CHAPTER 102
Stochastic Optimization

ANTON J. KLEYWEGT
ALEXANDER SHAPIRO
Georgia Institute of Technology

1. INTRODUCTION

Decision makers often have to make decisions in the presence of uncertainty. Decision problems are often formulated as optimization problems, and thus in many situations decision makers wish to solve optimization problems that depend on parameters which are unknown. Typically, it is quite difficult to formulate and solve such problems, both conceptually and numerically. The difficulty already starts at the conceptual stage of modeling. Often there are a variety of ways in which the uncertainty can be formalized. In the formulation of optimization problems, one usually attempts to find a good trade-off between the realism of the optimization model, which affects the usefulness and quality of the obtained decisions, and the tractability of the problem, so that it can be solved analytically or numerically. As a result of these considerations, there are a large number of different approaches for formulating and solving optimization problems under uncertainty. It is impossible to give a complete survey of all such methods in one article. Therefore, this chapter aims only to give a flavor of prominent approaches to optimization under uncertainty.

2. OPTIMIZATION UNDER UNCERTAINTY

To describe some issues involved in optimization under uncertainty, we start with a static optimization problem. Suppose we want to maximize an objective function $G(x, \omega)$, where x denotes the decision

Handbook of Industrial Engineering: Technology and Operations Management, Third Edition.
Edited by Gavriel Salvendy

to be made, χ denotes the set of all feasible decisions, ω denotes an outcome that is unknown at the time the decision has to be made, and Ω denotes the set of all possible outcomes.

There are several approaches for dealing with optimization under uncertainty. Some of these approaches are illustrated next in the context of an example.

Example 1 (Newsvendor problem). Many companies sell seasonal products, such as fashion articles, airline seats, Christmas decorations, magazines, and newspapers. These products are characterized by a relatively short selling season, after which the value of the products decreases substantially. Often a decision has to be made how much of such a product to manufacture or purchase before the selling season starts. Once the selling season has started, there is not enough time remaining in the season to change this decision and implement the change, so that at this stage the quantity of the product is given. During the season the decision maker may be able to make other types of decisions to pursue desirable results, such as to change the price of the product as the season progresses and sales of the product take place. Such behavior is familiar in many industries. Another characteristic of such a situation is that the decisions have to be made before the eventual outcomes become known to the decision maker. For example, the decision maker has to decide how much of the product to manufacture or purchase before the demand for the product becomes known. Thus, decisions have to be made without knowing which outcome will take place.

Suppose that a manager has to decide how much of a seasonal product to order. Thus, the decision variable x is a nonnegative number representing the order quantity. The cost of the product to the company is c per unit of the product. During the selling season the product can be sold at a price (revenue) of r per unit of the product. After the selling season, any remaining product can be disposed of at a salvage value of s per unit of the product, where typically $s < r$. The demand D for the product is unkown at the time the order decision x has to be made. If the demand D turns out to be greater than the order quantity x, then the whole quantity x of the product is sold during the season, and no product remains at the end of the season, so that the total revenue and the profit turn out to be rx and $rx - cx = (r - c)x$, respectively. If the demand D turns out to be less than the order quantity x, then quantity D of the product is sold during the season, and the remaining amount of product at the end of the season is $x - D$, so that the total revenue and the profit turn out to be $rD + s(x - D)$ and $rD + s(x - D) - cx = (s - c)x + (r - s)D$, respectively. Thus, the profit is given by

$$G(x, D) = \begin{cases} (s - c)x + (r - s)D & \text{if } x \geq D \\ (r - c)x & \text{if } x < D \end{cases} \tag{1}$$

The manager would like to choose x to maximize the profit $G(x, D)$, but the dilemma is that D is unknown, or in other words is uncertain, at the time the decision should be made.

Note that if $r \leq c$ and $s \leq c$, then the company can make no profit from buying and selling the product, so that the optimal order quantity is $x^* = 0$, irrespective of what the demand D turns out to be. Also, if $s \geq c$, then any unsold product at the end of the season can be disposed of at a value at least equal to the cost of the product, so that it is optimal to order as much as possible, irrespective of what the demand D turns out to be. These, of course, are obvious cases. Therefore, we assume in the remainder of this example that $s < c < r$. Under this assumption, for any given $D \geq 0$, the function $G(\cdot, D)$ is a piecewise linear function with positive slope $r - c$ for $x < D$ and negative slope $s - c$ for $x > D$. Therefore, if the demand D is known at the time the order decision has to be made, then the best decision is to choose order quantity $x^* = D$.

However, if D is not known, then the problem becomes more difficult. Sometimes a manager may want to hedge against the worst possible outcome. Suppose the manager thinks that the demand D will turn out to be some number in the interval $[a, b]$ with $a < b$, that is, the lower and upper bounds for the demand are known to the manager. In that case, in order to hedge against the worst possible scenario, the manager will choose the value of x that gives the best profit under the worst possible outcome. That is, the manager will maximize the function $g(x) \equiv \min_{D \in [a,b]} G(x, D)$ over $x \geq 0$. This leads to the following max-min problem:

$$\max_{x \geq 0} \min_{D \in [a,b]} G(x, D) \tag{2}$$

It is not difficult to see that $g(x) = G(x, a)$, and hence $x^* = a$ is the optimal solution from the point of view of the worst-case scenario. Clearly, in many cases this will be an overly conservative decision.

Sometimes a manager may want to make the decision that under the worst possible outcome will still appear as good as possible compared with what would have been the best decision with hindsight, that is, after the outcome becomes known. For any outcome of the demand D, let

$$g^*(D) \equiv \max_{x \geq 0} G(x, D) = (r - c)D$$

denote the optimal profit with hindsight, also called the optimal value with perfect information. The optimal decision with perfect information, $x^* = D$, is sometimes called the wait-and-see solution. Suppose the manager chose to order quantity x, so that the actual profit turned out to be $G(x, D)$. The amount of profit that the company missed out on because of a suboptimal decision is given by $g^*(D) - G(x, D)$. This quantity is often called the absolute regret. The manager may want to choose the value of x that minimizes the absolute regret under the worst possible outcome. For any decision x, the worst possible outcome is given by $\max_{D \in [a,b]} [g^*(D) - G(x, D)]$. Since the manager wants to choose the value of x that minimizes the absolute regret under the worst possible outcome, this leads to the following min-max problem:

$$\min_{x \geq 0} \max_{D \in [a,b]} [g^*(D) - G(x, D)] \tag{3}$$

The optimal solution of this problem is $x^* = [(c - s)a + (r - c)b]/(r - s)$. Note that x^* is a convex combination of a and b, and thus $a < x^* < b$. The larger the salvage loss per unit $c - s$, the closer x^* is to a, and the larger the profit per unit $r - c$, the closer x^* is to b. That seems to be a more reasonable decision than $x^* = a$.

It was assumed in both variants of the worst-case approach discussed above that no a priori information about the demand D was available to the manager except the lower and upper bounds for the demand. In some situations this may be a reasonable assumption and the worst-case approach could make sense if the range of the demand is known and is not too large.

Another approach to decision making under uncertainty, different from the worst-case approaches described above, is the stochastic optimization approach, on which we focus in the remainder of this article. Suppose that the demand D can be viewed as a *random variable*. This means that the probability distribution of D is known, or at least can be estimated, by using historical data and/or a priori information available to the manager. Let $F(w) \equiv \mathbb{P}(D \leq w)$ be the corresponding cumulative distribution function (cdf) of D. Then one can try to optimize the objective function on *average*, that is, to maximize the expected profit $\mathbb{E}[G(x, D)] = \int_0^\infty G(x, w)\, dF(w)$. This leads to the stochastic program

$$\max_{x \geq 0} \{g(x) \equiv \mathbb{E}[G(x, D)]\} \tag{4}$$

In the present case it is not difficult to solve the above optimization problem in a closed form. For any $D \geq 0$, the function $G(\cdot, D)$ is concave (and piecewise linear). Therefore, the expected value function $g(\cdot)$ is also concave. Suppose for a moment that $F(\cdot)$ is continuous at a point $x > 0$. Then

$$g(x) = \int_0^x [(s - c)x + (r - s)w]\, dF(w) + \int_x^\infty (r - c)x\, dF(w)$$

Using integration by parts it is possible to calculate that

$$g(x) = (r - c)x - (r - s) \int_0^x F(w)\, dw \tag{5}$$

The function $g(\cdot)$ is concave, and hence continuous, and therefore Eq. (5) holds even if $F(\cdot)$ is discontinuous at x. It follows that $g(\cdot)$ is differentiable at x iff $F(\cdot)$ is continuous at x, in which case

$$g'(x) = r - c - (r - s)F(x) \tag{6}$$

Consider the inverse $F^{-1}(\alpha) \equiv \min\{x : F(x) \geq \alpha\}$ function of the cdf F, which is defined for $\alpha \in (0, 1)$. ($F^{-1}(\alpha)$ is called the α-quantile of the cdf F.) Since $g(\cdot)$ is concave, a necessary and sufficient condition for $x^* > 0$ to be an optimal solution of problem (4), is that $g'(x^*) = 0$, provided that $g(\cdot)$ is differentiable at x^*. Note that because $s < c < r$, it follows that $0 < (r - c)/(r - s) < 1$. Consequently, an optimal solution of (4) is given by

$$x^* = F^{-1}\left(\frac{r - c}{r - s}\right) \tag{7}$$

This holds even if $F(\cdot)$ is discontinuous at x^*.

