that profit maximization boils down to revenue maximization.

We consider C units of a single resource (say, seats on an aircraft), which must be allocated to n fare classes, indexed by $j = 1, 2, \ldots, n$. Units in class j are sold at price p_j, where

$$p_1 > p_2 > \cdots > p_n.$$

Thus, class 1 is the first class, from which the highest revenue is earned.

We assume that the seats allocated to different classes are actually identical, but they are bundled with ancillaries (like cancellation rights, weekend restrictions, on-board meals, etc.) to offer different packages.

The price of each class is fixed, but we control the available inventory by restricting the seats available for each class.

Demand D_j for each class is random, but it may be realized according to different patterns, giving rise to different problem formulations.

The two essential features that we consider here are:

- *Customer behavior*. In a perfectly segmented market, any passenger is willing to buy only one kind of class. Hence, we do not need to model passengers' preferences. On the contrary, if we want to account for preferences, we must define a choice model.

- *Timing of demand*. It would be nice if demand occurred sequentially, class 1 first and then down to class n. The worst case, in some sense, is when low-budget customers arrive first.

  A model where there exist disjoint time intervals, during which exactly one class is in demand, is called "static." The term "dynamic" is reserved to models where customer requests for different classes are interleaved over time.

We can express the decision policy in terms of maximum amount of seats available per class (protection levels). Alternatively, we could use bid-prices (i.e., the minimum price at which we are willing to sell a seat).

Demands for classes are assumed independent and occur sequentially, D_n first. No customer is willing to switch to another class.

Therefore, we may associate decision stages with classes, indexed in decreasing order of the index $j = n, n-1, \dots, 2, 1$.

The natural state variable is s_j, the integer number representing the residual capacity at the beginning of stage j; the initial state is $s_n = C$, the number of available seats on the aircraft.

In order to decide at each stage j how much of the remaining capacity is offered to class j, we need to find a sequence of value functions $V_j(s_j)$.

The overall objective is to find the maximum expected revenue that we may collect by selling the C available seats to the n classes, denoted by $V_n(C)$, and the boundary condition is

$$V_0(s_0) = 0, \qquad s_0 = 0, 1, \dots, C.$$

We do not consider groups or families and, given independence, there is no demand learning effect along the way.

A rather surprising finding is that we may build the DP recursion by assuming the following event sequence, for each class j:

1. First, we observe demand D_j for class j.
2. Then, we decide how many requests for class j to accept, a decision represented by the integer number x_j.
3. Then, we collect revenue $p_j x_j$ and proceed to stage $j-1$, with the updated state variable $s_{j-1} = s_j - x_j$.

This does not seem plausible, as we are supposed to make decisions *before* observing demand. However, we shall prove that the decision x_j does not rely on the probability distribution of D_j.

The intuition is that we do not really need to declare x_j beforehand. We may observe each individual request for class j sequentially, and each time we decide whether to accept it or not. If we reject a request, then we declare class j closed.

Formally, the DP recursion takes the swapped form that we introduced in Eq. (9):

$$V_j(s_j) = \mathsf{E}\left[\max_{0 \le x_j \le \min\{s_j, D_j\}} \Big(p_j x_j + V_{j-1}(s_j - x_j)\Big)\right].$$

As it turns out, it may be convenient to introduce an inconsequential index shift in j and omit the stage subscript in decision variable x and state s to ease notation:

$$V_{j+1}(s) = \mathsf{E}\left[\max_{0\leq x\leq \min\{s,D_{j+1}\}} \Big(p_{j+1}x + V_j(s-x)\Big)\right]. \tag{16}$$

The decision x at stage $j+1$ is naturally bounded by the residual capacity s and by the realized demand D_{j+1}.

In order to analyze the structure of the optimal policy and justify the above form, it is convenient to introduce the expected marginal value of capacity

$$\Delta V_j(s) \doteq V_j(s) - V_j(s-1),$$

for each stage j and residual capacity s.

Intuitively, the marginal value of capacity measures, for a given residual capacity s, the opportunity cost of an aircraft seat, i.e., how much revenue we are expected to lose if we give up a seat.

Then, we may rewrite Eq. (16) as

$$V_{j+1}(s) = V_j(s) + \mathsf{E}\left[\max_{0\leq x\leq \min\{s,D_{j+1}\}} \left\{\sum_{z=1}^{x} \big(p_{j+1} - \Delta V_j(s+1-z)\big)\right\}\right]. \tag{17}$$

This rewriting is based on a standard trick of the trade, i.e., a telescoping sum:

$$
\begin{aligned}
V_j(s-x) &= V_j(s) - \big[V_j(s) - V_j(s-x)\big] \\
&= V_j(s) - \big[V_j(s) \pm V_j(s-1) \pm \cdots \pm V_j(s+1-x) - V_j(s-x)\big] \\
&= V_j(s) - \sum_{z=1}^{x} \big[V_j(s+1-z) - V_j(s-z)\big] \\
&= V_j(s) - \sum_{z=1}^{x} \Delta V_j(s+1-z),
\end{aligned}
$$

where we use the shorthand $\pm V \equiv +V - V$. By plugging the last expression into Eq. (16) we easily obtain Eq. (17).

The following facts about expected marginal values can be proved:

$$
\Delta V_j(s+1) \le \Delta V_j(s), \qquad \Delta V_{j+1}(s) \ge \Delta V_j(s), \qquad \forall s, j.
$$

The idea is that that marginal values are decreasing with available capacity and that the value of capacity is larger when more stages are remaining.

As a consequence, the terms

$$
p_{j+1} - \Delta V_j(s+1-z)
$$

in the sum of Eq. (17) are decreasing with respect to z.