Clearly the above approach explicitly depends on knowledge of the probability distribution of the demand D. In practice, the corresponding cdf $F(\cdot)$ is never known exactly and can be approximated (estimated) at best. Nevertheless, often the obtained optimal solution is robust with respect to perturbations of the cdf $F(\cdot)$.

Another point worth mentioning is that by solving (4), the manager tries to optimize the profit on average. However, the realized profit $G(x^*, D)$ could be very different from the corresponding expected value $g(x^*)$, depending on the particular realization of the demand D. This may happen if $G(x^*, D)$, considered as a random variable, has a large variability that could be measured by its variance $\mathrm{Var}[G(x^*, D)]$. Therefore, if the manager wants to hedge against such variability, he may consider the following optimization problem

$$\max_{x \geq 0} \{g_\beta(x) \equiv \mathbb{E}[G(x, D)] - \beta \, \mathrm{Var}[G(x, D)]\} \tag{8}$$

The coefficient $\beta \geq 0$ represents the weight given to the conservative part of the decision. If β is large, then the above optimization problem tries to find a solution with minimal profit variance, while if $\beta = 0$, then problem (8) coincides with problem (4). Note that since the variance $\mathrm{Var}[G(x, D)] \equiv \mathbb{E}[(G(x, D) - \mathbb{E}[G(x, D)])^2]$ is itself an expected value, from a mathematical point of view, problem (8) is similar to the expected value problem (4). Thus, the problem of optimizing the expected value of an objective function $G(x, D)$ is very general—it could include the means, variances, quantiles, and almost any other aspects of random variables of interest.

The following deterministic optimization approach is also often used for decision making under uncertainty. The random variable D is replaced by its mean $\mu = \mathbb{E}[D]$, and then the following deterministic optimization problem is solved:

$$\max_{x \geq 0} G(x, \mu) \tag{9}$$

A resulting optimal solution $\bar{x}$ is sometimes called an expected value solution. Of course, this approach requires that the mean of the random variable D be known to the decision maker. In the present example, the optimal solution of this deterministic optimization problem is $\bar{x} = \mu$. Note that the mean solution $\bar{x}$ can be very different from the solution x^* given in (7). It is well known that the quantiles are much more stable to variations of the cdf F than the corresponding mean value. Therefore, the optimal solution x^* of the stochastic optimization problem is more robust with respect to variations of the probability distributions than an optimal solution $\bar{x}$ of the corresponding deterministic optimization problem. This should be not surprising, since the deterministic problem (9) can be formulated in the framework of the stochastic optimization problem (4) by considering the trivial distribution of D being identically equal to μ.

Also note that, for any x, $G(x, D)$ is concave in D. As a result, it follows from Jensen's inequality that $G(x, \mu) \geq \mathbb{E}[G(x, D)]$, and hence

$$\max_{x \geq 0} G(x, \mu) \geq \max_{x \geq 0} \mathbb{E}[G(x, D)]$$

Thus, the optimal value of the deterministic optimization problem is biased upward relative to the optimal value of the stochastic optimization problem.

One can also try to solve the optimization problem

$$\max_{x \geq 0} G(x, D) \tag{10}$$

for different realizations of D, and then take the expected value of the obtained solutions as the final solution. In the present example, for any realization D, the optimal solution of (10) is $x = D$, and hence the expected value of these solutions, and final solution, is $\bar{x} = \mathbb{E}[D]$. Note that this approach does not have a clear rationale, and moreover, in many optimization problems it may not make sense to take the expected value of the obtained solutions. This is usually the case in optimization problems with discrete solutions, for example, when a solution is a path in a network, there does not seem to be a useful way to take the average of several different paths. Therefore, we do not discuss this approach further.

3. STOCHASTIC PROGRAMMING

The discussion of the above example motivates us to introduce the following model optimization problem, referred to as a *stochastic programming* problem:

$$\min_{x \in \chi} \{g(x) \equiv \mathbb{E}[G(x, \omega)]\} \tag{11}$$

(We consider a minimization rather than a maximization problem for the sake of notational convenience.) Here $\chi \subset \mathbb{R}^n$ is a set of permissible values of the vector x of decision variables and is referred to as the feasible set of problem (11). Often χ is defined by a (finite) number of smooth (or even linear) constraints. In some other situations the set χ is finite. In that case problem (11) is called a *discrete* stochastic optimization problem (this should not be confused with the case of discrete probability distributions). Variable ω represents random (or stochastic) aspects of the problem. Often ω can be modeled as a finite dimensional random vector, or in more involved cases as a random process. In the abstract framework we can view ω as an element of the probability space $(\Omega, \mathcal{F}, P)$ with the known probability measure (distribution) P.

It is also possible to consider the following extensions of the basic problem (11).

- One may need to optimize a function of the expected value function $g(x)$. This happened, for example, in problem (8), where the manager wanted to optimize a linear combination of the expected value and the variance of the profit. Although important from a modeling point of view, such an extension usually does not introduce additional technical difficulties into the problem.
- The feasible set can also be defined by constraints given in a form of expected value functions. For example, suppose that we want to optimize an objective function subject to the constraint that the event $\{h(x, W) \geq 0\}$, where W is a random vector with a known probability distribution and $h(\cdot, \cdot)$ is a given function, should happen with a probability not bigger than a given number $p \in (0, 1)$. Probability of this event can be represented as the expected value $\mathbb{E}[\psi(x, W)]$, where

$$\psi(x, w) \equiv \begin{cases} 1 & \text{if} \quad h(x, w) \geq 0 \\ 0 & \text{if} \quad h(x, w) < 0 \end{cases}$$

Therefore, this constraint can be written in the form $\mathbb{E}[\psi(x, W)] \leq p$. Problems with such probabilistic constraints are called *chance constrained* problems. Note that even if the function $h(\cdot, \cdot)$ is continuous, the corresponding indicator function $\psi(\cdot, \cdot)$ is discontinuous unless it is identically equal to zero or one. Because of that, it may be technically difficult to handle such a problem.

- In some cases the involved probability distribution P_θ depends on parameter vector θ, whose components also represent decision variables. That is, the expected value objective function is given in the form

$$g(x, \theta) \equiv \mathbb{E}_\theta[G(x, \omega)] = \int_\Omega G(x, \omega)\, dP_\theta(\omega) \tag{12}$$

By using a transformation it is sometimes possible to represent the above function $g(\cdot)$ as the expected value of a function, depending on x and θ, with respect to a probability distribution that is independent of θ. We shall discuss such likelihood ratio transformations in Section 3.4

The above formulation of stochastic programs is somewhat too general and abstract. In order to proceed with a useful analysis we need to identify particular classes of such problems that on one hand are interesting from the point of view of applications and on the other hand are computationally tractable. In the following sections we introduce several classes of such problems and discuss various techniques for their solution.

3.1. Stochastic Programming with Recourse

Consider again problem (4) of the newsvendor example. We may view that problem as a two-stage problem. At the first stage a decision should be made about the quantity x to order. At this stage the demand D is not known. At the second stage a realization of the demand D becomes known and, given the first stage decision x, the manager makes a decision about the quantities y and z to sell at prices r and s, respectively. Clearly the manager would like to choose y and z in such a way as to maximize the profit. It is possible to formulate the second stage problem as the simple linear program

$$\max_{y,z} ry + sz \quad \text{subject to } y \leq D,\ y + z \leq x,\ y \geq 0,\ z \geq 0 \tag{13}$$

The optimal solution of the above problem (13) is $y^* = \min\{x, D\}$, $z^* = \max\{x - D, 0\}$, and its

optimal value is the profit $G(x, D)$ defined in (1). Now at the first stage, before a realization of the demand D becomes known, the manager chooses a value for the first-stage decision variable x by maximizing the expected value of the second-stage optimal profit $G(x, D)$.

This is the basic idea of a two-stage stochastic program with recourse. At the first stage, before a realization of the random variables ω becomes known, one chooses the first-stage decision variables x to optimize the expected value $g(x) \equiv \mathbb{E}[G(x, \omega)]$ of an objective function $G(x, \omega)$ that depends on the optimal second stage objective function.

A *two-stage stochastic linear program with fixed recourse* is a two-stage stochastic program of the form

$$\begin{aligned} &\min_x c^T x + \mathbb{E}[Q(x, \xi)] \\ &\text{s.t. } Ax = b,\ x \geq 0 \end{aligned} \tag{14}$$

where $Q(x, \xi)$ is the optimal value of the second-stage problem

$$\begin{aligned} &\min_y q(\omega)^T y \\ &\text{s.t. } T(\omega)x + Wy = h(\omega),\ y \geq 0 \end{aligned} \tag{15}$$

The second-stage problem depends on the data $\xi(\omega) \equiv (q(\omega), h(\omega), T(\omega))$, elements of which can be random, while the matrix W is assumed to be known beforehand. The matrices $T(\omega)$ and W are called the *technology* and *recourse* matrices, respectively. The expectation $\mathbb{E}[Q(x, \xi)]$ is taken with respect to the random vector $\xi = \xi(\omega)$, whose probability distribution is assumed to be known. The above formulation originated in the works of Dantzig (1955) and Beale (1955).

Note that the optimal solution $y^* = y^*(\omega)$ of the second-stage problem (15) depends on the random data $\xi = \xi(\omega)$ and therefore is random. One can write $Q(x, \xi(\omega)) = q(\omega)^T y^*(\omega)$.

The next question is how one can solve the above two-stage problem numerically. Suppose that the random data have a *discrete* distribution with a finite number K of possible realizations $\xi_k = (q_k, h_k, T_k)$, $k = 1, \ldots, K$, (sometimes called *scenarios*), with the corresponding probabilities p_k. In that case, $\mathbb{E}[Q(x, \xi)] = \sum_{k=1}^K p_k Q(x, \xi_k)$, where

$$Q(x, \xi_k) = \min \{q_k^T y_k : T_k x + W y_k = h_k,\ y_k \geq 0\}$$

Therefore, the above two-stage problem can be formulated as one large linear program:

$$\begin{aligned} &\min\ c^T x + \textstyle\sum_{k=1}^K p_k q_k^T y_k \\ &\text{s.t. } Ax = b \\ &\qquad T_k x + W y_k = h_k \\ &\qquad x \geq 0,\ y_k \geq 0,\ k = 1, \ldots, K \end{aligned} \tag{16}$$

The linear program (16) has a certain block structure that makes it amenable to various decomposition methods. One such decomposition method is the popular L-shaped method developed by Van Slyke and Wets (1969). We refer the interested reader to the recent books by Kall and Wallace (1994) and Birge and Louveaux (1997) for a thorough discussion of stochastic programming with recourse.

The above numerical approach works reasonably well if the number K of scenarios is not too large. Suppose, however, that the random vector ξ has m independently distributed components, each having just three possible realizations. Then the total number of different scenarios is $K = 3^m$. That is, the number of scenarios grows exponentially fast in the number m of random variables. In that case, even for a moderate number of random variables, say $m = 100$, the number of scenarios becomes so large that even modern computers cannot cope with the required calculations. It seems that the only way to deal with such exponential growth of the number of scenarios is to use sampling. Such approaches are discussed in Section 3.2.

It may also happen that some of the decision variables at the first or second stage are integers, such as binary variables representing "yes" or "no" decisions. Such integer (or discrete) stochastic programs are especially difficult to solve, and only very moderate progress has been reported so far. A discussion of two-stage stochastic integer programs with recourse can be found in Birge and Louveaux (1997). A branch and bound approach for solving stochastic discrete optimization problems was suggested by Norkin et al. (1998). Schultz et al. (1998) suggested an algebraic approach for solving stochastic programs with integer recourse by using a framework of Gröbner basis reductions. For a recent survey of mainly theoretical results on stochastic integer programming see Klein Haneveld and Van der Vlerk (1999).

Conceptually the idea of two-stage programming with recourse can be readily extended to *multistage* programming with recourse. Such an approach tries to model the situation where decisions

are made periodically (in stages) based on currently known realizations of some of the random variables. An H-stage stochastic linear program with fixed recourse can be written in the form

$$\begin{aligned} &\min c^1x^1 + \mathbb{E}\{\min c^2(\omega)x^2(\omega) + \ldots + \mathbb{E}[\min c^H(\omega)x^H(\omega)]\} \\ &\text{s.t. } W^1x^1 = h^1 \\ &\quad T^1(\omega)x^1 + W^2x^2(\omega) = h^2(\omega) \\ &\qquad \ldots\ldots \\ &\quad T^{H-1}(\omega)x^{H-1}(\omega) + W^Hx^H(\omega) = h^H(\omega) \\ &\quad x^1 \geq 0,\ x^2(\omega) \geq 0,\ \ldots,\ x^H(\omega) \geq 0 \end{aligned} \tag{17}$$

The decision variables $x^2(\omega), \ldots, x^H(\omega)$ are allowed to depend on the random data ω. However, the decision $x^t(\omega)$ at time t can only depend on the part of the random data that is known at time t (these restrictions are often called nonanticipativity constraints). The expectations are taken with respect to the distribution of the random variables whose realizations are not yet known.

Again, if the distribution of the random data is discrete with a finite number of possible realizations, then problem (17) can be written as one large linear program. However, it is clear that even for a small number of stages and a moderate number of random variables the total number of possible scenarios will be astronomical. Therefore, a current approach to such problems is to generate a reasonable number of scenarios and solve the corresponding (deterministic) linear program, hoping to catch at least the flavor of the stochastic aspect of the problem. The argument is that the solution obtained in this way is more robust than the solution obtained by replacing the random variables with their means.

Often the same practical problem can be modeled in different ways. For instance, one can model a problem as a two-stage stochastic program with recourse, putting all random variables whose realizations are not yet known at the second stage of the problem. Then, as realizations of some of the random variables become known, the solutions are periodically updated in a two-stage rolling horizon fashion, every time by solving an updated two-stage problem. Such an approach is different from a multistage program with recourse, where every time a decision is to be made, the modeler tries to take into account that decisions will be made at several stages in the future.

3.2. Sampling Methods

In this section we discuss a different approach that uses Monte Carlo sampling techniques to solve stochastic optimization problems.

Example 2. Consider a stochastic process I_t, $t = 1, 2, \ldots$, governed by the recursive equation

$$I_t = [I_{t-1} + x_t - D_t]^+ \tag{18}$$

with initial value I_0. Here D_t are random variables and x_t represent decision variables. (Note that $[a]^+ \equiv \max\{a, 0\}$.) The above process I_t can describe the waiting time of the tth customer in a $G/G/1$ queue, where D_t is the interarrival time between the $(t-1)$th and tth customers and x_t is the service time of $(t-1)$th customer. Alternatively, I_t may represent the inventory of a certain product at time t, with D_t and x_t representing the demand and production (or ordering) quantities, respectively, of the product at time t.

Suppose that the process is considered over a finite horizon with time periods $t = 1, \ldots, T$. Our goal is to minimize (or maximize) the expected value of an objective function involving $I_1, \ldots, I_T$. For instance, one may be interested in maximizing the expected value of a profit given (Albritton et al. 1999);

$$\begin{aligned} G(x, W) &\equiv \Sigma_{t=1}^{T} \{\pi_t \min[I_{t-1} + x_t, D_t] - h_tI_t\} \\ &= \Sigma_{t=1}^{T} \pi_tx_t + \Sigma_{t=1}^{T-1} (\pi_{t+1} - \pi_t - h_t)I_t + \pi_1I_0 - (\pi_T + h_T)I_T \end{aligned} \tag{19}$$

Here $x = (x_1, \ldots, x_T)$ is a vector of decision variables, $W = (D_1, \ldots, D_T)$ is a random vector of the demands at periods $t = 1, \ldots, T$, and π_t and h_t are nonnegative parameters representing the marginal profit and the holding cost, respectively, of the product at period t.

If the initial value I_0 is sufficiently large, then with probability close to one, variables $I_1, \ldots, I_T$ stay above zero. In that case $I_1, \ldots, I_T$ become linear functions of the random data vector W, and hence components of the random vector W can be replaced by their means. However, in many practical situations the process I_t hits zero with high probability over the considered horizon T. In such cases the corresponding expected value function $g(x) \equiv \mathbb{E}[G(x, W)]$ cannot be written in a closed

form. One can use a Monte Carlo simulation procedure to evaluate $g(x)$. Note that for any given realization of D_t, the corresponding values of I_t, and hence the value of $G(x, W)$, can be easily calculated using the iterative formula (18).

That is, let $W^i = (D_1^i, \ldots, D_T^i)$, $i = 1, \ldots, N$, be a random (or pseudorandom) sample of N independent realizations of the random vector W generated by computer, that is, there are N generated realizations of the demand process D_t, $t = 1, 2, \ldots, T$, over the horizon t. Then for any given x the corresponding expected value $g(x)$ can be approximated (estimated) by the sample average

$$\hat{g}_N(x) \equiv \frac{1}{N} \sum_{i=1}^{N} G(x, W^i) \tag{20}$$

We have that $\mathbb{E}[\hat{g}_N(x)] = g(x)$, and by the law of large numbers, that $\hat{g}_N(x)$ converges to $g(x)$ with probability one (w.p.1) as $N \to \infty$. That is, $\hat{g}_N(x)$ is an *unbiased* and *consistent* estimator of $g(x)$.

Any reasonably efficient method for optimizing the expected value function $g(x)$, say by using its sample average approximations, is based on estimation of its first (and maybe second) order derivatives. This has an independent interest and is called *sensitivity* or *perturbation* analysis. We will discuss that in Section 3.3. Recall that $\nabla g(x) \equiv (\partial g(x)/\partial x_1, \ldots, \partial g(x)/\partial x_T)$ is called the gradient vector of $g(\cdot)$ at x.

It is possible to consider a stationary distribution of the process I_t (if it exists) and to optimize the expected value of an objective function with respect to the stationary distribution. Typically, such a stationary distribution cannot be written in a closed form and is difficult to compute accurately. This introduces additional technical difficulties into the problem. Also, in some situations the probability distribution of the random variables D_t is given in a parametric form whose parameters are decision variables. We will discuss dealing with such cases later.