To find the optimal solution, we should keep increasing x, i.e., cumulating terms in the sum and accepting ticket requests for class $j+1$, until we find the first negative term or the upper bound of the sum, $\min\{s, D_{j+1}\}$, is reached.

We should keep accepting offers while the revenue p_{j+1} is larger than the opportunity cost measured by the marginal value of a seat ΔV_j for the next classes.

This defines an optimal *nested* protection level,

$$y_j^* \doteq \max\{y : p_{j+1} < \Delta V_j(y)\}, \qquad j = 1, \ldots, n-1. \tag{18}$$

This is the amount of seats reserved to classes $j, j-1, \ldots, 1$; the term "nested" is due to the fact that capacity is allocated to all classes higher than j included, rather than to a single class.

Equation (18) says that we should increase the protection level y_j until we find that the revenue p_{j+1} from the previous class $j+1$ compensates the opportunity cost of a seat.

The optimal protection level for these next classes constrains the number of tickets we may sell for class $j+1$, and we may express the optimal decision at stage $j+1$ as

$$x_{j+1}^*(s_{j+1}, D_{j+1}) = \min\{(s_{j+1} - y_j^*)^+, D_{j+1}\}.$$

This means that the maximum number of seats sold to class $j+1$ is the minimum between the observed demand and the excess residual capacity with respect to the protection level for the higher classes.

Indeed, we do not need to know D_{j+1} in advance, since we may keep satisfying demand for class $j+1$ until either the stage ends and we move on to the next class, or the protection level is reached, or we just run out of seats. This justifies the swap of expectation and maximization in Eq. (17).

Let us relax the assumption that demand for classes occur sequentially over disjoint time intervals. However, we still assume a rigid market segmentation.

Time is involved in the DP recursion, but a simple model is obtained if we assume that time intervals are small enough that we may observe at most one arrival per time interval.

Let us denote by $\lambda_j(t)$ the probability of an arrival for class j during time interval t, $t = 1, \ldots, T$. Since we assume a perfectly segmented market, these probabilities refer to independent events and must satisfy the consistency constraint:

$$\sum_{j=1}^{n} \lambda_j(t) \leq 1, \qquad \forall t.$$

The DP recursion in this case aims at finding value functions $V_t(s)$, where s is residual capacity, subject to the boundary conditions

$$\begin{aligned} V_t(0) &= 0, \qquad & t &= 1, \ldots, T, \\ V_{T+1}(s) &= 0, \qquad & s &= 0, 1, \ldots, C, \end{aligned}$$

where $T + 1$ denotes the end of the time horizon (when the aircraft takes off).

We may denote by $R(t)$ the available revenue at time t, which is a random variable taking value p_j when an arrival of class j occurs, 0 otherwise.

Whenever there is an arrival, we have to decide whether we accept the request or not, a decision that may be denoted by the binary decision variable x.

As in the static model, we do not need to make a decision in advance, since we just need to react to a request. Hence, the DP recursion takes again the "swapped" form

$$V_t(s) = \mathrm{E}\left\{ \max_{x \in \{0,1\}} \Big[R(t)x + V_{t+1}(s - x) \Big] \right\}.$$

Using expected marginal values of capacity again, a policy in terms of protection levels may be devised. In this case, however, protection levels may be time-varying.

If we relax the assumption of perfectly segmented markets, then we must define a model of customer choice.

One way of doing so is to partition the market into segments and to estimate the fraction of passengers belonging to each segment, as well as the probability that a class will be purchased by a passenger of each segment, if she is offered that class.

The control decision, at each time instant t, is the subset of classes that we offer.

Let $\mathcal{N}$ be the set of available classes, indexed by j and associated with revenue p_j. It is convenient to introduce $p_0 = 0$, the zero revenue earned when there is a passenger willing to fly but, given the subset of classes currently offered, she decides not to purchase any ticket.

Formally, a potential passenger request arrives at time t with probability λ and, given the offered subset of classes $\mathcal{S}_t \subseteq \mathcal{N} = \{1, \ldots, n\}$, she will make a choice.

Let $P_j(\mathcal{S}_t)$ be the probability that the customer chooses class j, as a function of the offered set, where we include the probability $P_0(\mathcal{S}_t)$ of no-purchase.

These probabilities may be estimated, based on the aforementioned market segmentation model, and they are subject to the natural constraints

$$
\begin{aligned}
& P_j(\mathcal{S}) \geq 0, && \mathcal{S} \subseteq \mathcal{N},\ j \in \mathcal{S} \cup \{0\} \\
& \sum_{j \in \mathcal{S}} P_j(\mathcal{S}) + P_0(\mathcal{S}) = 1, && \mathcal{S} \subseteq \mathcal{N}.
\end{aligned}
$$

In this model, the decision variable is actually a set, i.e., the subset $\mathcal{S}_t$ of classes offered at time t.

The decision-dependent purchase probabilities at time t are

$$
\begin{aligned}
&\lambda P_j(\mathcal{S}_t), && j = 1, \ldots, n \\
&(1-\lambda) + \lambda P_0(\mathcal{S}_t), && j = 0.
\end{aligned}
$$

In the last case, we may have a no purchase either because no one came, or because the passenger did not like the offered assortment of choices.

The DP recursion is

$$
V_t(s) = \max_{\mathcal{S}_t \subseteq \mathcal{N}} \left\{ \sum_{j \in \mathcal{S}_t} \lambda P_j(\mathcal{S}_t)\big(p_j + V_{t+1}(s-1)\big) + \big(\lambda P_0(\mathcal{S}_t) + 1 - \lambda\big) V_{t+1}(s) \right\}.
$$

In this case, we must lay our cards down on the table *before* the passenger's decision. As a consequence, the DP recursion is in the usual form, featuring the maximization of an expectation.