3.3. Perturbation Analysis

Consider the expected value function $g(x) \equiv \mathbb{E}[G(x, \omega)]$. An important question is under which conditions the first order derivatives of $g(x)$ can be taken inside the expected value, that is, under which conditions the equation

$$\nabla g(x) \equiv \nabla \mathbb{E}[G(x, \omega)] = \mathbb{E}[\nabla_x G(x, \omega)] \tag{21}$$

is correct. One reason why this question is important is the following. Let $\omega^1, \ldots, \omega^N$ denote a random sample of N independent realizations of the random variable with common probability distribution P, and let

$$\hat{g}_N(x) \equiv \frac{1}{\mathrm{N}} \sum_{i=1}^{N} G(x, \omega^i) \tag{22}$$

be the corresponding sample average function. If the interchangeability equation (21) holds, then

$$\mathbb{E}[\nabla \hat{g}_N(x)] = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}[\nabla_x G(x, \omega^i)] = \frac{1}{N} \sum_{i=1}^{N} \nabla \mathbb{E}[G(x, \omega^i)] = \nabla g(x) \tag{23}$$

and hence $\nabla \hat{g}_N(x)$ is an unbiased and consistent estimator of $\nabla g(x)$.

Let us observe that in both examples 1 and 2 the function $G(\cdot, \omega)$ is piecewise linear for any realization of ω, and hence is not everywhere differentiable. The same holds for the optimal value function $Q(\cdot, \xi)$ of the second-stage problem (15). If the distribution of the corresponding random variables is discrete, then the resulting expected value function is also piecewise linear and hence is not everywhere differentiable.

On the other hand, expectation with respect to a continuous distribution typically smoothes the corresponding function and in such cases Eq. (21) often is applicable. It is possible to show that if the following two conditions hold at a point x, then $g(\cdot)$ is differentiable at x and Eq. (21) holds:

1. The function $G(\cdot, \omega)$ is differentiable at x w.p.1.
2. There exists a positive valued random variable $K(\omega)$ such that $\mathbb{E}[K(\omega)]$ is finite and the inequality

$$|G(x_1, \omega) - G(x_2, \omega)| \leq K(\omega)\|x_1 - x_2\| \tag{24}$$

holds w.p.1 for all x_1, x_2 in a neighborhood of x.

If the function $G(\cdot, \omega)$ is not differentiable at x w.p.1 (i.e., for P-almost every $\omega \in \Omega$), then the right-hand side of Eq. (21) does not make sense. Therefore, clearly the above condition (1) is necessary for (21) to hold. Note that condition (1) requires $G(\cdot, \omega)$ to be differentiable w.p.1 at the given (fixed) point x and does not require differentiability of $G(\cdot, \omega)$ everywhere. The second condition (ii) requires $G(\cdot, \omega)$ to be continuous (in fact Lipschitz continuous) w.p.1 in a neighborhood of x.

Consider, for instance, function $G(x, D)$ of example 1 defined in 1. For any given D, the function $G(\cdot, D)$ is piecewise linear and differentiable at every point x except at $x = D$. If the cdf $F(\cdot)$ of D is continuous at x, then the probability of the event $\{D = x\}$ is zero and hence the interchangeability equation (21) holds. Then $\partial G(x, D)/\partial x$ is equal to $s - c$ if $x > D$, and is equal to $r - c$ if $x < D$. Therefore, if $F(\cdot)$ is continuous at x, then $G(\cdot, D)$ is differentiable at x and

$$g'(x) = (s - c)\mathbb{P}(D < x) + (r - c)\mathbb{P}(D > x)$$

which gives the same equation as (6). Note that the function $\partial G(\cdot, D)/\partial x$ is discontinuous at $x = D$. Therefore, the second order derivative of $\mathbb{E}[G(\cdot, D)]$ cannot be taken inside the expected value. Indeed, the second order derivative of $G(\cdot, D)$ is zero whenever it exists. Such behavior is typical in many interesting applications.

Let us calculate the derivatives of the process I_t, defined by the recursive equation (18), for a particular realization of the random variables D_t. Let τ_1 denote the first time that the process I_t hits zero, that is, $\tau_1 \geq 1$ is the first time $I_{\tau_1 - 1} + x_{\tau_1} - D_{\tau_1}$ becomes less than or equal to zero, and hence $I_{\tau_1} = 0$. Let $\tau_2 > \tau_1$ be the second time that I_t hits zero, etc. Note that if $I_{\tau_1 + 1} = 0$, then $\tau_2 = \tau_1 + 1$, and so on. Let $1 \leq \tau_1 < \ldots < \tau_n \leq T$ be the sequence of hitting times. (In queueing terminology, τ_i represents the starting time of a new busy cycle of the corresponding queue.) For a given time $t \in \{1, \ldots, T\}$, let $\tau_{i-1} \leq t < \tau_i$ Suppose that the events $\{I_{\tau - 1} + x_\tau - D_\tau = 0\}$, $\tau = 1, \ldots, T$, occur with probability zero. Then, for almost every W, the gradient of I_s with respect to the components of vector x_t can be written as follows:

$$\nabla_{x_t} I_s = \begin{cases} 1 & \text{if } t \leq s < \tau_i \text{ and } t \neq \tau_{i-1} \\ 0 & \text{otherwise} \end{cases} \tag{25}$$

Thus, by using Eqs. (19) and (25), one can calculate the gradient of the sample average function $\hat{g}_N(\cdot)$ of example (2), and hence one can consistently estimate the gradient of the expected value function $g(\cdot)$.

Consider the process I_t defined by the recursive equation (18) again. Suppose now that variables x_t do not depend on t, and let x denote their common value. Suppose further that D_t, $t = 1, \ldots,$ are independently and identically distributed with mean $\mu > 0$. Then for $x < \mu$ the process I_t is stable and has a stationary (steady-state) distribution. Let $g(x)$ be the steady-state mean (the expected value with respect to the stationary distribution) of the process $I_t = I_t(x)$. By the theory of regenerative processes it follows that for every $x \in (0, \mu)$ and any realization (called sample path) of the process D_t, $t = 1, \ldots,$ the-long run average $\hat{g}_T(x) \equiv \Sigma_{t=1}^T I_t(x)/T$ converges w.p.1 to $g(x)$ as $T \to \infty$. It is possible to show that $\nabla \hat{g}_T(x)$ also converges w.p.1 to $\nabla g(x)$ as $T \to \infty$. That is, by differentiating the long-run average of a sample path of the process I_t we obtain a consistent estimate of the corresponding derivative of the steady-state mean $g(x)$. Note that $\nabla I_t(x) = t - \tau_{i-1}$ for $\tau_{i-1} \leq t < \tau_i$, and hence the derivative of the long-run average of a sample path of the process I_t can be easily calculated.

The idea of differentiation of a sample path of a process in order to estimate the corresponding derivative of the steady-state mean function by a single simulation run is at the heart of *infinitesimal perturbation analysis*. We refer the interested reader to Glasserman (1991) and Ho and Cao (1991) for a thorough discussion of that topic.

3.4. Likelihood Ratio Method

The Monte Carlo sampling approach to derivative estimation introduced in Section 3.3 does not work if the function $G(\cdot, \omega)$ is discontinuous or if the corresponding probability distribution also depends on decision variables. In this section we discuss an alternative approach to derivative estimation known as the *likelihood ratio* (or *score function*) method.

Suppose that the expected value function is given in the form $g(\theta) \equiv \mathbb{E}_\theta[G(W)]$, where W is a random vector whose distribution depends on the parameter vector θ. Suppose further that the distribution of W has a probability density function (pdf) $f(\theta, w)$. Then for a chosen pdf $\phi(w)$ we can write

$$\mathbb{E}_\theta[G(W)] = \int G(w) f(\theta, w)\, dw = \int G(w) \frac{f(\theta, w)}{\phi(w)} \phi(w)\, dw$$

and hence

$$g(\theta) = \mathbb{E}_{\phi}[G(Z)L(\theta, Z)] \tag{26}$$

where $L(\theta, z) \equiv f(\theta, z)/\phi(z)$ is the so-called likelihood ratio function, $Z \sim \phi(\cdot)$ and $\mathbb{E}_{\phi}[\cdot]$ means that the expectation is taken with respect to the pdf ϕ. We assume in the definition of the likelihood ratio function that $0/0 = 0$ and that the pdf ϕ is such that if $\phi(w)$ is zero for some w, then $f(\theta, w)$ is also zero, that is, we do not divide a positive number by zero.

The expected value in the right-hand side of (26) is taken with respect to the distribution ϕ, which does not depend on the vector θ. Therefore, under appropriate conditions ensuring interchangeability of the differentiation and integration operators, we can write

$$\nabla g(\theta) = \mathbb{E}_{\phi}[G(Z)\nabla_{\theta}L(\theta, Z)] \tag{27}$$

In particular, if for a given θ_0 we choose $\phi(\cdot) \equiv f(\theta_0, \cdot)$, then $\nabla_{\theta}L(\theta, z) = \nabla_{\theta}f(\theta, z)/f(\theta_0, z)$, and hence $\nabla_{\theta}L(\theta_0, z) = \nabla_{\theta}\ln[f(\theta_0, z)]$. The function $\nabla_{\theta}\ln[f(\theta, z)]$ is called the score function; thus the name of this technique.

Now by generating a random sample $Z^1, \ldots, Z^N$ from the pdf $\phi(\cdot)$, one can estimate $g(\theta)$ and $\nabla g(\theta)$ by the respective sample averages

$$\tilde{g}_N(\theta) \equiv \frac{1}{N}\sum_{i=1}^{N} G(Z^i)L(\theta, Z^i) \tag{28}$$

$$\nabla \tilde{g}_N(\theta) \equiv \frac{1}{N}\sum_{i=1}^{N} G(Z^i)\nabla_{\theta}L(\theta, Z^i) \tag{29}$$

This can be readily extended to situations where function $G(x, W)$ also depends on decision variables.

Typically, the density functions used in applications depend on the decision variables in a smooth and even analytic way. Therefore, usually there is no problem in taking derivatives inside the expected value in the right-hand side of (26). When applicable, the likelihood ratio method often also allows estimation of second and higher order derivatives. However, note that the likelihood ratio method is notoriously unstable and a bad choice of the pdf ϕ may result in huge variances of the corresponding estimators. This should not be surprising since the likelihood ratio function may involve divisions by very small numbers, which of course is a very unstable procedure. We refer to Glynn (1990) and Rubinstein and Shapiro (1993) for a further discussion of the likelihood ratio method.

As an example consider the optimal value function of the second stage problem (15). Suppose that only the right-hand-side vector $h = h(\omega)$ of the second-stage problem is random. Then $Q(x, h) = G(h - Tx)$, where $G(\chi) \equiv \min\{q^T y : Wy = \chi, y \geq 0\}$. Suppose that the random vector h has a pdf $f(\cdot)$. By using the transformation $z = h - Tx$, we obtain

$$\mathbb{E}_f[Q(x, h)] = \int G(\eta - Tx)f(\eta)\, d\eta = \int G(z)f(z + Tx)\, dz = \mathbb{E}_{\phi}[G(Z)L(x, Z)] \tag{30}$$

Here ϕ is a chosen pdf, z is a random vector having pdf ϕ, and $L(x, z) \equiv f(z + Tx)/\phi(z)$ is the corresponding likelihood ratio function. It can be shown by duality arguments of linear programming that $G(\cdot)$ is a piecewise linear convex function. Therefore, $\nabla_x Q(x, h)$ is piecewise constant and discontinuous, and hence second order derivatives of $\mathbb{E}_f[Q(x, h)]$ cannot be taken inside the expected value. On the other hand, the likelihood ratio function is as smooth as the pdf $f(\cdot)$. Therefore, if $f(\cdot)$ is twice differentiable, then the second order derivatives can be taken inside the expected value in the right-hand side of (30), and consequently the second order derivatives of $\mathbb{E}_f[Q(x, h)]$ can be consistently estimated by a sample average.

3.5. Simulation-Based Optimization Methods

There are basically two approaches to the numerical solution of stochastic optimization problems by using Monte Carlo sampling techniques. One approach is known as the *stochastic approximation* method and originated in Robbins and Monro (1951). The other method was discovered and rediscovered by different researchers and is known under various names.

Suppose that the feasible set χ is convex and that at any point $x \in \chi$ an estimate $\hat{\gamma}(x)$ of the gradient $\nabla g(x)$ can be computed, say by a Monte Carlo simulation method. The stochastic approximation method generates the iterates by the recursive equation

$$x_{\nu+1} = \Pi_{\chi}(x_{\nu} - \alpha_{\nu}\hat{\gamma}(x_{\nu})) \tag{31}$$

where $\alpha_{\nu} > 0$ are chosen step sizes and Π_{χ} denotes the projection onto the set χ, that is, $\Pi_{\chi}(x)$ is the point in χ closest to x. Under certain regularity conditions the iterates x_{ν} converge to a locally

optimal solution of the corresponding stochastic optimization problem, that is, to a local minimizer x^* of $g(x)$ over χ. Typically, in order to guarantee this convergence the following two conditions are imposed on the step sizes: (1) $\Sigma_{\nu=1}^{\infty} \alpha_\nu = \infty$, and (2) $\Sigma_{\nu=1}^{\infty} \alpha_\nu^2 < \infty$. For example, one can take $\alpha_\nu \equiv c/\nu$ for some $c > 0$.

If the exact value $\gamma_\nu \equiv \nabla g(x_\nu)$ of the gradient is known, then $-\gamma_\nu$ gives the direction of steepest descent at the point x_ν. This guarantees that if $\gamma_\nu \neq 0$, then moving along the direction $-\gamma_\nu$ the value of the objective function decreases, that is, $g(x_\nu - \alpha\gamma_\nu) < g(x_\nu)$ for $\alpha > 0$ small enough. The iterative procedure (31) tries to mimic that idea by using the estimates $\hat{\gamma}(x_\nu)$ of the corresponding "true" gradients. The projection Π_χ is needed in order to enforce feasibility of the generated iterates. If the problem is unconstrained, that is, the feasible set χ coincides with the whole space, then this projection is the identity mapping and can be omitted from (31). Note that $\hat{\gamma}(x_\nu)$ does not need to be an accurate estimator of $\nabla g(x_\nu)$.

Kushner and Clark (1978) and Benveniste et al. (1990) contain expositions of the theory of stochastic approximation. Applications of the stochastic approximation method, combined with the infinitesimal perturbation analysis technique for gradient estimation, to the optimization of the steady-state means of single-server queues were studied by Chong and Ramadge (1992) and L'Ecuyer and Glynn (1994).

An attractive feature of the stochastic approximation method is its simplicity and ease of implementation in those cases in which the projection $\Pi_\chi(\cdot)$ can be easily computed. However, it also has severe shortcomings. The crucial question in implementations is the choice of the step sizes α_ν. Small step sizes result in very slow progress towards the optimum while large step sizes make the iterates zigzag. Also, a few wrong steps in the beginning of the procedure may require many iterations to correct. For instance, the algorithm is extremely sensitive to the choice of the constant c in the step size rule $\alpha_\nu = c/\nu$. Therefore, various step size rules were suggested in which the step sizes are chosen adaptively (see Ruppert 1991 for a discussion of that topic).

Another drawback of the stochastic approximation method is that it lacks good stopping criteria and often has difficulties with handling even relatively simple linear constraints.

Another simulation-based approach to stochastic optimization is based on the following idea. Let $\hat{g}_N(x)$ be the sample average function defined in (22), based on a sample of size N. Consider the optimization problem

$$\min_{x \in \chi} \hat{g}_N(x) \tag{32}$$

We can view the above problem as the sample average approximation of the "true" (or expected value) problem (11). The function $\hat{g}_N(x)$ is random in the sense that it depends on the corresponding sample. However, note that once the sample is generated, $\hat{g}_N(x)$ becomes a deterministic function whose values and derivatives can be computed for a given value of the argument x. Consequently, problem (32) becomes a deterministic optimization problem and one can solve it with an appropriate deterministic optimization algorithm.

Let $\hat{v}_N$ and $\hat{x}_N$ denote the optimal objective value and an optimal solution of the sample average problem (32), respectively. By the law of large numbers we have that $\hat{g}_N(x)$ converges to $g(x)$ w.p.1 as $N \to \infty$. It is possible to show that under mild additional conditions, $\hat{v}_N$ and $\hat{x}_N$ converge w.p.1 to the optimal objective value and an optimal solution of the true problem (11), respectively. That is, $\hat{v}_N$ and $\hat{x}_N$ are consistent estimators of their "true" counterparts.

This approach to the numerical solution of stochastic optimization problems is a natural outgrowth of the Monte Carlo method of estimation of the expected value of a random function. The method is known by various names, and it is difficult to point out who was the first to suggest this approach. In the recent literature a variant of this method, based on the likelihood ratio estimator $\bar{g}_N(x)$, was suggested in Rubinstein and Shapiro (1990) under the name *stochastic counterpart method* (also see Rubinstein and Shapiro 1993 for a thorough discussion of such a likelihood ratio–sample approximation approach). In Robinson (1996) such an approach is called the *sample path method*. This idea can also be applied to cases in which the set χ is finite, that is, to stochastic discrete optimization problems (Kleywegt and Shapiro 1999).

Of course, in a practical implementation of such a method, one has to choose a specific algorithm for solving the sample average approximation problem (32). For example, in the unconstrained case, one can use the steepest descent method. That is, iterates are computed by the procedure

$$x_{\nu+1} = x_\nu - \alpha_\nu \nabla \hat{g}_N(x_\nu) \tag{33}$$

where the step size α_ν is obtained by a line search, such as $\alpha_\nu \equiv \arg\min_\alpha \hat{g}_N(x_\nu - \alpha \nabla \hat{g}_N(x_\nu))$. Note that this procedure is different from the stochastic approximation method (31) in two respects. Typically a reasonably large sample size N is used in this procedure, and, more importantly, the step sizes are calculated by a line search instead of being defined a priori. In many interesting cases

$\hat{g}_N(x)$ is a piecewise smooth (and even piecewise linear) function and the feasible set is defined by linear constraints. In such cases bundle-type optimization algorithms are quite efficient (see Hiriart-Urruty and Lemarechal 1993 for a discussion of the bundle method).

A well-developed statistical inference of the estimators $\hat{v}_N$ and $\hat{x}_N$ exists (Rubinstein and Shapiro 1993). That inference aids in the construction of stopping rules, validation analysis, and error bounds for obtained solutions and, furthermore, suggests variance reduction methods that may substantially enhance the rate of convergence of the numerical procedure. For a discussion of this topic and an application to two-stage stochastic programming with recourse, we refer to Shapiro and Homem-de-Mello (1998).

If the function $g(x)$ is twice differentiable, then the above sample path method produces estimators that converge to an optimal solution of the true problem at the same asymptotic rate as the stochastic approximation method, provided that the stochastic approximation method is applied with the *asymptotically optimal* step sizes (Shapiro 1996). On the other hand, if the underlying probability distribution is discrete and $g(x)$ is piecewise linear and convex, then w.p.1 the sample path method provides an exact optimal solution of the true problem for N large enough, and moreover the probability of that event approaches one exponentially fast as $N \to \infty$ (Shapiro and Homem-de-Mello 1999).

4. DYNAMIC PROGRAMMING

Dynamic programming (DP) is an approach for the modeling of dynamic and stochastic decision problems, the analysis of the structural properties of these problems, and the solution of these problems. Dynamic programs are also referred to as Markov decision processes (MDP). Slight distinctions can be made between DP and MDP, such as that in the case of some deterministic problems the term *dynamic programming* is used rather than *Markov decision processes*. The term *stochastic optimal control* is also often used for these types of problems. We shall use these terms synonymously.

Dynamic programs and multistage stochastic programs deal with essentially the same types of problems, namely dynamic and stochastic decision problems. The major distinction between dynamic programming and stochastic programming is in the structures that are used to formulate the models. For example, in DP, the so-called state of the process, as well as the value function, that depends on the state, are two structures that play a central role, but these concepts are usually not used in stochastic programs. Section 4.1 provides an introduction to concepts that are important in dynamic programming.

Much has been written about dynamic programming. Some books in this area are Bellman (1957), Bellman (1961), Bellman and Dreyfus (1962), Nemhauser (1966), Hinderer (1970), Bertsekas and Shreve (1978), Denardo (1982), Ross (1983), Puterman (1994), Bertsekas (1995), and Sennott (1999).

The dynamic programming modeling concepts presented in this chapter are illustrated with an example, which is both a multiperiod extension of the single-period newsvendor problem of Section 2 as well as an example of a dynamic pricing problem.

Example 3 (Revenue management problem). Managers often have to make decisions repeatedly over time regarding how much inventory to obtain for future sales as well as how to determine the selling prices. This may involve inventory of one or more products, and the inventory may be located at one or more locations, such as warehouses and retail stores. The inventory may be obtained from a production operation that is part of the same company as the decision maker, and such a production operation may be a manufacturing operation or a service operation, such as an airline, hotel, or car rental company, or the inventory may be purchased from independent suppliers. The decision maker may also have the option to move inventory between locations, such as from warehouses to retail stores. Often the prices of the products can be varied over time to attempt to find the most favorable balance between the supply of the products and the dynamically evolving demand for the products. Such a decision maker can have several objectives, such as to maximize the expected profit over the long run. The profit involves both revenue, which is affected by the pricing decisions, as well as cost, which is affected by the inventory replenishment decisions.

In Section 4.1 examples are given of the formulation of such a revenue management problem with a single product at a single location as a dynamic program.

4.1. Basic Concepts in Dynamic Programming

In this section the basic concepts used in dynamic programming models are introduced.

4.1.1. Decision Times

A dynamic programming model should distinguish between the decisions made at different points in time. The major reason for this is that the information available to the decision maker is different at different points in time—typically more information is available at later points in time (in fact, many people hold this to be the definition of time).

A second reason why distinguishing decision points is useful is that for many types of DP models it facilitates the computation of solutions. This seems to be the major reason why dynamic programming is used for deterministic decision problems. In this context, the time parameter in the model does not need to correspond to the notion of time in the application. The important feature is that a solution is decomposed into a sequence of distinct decisions. This facilitates computation of the solution if it is easier to compute the individual decisions and then put them together to form a solution than it is to compute a solution in a more direct way.

The following are examples of ways in which the decision points can be determined in a DP model:

- Decisions can be made at predetermined discrete points in time. In the revenue management example, the decision maker may make a decision once per day regarding what prices to set during the day, as well as how much to order on that day.
- Decisions can be made continuously in time. In the revenue management example, the decision maker may change prices continuously in time (which is likely to require a sophisticated way of communicating the continuously changing prices).
- Decisions can be made at random points in time when specific events take place. In the revenue management example, the decision maker may decide on prices at the random points in time when customer requests are received and may decide whether to order and how much to order at the random points in time when the inventory changes.

A well-formulated DP model specifies the way in which the decision points in time are determined.

Most of the results presented in this article are for DP models where decisions are made at predetermined discrete points in time, denoted by $t = 0, 1, \ldots, T$, where T denotes the length of the time horizon. DP models with infinite time horizons are also considered. DP models such as these are often called discrete-time DP models.

4.1.2. States

A fundamental concept in DP is that of a state, denoted by s. The set $\mathbb{S}$ of all possible states is called the state space. The decision problem is often described as a controlled stochastic process that occupies a state $S(t)$ at each point in time t.

Describing the stochastic process for a given decision problem is an exercise in modeling. The modeler has to determine an appropriate choice of state description for the problem. The basic idea is that the state should be a sufficient, and efficient, summary of the available information that affects the future of the stochastic process. For example, for the revenue management problem, choosing the state to be the amount of the product in inventory may be an appropriate choice. If there is a cost involved in changing the price, then the previous price should also form part of the state. Also, if competitors' prices affect the demand for the product, then additional information about competitors' prices and behavior should be included in the state.

Several considerations should be taken into account when choosing the state description, some of which are described in more detail in later sections. A brief overview is as follows. The state should be a sufficient summary of the available information that affects the future of the stochastic process in the following sense. The state at a point in time should not contain information that is not available to the decision maker at that time, because the decision is based on the state at that point in time. (There are also problems, called partially observed Markov decision processes, in which what is also called the state contains information that is not available to the decision maker. These problems are often handled by converting them to Markov decision processes with observable states. This topic is discussed in Bertsekas [1995].) The set of feasible decisions at a point in time should depend only on the state at that point in time, and maybe on the time itself, and not on any additional information. Also, the costs and transition probabilities at a point in time should depend only on the state at that point in time, the decision made at that point in time, and maybe on the time itself, and not on any additional information. Another consideration is that often one would like to choose the number of states to be as small as possible since the computational effort of many algorithms increase with the size of the state space. However, the number of states is not the only factor that affects the computational effort. Sometimes it may be more efficient to choose a state description that leads to a larger state space. In this sense the state should be an efficient summary of the available information.

The state space $\mathbb{S}$ can be a finite, countably infinite, or uncountable set. This article addresses dynamic programs with finite or countably infinite, also called discrete, state spaces $\mathbb{S}$.

4.1.3. Decisions

At each decision point in time, the decision maker has to choose a decision, also called an action or control. At any point in time t, the state s at time t, and the time t, should be sufficient to determine

the set $\mathfrak{A}(s, t)$ of feasible decisions, that is, no additional information is needed to determine the admissible decisions. (Note that the definition of the state of the process should be chosen in such a way that this holds for the decision problem under consideration.) Sometimes the set of feasible decisions depends only on the current state s, in which case the set of feasible decisions is denoted by $\mathfrak{A}(s)$. Although most examples have finite sets $\mathfrak{A}(s, t)$ or $\mathfrak{A}(s)$, these sets may also be countably or uncountably infinite.

In the revenue management example, the decisions involve how much of the product to order, as well as how to set the price. Thus, decision $a = (q, r)$ denotes that quantity q is ordered and that the price is set at r. Suppose the supplier requires that an integer amount between a and b be ordered at a time. Also suppose that the state s denotes the current inventory, and that the inventory may not exceed capacity Q at any time. Then the order quantity may be no more than $Q - s$. Also suppose that the price can be set to be any real number between r_1 and r_2. Then the set of feasible decisions is $\mathfrak{A}(s) = \{a, a + 1, a + 2, \ldots, \min\{Q - s, b\}\} \times [r_1, r_2]$.

The decision maker may randomly select a decision. For example, the decision maker may roll a die and base the decision on the outcome of the die roll. This type of decision is called a randomized decision, as opposed to a nonrandomized, or deterministic, decision. A randomized decision for state s at time t can be represented by a probability distribution on $\mathfrak{A}(s, t)$ or $\mathfrak{A}(s)$. The decision at time t is denoted by $A(t)$.

4.1.4. Transition Probabilities

The dynamic process changes from state to state over time. The transitions between states may be deterministic or random. The presentation here is for a dynamic program with discrete time parameter $t = 0, 1, \ldots$, and with random transitions.

The transitions have a memoryless, or Markovian, property, in the following sense. Given the history $H(t) \equiv (S(0), A(0), S(1), A(1), \ldots, S(t))$ of the process up to time t, as well as the decision $A(t) \in \mathfrak{A}(S(t), t)$ at time t, the probability distribution of the state that the process is in at time $t + 1$ depends only on $S(t)$, $A(t)$, and t, that is, the additional information in the history $H(t)$ of the process up to time t provides no additional information for the probability distribution of the state at time $t + 1$. (Note that the definition of the state of the process should be chosen in such a way that the probability distribution has this memoryless property.)

Such memoryless random transitions can be represented in several ways. One representation is by transition probabilities, which are denoted by $p[s'|s, a, t] \equiv \mathbb{P}[S(t + 1) = s'|H(t), S(t) = s, A(t) = a]$. Another representation is by a transition function f, such that given $H(t)$, $S(t) = s$, and $A(t) = a$, the state at time $t + 1$ is $S(t + 1) = f(s, a, t, \omega)$, where ω is a random variable with a known probability distribution. The two representations are equivalent, and in this article we use mostly transition probabilities. When the transition probabilities do not depend on the time t beside depending on the state s and decision a at time t, they are denoted by $p[s'|s, a]$.

In the revenue management example, suppose the demand has probability mass function $\tilde{p}(r, d) \equiv \mathbb{P}[D = d|\text{price} = r]$ with $d \in \{0, 1, 2, \ldots\}$. Also suppose that a quantity q that is ordered at time t is received before time $t + 1$, and that unsatisfied demand is back-ordered. Then $\mathbb{S} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$, and the transition probabilities are as follows.

$$p[s'|s, (q, r)] = \begin{cases} \tilde{p}(r, s + q - s') & \text{if } s' \leq s + q \\ 0 & \text{if } s' > s + q \end{cases}$$

If a quantity q that is ordered at time t is received after the demand at time t, and unsatisfied demand is lost, then $\mathbb{S} = \{0, 1, 2, \ldots\}$, and the transition probabilities are as follows:

$$p[s'|s, (q, r)] = \begin{cases} \tilde{p}(r, s + q - s') & \text{if } q < s' \leq s + q \\ \sum_{d=s}^{\infty} \tilde{p}(r, d) & \text{if } s' = q \\ 0 & \text{if } s' < q \text{ or } s' > s + q \end{cases}$$

4.1.5. Rewards and Costs

Dynamic decision problems often have as objective to maximize the sum of the rewards obtained in each time period, or equivalently, to minimize the sum of the costs incurred in each time period. Other types of objectives sometimes encountered are to maximize or minimize the product of a sequence of numbers resulting from a sequence of decisions, or to maximize or minimize the maximum or minimum of a sequence of resulting numbers.

In this article we focus mainly on the objective of maximizing the expected sum of the rewards obtained in each time period. At any point in time t, the state s at time t, the decision $a \in \mathfrak{A}(s, t)$ at time t, and the time t, should be sufficient to determine the expected reward $r(s, a, t)$ at time t.

(Again, the definition of the state should be chosen so that this holds for the decision problem under consideration.) When the rewards do not depend on the time t beside depending on the state s and decision a at time t, they are denoted by $r(s, a)$.

Note that even if in the application the reward $\tilde{r}(s, a, t, s')$ at time t depends on the state s' at time $t + 1$, in addition to the state s and decision a at time t, and the time t, the expected reward at time t can still be found as a function of only s, a, and t, because

$$r(s, a, t) = \mathbb{E}[\tilde{r}(s, a, t, s')] = \sum_{s' \in \mathcal{S}} \tilde{r}(s, a, t, s') p[s' | s, a, t]$$

In the revenue management example, suppose unsatisfied demand is back-ordered and that an inventory cost/shortage penalty of $h(s)$ is incurred when the inventory level is s at the beginning of the time period. Then $\tilde{r}(s, (q, r'), s') = r'(s + q - s') - h(s)$ with $s' \leq s + q$. Thus,

$$r(s, (q, r')) = \sum_{d=0}^{\infty} \tilde{p}(r', d) r' d - h(s)$$

If unsatisfied demand is lost, then $\tilde{r}(s, (q, r'), s') = r'(s + q - s') - h(s)$ with $q \leq s' \leq s + q$. Thus,

$$r(s, (q, r')) = \sum_{d=0}^{s-1} \tilde{p}(r', d) r' d + \sum_{d=s}^{\infty} \tilde{p}(r', d) r' s - h(s)$$

In finite horizon problems, there may be a salvage value $v(s)$ if the process terminates in state s at the end of the time horizon t. Such a feature can be incorporated in the previous notation by letting $\mathcal{A}(s, T) = \{0\}$, and $r(s, 0, T) = v(s)$ for all $s \in \mathcal{S}$.

Often the rewards are discounted with a discount factor $\alpha \in [0, 1]$, so that the discounted expected value of the reward at time t is $\alpha^t r(s, a, t)$. Such a feature can again be incorporated in the previous notation by letting $r(s, a, t) = \alpha^t \bar{r}(s, a, t)$ for all s, a, and t, where $\bar{r}$ denotes the undiscounted reward function. When the undiscounted reward does not depend on time, it is convenient to denote explicitly the discounted reward by $\alpha^t r(s, a)$.

4.1.6. Policies

A policy, sometimes called a strategy, prescribes the way a decision is to be made at each point in time, given the information available to the decision maker at the point in time. Therefore, a policy is a solution for a dynamic program.

There are different classes of policies of interest, depending on which of the available information the decisions are based on. A policy can base decisions on all the information in the history of the process up to the time the decision is to be made. Such policies are called history-dependent policies. Given the memoryless nature of the transition probabilities, as well as the fact that the sets of feasible decisions and the expected rewards depend on the history of the process only through the current state, it seems intuitive that it should be sufficient to consider policies that base decisions only on the current state and time, and not on any additional information in the history of the process. Such policies are called memoryless, or Markovian, policies. If the transition probabilities, sets of feasible decisions, and rewards do not depend on the current time, then it also seems intuitive that it should be sufficient to consider policies that base decisions only on the current state, and not on any additional information in the history of the process or on the current time. (However, this intuition may be wrong, as shown by counterexample in Section 4.1.7). Under such policies, decisions are made in the same way each time the process is in the same state. Such policies are called stationary policies.

The decision maker may also choose to use some irrelevant information to make a decision. For example, the decision maker may randomly select a decision by rolling a die or drawing a card from a deck of cards. Policies that allow such randomized decisions are called randomized policies, and policies that do not allow randomized decisions are called nonrandomized or deterministic policies.

Combining the above types of information that policies can base decisions on, the following types of policies are obtained: the class Π^{HR} of history dependent randomized policies, the class Π^{HD} of history dependent deterministic policies, the class Π^{MR} of memoryless randomized policies, the class Π^{MD} of memoryless deterministic policies, the class Π^{SR} of stationary randomized policies, and the class Π^{SD} of stationary deterministic policies. The classes of policies are related as follows: $\Pi^{SD} \subset \Pi^{MD} \subset \Pi^{HD} \subset \Pi^{HR}$, $\Pi^{SD} \subset \Pi^{MD} \subset \Pi^{MR} \subset \Pi^{HR}$, $\Pi^{SD} \subset \Pi^{SR} \subset \Pi^{MR} \subset \Pi^{HR}$.

For the revenue management problem, an example of a stationary deterministic policy is to order quantity $q = s_2 - s$ if the inventory level $s < s_1$, for chosen constants $s_1 \leq s_2$, and to set the price at level $r = \check{r}(s)$ for a chosen function $\check{r}(s)$ of the current state s. An example of a stationary randomized policy is to set the price at level $r = \check{r}_1(s)$ with probability $p_1(s)$ and at level $r = \check{r}_2(s)$

with probability $1 - p_1(s)$ for chosen functions $\check{r}_1(s)$, $\check{r}_2(s)$, and $p_1(s)$ of the current state s. An example of a memoryless deterministic policy is to order quantity $q = s_2(t) - s$ if the inventory level $s < s_1(t)$, for chosen functions $s_1(t) \leq s_2(t)$ of the current time t, and to set the price at level $r = \check{r}(s, t)$ for a chosen function $\check{r}(s, t)$ of the current state s and time t.

4.1.7. Example

In this section an example is presented that illustrates why it is sometimes desirable to consider more general classes of policies, such as memoryless and/or randomized policies, instead of stationary deterministic policies, even if the sets of feasible solutions, transition probabilities, and rewards are stationary. More such examples may be found in Ross (1970), Ross (1983), Puterman (1994), and Sennott (1999).

The examples are for dynamic programs with stationary input data and objective to minimize the long-run average cost per unit time, $\lim \sup_{T\to\infty} \mathbb{E}[\sum_{t=0}^{T-1} r(S(t), A(t))|S(0)]/T$. For any policy π, let

$$V^{\pi}(s) \equiv \limsup_{T\to\infty} \frac{1}{T} \mathbb{E}^{\pi} \left[\sum_{t=0}^{T-1} r(S(t), A(t)) \middle| S(0) = s \right]$$

denote the long-run average cost per unit time under policy π if the process starts in state s, where $\mathbb{E}^{\pi}[\cdot]$ denotes the expected value if policy π is followed.

A policy π^* is called optimal if $V^{\pi^*}(s) = \inf_{\pi\in\Pi^{HR}} V^{\pi}(s)$ for all states s.

Example 4. This example shows that even if the dynamic program has stationary input data, it does not always hold that for any policy $\tilde{\pi}$, and any $\epsilon > 0$, there exists a stationary deterministic policy π that has value function V^{π} within ϵ of the value function $V^{\tilde{\pi}}$ of policy $\tilde{\pi}$.

The state space $\mathcal{S} = \{0, 1, 1', 2, 2', 3, 3', \ldots\}$. Feasible decision sets are $\mathcal{A}(0) = \{a\}$, $\mathcal{A}(i) = \{a, b\}$, and $\mathcal{A}(i') = \{a\}$ for each $i \in \{1, 2, 3, \ldots\}$. When in state $i \in \{0, 1, 2, \ldots\}$, a cost of 2 is incurred, otherwise there is no cost. That is, the costs are $r(0, a) = 2$, $r(i, a) = r(i, b) = 2$, and $r(i', a) = 0$. The transition probabilities are as follows:

$$p[0|0, a] = 1,\ p[i + 1|i, a] = 1,\ p[i'|i, b] = 1 - p[0|i, b] = p_i \qquad \text{for all } i$$

$$p[1|1', a] = 1, \text{ and } p[(i - 1)'|i', a] = 1 \qquad \text{for all } i \geq 2$$

The values p_i can be chosen to satisfy

$$p_i < 1 \text{ for all } i, \text{ and } \prod_{i=1}^{\infty} p_i = \frac{3}{4}$$

Suppose the process starts in state 1. The idea is simple: we would like to go down the chain i', $(i - 1)', \ldots, 1'$ as much as possible. To do that, we also need to go up the chain $1, 2, \ldots, i$, and then go from state i to state i' by making decision b. When we make decision b in state i, there is a risk $1 - p_i > 0$ of making a transition to state 0, which is very bad.

A stationary deterministic policy π that chooses decision a for each state i, has long-run average cost per unit time of $V^{\pi}(1) = 2$, which is as bad as can be. The only other possibility for a stationary deterministic policy π is to choose decision b for the first time in state j. In that case, each time state j is visited, there is a positive probability $1 - p_j > 0$ of making a transition to state 0. It follows that the mean time until a transition to state 0 is made is less than $2j/(1 - p_j) < \infty$, and the long-run average cost per unit time is $V^{\pi}(1) = 2$. Thus, $V^{\pi}(1) = 2$ for all stationary deterministic policies π.

Consider the memoryless deterministic policy $\tilde{\pi}$ that on its jth visit to state 1, chooses decision a, $j - 1$ times, and then chooses decision b. With probability $\Pi_{i=1}^{\infty} p_i = 3/4$, the process never makes a transition to state 0 and the long-run average cost per unit time is 1. Otherwise, with probability $1 - \Pi_{i=1}^{\infty} p_i = 1/4$, the process makes a transition to state 0 and the long-run average cost per unit time is 2. Hence, the expected long-run average cost per unit time is $V^{\tilde{\pi}}(1) = 3/4 \times 1 + 1/4 \times 2 = 5/4$. Thus, there is no ε-optimal stationary deterministic policy for $\varepsilon \in (0, 3/4)$. In fact, by considering memoryless deterministic policies $\tilde{\pi}_k$ that on their jth visit to state 1, choose decision a, $j + k$ times and then choose decision b, one obtains policies with expected long-run average cost per unit time $V^{\tilde{\pi}_k}(1)$ arbitrarily close to 1 for sufficiently large values of k. It is clear that $V^{\pi}(1) \geq 1$ for all policies π, and thus $V^*(1) = 1$, and there is no ε-optimal stationary deterministic policy for $\varepsilon \in (0, 1)$.

4.2. Finite Horizon Dynamic Programs

In this section we investigate dynamic programming models for optimization problems with the form

$$\max_{(A(0),A(1),\ldots,A(T))} \mathbb{E}\left[\sum_{t=0}^{T} r(S(t), A(t), t)\right] \tag{34}$$

where $T < \infty$ is the known finite horizon length and decisions $A(t)$, $t = 0, 1, \ldots, T$, have to be feasible and may depend only on the information available to the decision maker at each time t, that is, the history $H(t)$ of the process up to time t, and possibly some randomization. For the presentation we assume that $\mathcal{S}$ is countable and r is bounded. Similar results hold in more general cases, subject to regularity conditions.

4.2.1. Optimality Results

From the memoryless properties of the feasible sets, transition probabilities, and rewards, it is intuitive that it should be sufficient to consider memoryless deterministic policies. This can be shown to be true for finite horizon problems of the form (34).

The value function V^π of a memoryless policy π is defined by

$$V^\pi(s, t) \equiv \mathbb{E}^\pi\left[\sum_{\tau=t}^{T} r(S(\tau), A(\tau), \tau) \middle| S(t) = s\right] \tag{35}$$

Then, because it is sufficient to consider memoryless deterministic policies, the optimal value function V^* is given by

$$V^*(s, t) \equiv \sup_{\pi \in \Pi^{MD}} V^\pi(s, t) \tag{36}$$

It is easy to see that the value function V^π of a memoryless policy π satisfies the following inductive equation:

$$V^\pi(s, t) = r(s, \pi(s, t), t) + \sum_{s' \in \mathcal{S}} p[s'|s, \pi(s, t), t)]V^\pi(s', t + 1) \tag{37}$$

(Recall that $\pi(s, t)$ denotes the decision under policy π if the process is in state s at time t. If π is a randomized policy, then the understanding is that the expected value is computed with the decision distributed according to probability distribution $\pi(s, t)$. Also, even history-dependent policies satisfy a similar inductive equation, except that the value function depends on the history up to time t.) Similarly, the optimal value function V^* satisfies the following inductive optimality equation:

$$V^*(s, t) = \sup_{a \in \mathcal{A}(s,t)} \left\{r(s, a, t) + \sum_{s' \in \mathcal{S}} p[s'|s, a, t]V^*(s', t + 1)\right\} \tag{38}$$

4.2.2. Finite Horizon Algorithm

Solving a finite horizon dynamic program usually involves using (38) to compute V^* with the following backward induction algorithm. An optimal policy $\pi^* \in \Pi^{MD}$ is then obtained using (40), or an ε-optimal policy $\pi^*_\varepsilon \in \Pi^{MD}$ is obtained using (41).

Finite horizon backward induction algorithm.

0. Set $V^*(s, T + 1) = 0$ for all $s \in \mathcal{S}$.

1. For $t = T, \ldots, 1$, repeat steps 2 and 3.

2. For each $s \in \mathcal{S}$, compute

$$V^*(s, t) = \sup_{a \in \mathcal{A}(s,t)} \left\{r(s, a, t) + \sum_{s' \in \mathcal{S}} p[s'|s, a, t]V^*(s', t + 1)\right\} \tag{39}$$

3. For each $s \in \mathcal{S}$, choose a decision

$$\pi^*(s, t) \in \arg \max_{a \in \mathcal{A}(s,t)} \left\{ r(s, a, t) + \sum_{s' \in \mathcal{S}} p[s'|s, a, t] V^*(s', t + 1) \right\} \tag{40}$$

if the maximum on the right hand side is attained. Otherwise, for any chosen $\varepsilon > 0$, choose a decision $\pi^*_\varepsilon(s, t)$ such that

$$\begin{aligned} r(s, \pi^*_\varepsilon(s, t), t) + \sum_{s' \in \mathcal{S}} p[s'|s, \pi^*_\varepsilon(s, t), t] V^*(s', t + 1) + \frac{\varepsilon}{T + 1} \\ > \sup_{a \in \mathcal{A}(s,t)} \left\{ r(s, a, t) + \sum_{s' \in \mathcal{S}} p[s'|s, a, t] V^*(s', t + 1) \right\} \end{aligned} \tag{41}$$

The value function V^π of a policy π can be calculated with a similar algorithm, except that (37) is used instead of (39), that is, the maximization on the right-hand side of (39) is replaced by the decision under policy π, and step 3 is omitted.

4.2.3. Structural Properties

Dynamic programming is useful not only for the computation of optimal policies and optimal expected values, but also for determining insightful structural characteristics of optimal policies. In fact, for many interesting applications the state space is too big to compute optimal policies and optimal expected values exactly, but dynamic programming can still be used to establish qualitative characteristics of optimal quantities. Some such structural properties are illustrated with examples.

Example 5 (inventory replenishment). A business purchases and sells a particular product. A decision maker has to decide regularly, say once every day, how much of the product to buy. The business does not have to wait to receive the purchased product. In contrast to the newsvendor problem, here product that is not sold on a particular day can be kept in inventory for the future. The business pays a fixed cost K plus a variable cost c per unit of product each time product is purchased. Thus, if a units of product are purchased, then the purchasing cost is $K + ca$ if $a > 0$, and it is 0 if $a = 0$. In addition, if the inventory level at the beginning of the day is s, and a units of product is purchased, then an inventory cost of $h(s + a)$ is incurred, where h is a convex function. The demand for the product on different days are independent and identically distributed. If the demand D is greater than the available inventory $s + a$, then the excess demand is backlogged until additional inventory is obtained, at which time the backlogged demand is filled immediately. Inventory remaining at the end of the time horizon has no value. The objective is to minimize the expected total cost over the time horizon. This problem can be formulated as a discrete-time dynamic program. The state $S(t)$ is the inventory at the beginning of day t. The decision $A(t)$ is the quantity purchased on day t, and the single-stage cost $r(s, a) = (K + ca) I_{\{a>0\}} + h(s + a)$. The transitions are given by $S(t + 1) = S(t) + A(t) - D(t)$. Dynamic programming can be used to show that the following policy is optimal. If the inventory level $S(t) < \sigma^*(t)$, where $\sigma^*(t)$ is called the optimal reorder point at time t, then it is optimal to purchase $\Sigma^*(t) - S(t)$ units of product at time t, where $\Sigma^*(t)$ is called the optimal order-up-to point at time t. If the inventory level $S(t) \geq \sigma^*(t)$, then it is optimal not to purchase any product. Such a policy is often called an (s, S)-policy, or a (σ, Σ)-policy. Similar results hold in the infinite horizon case, except that σ^* and Σ^* do not depend on time t anymore.

Example 6 (resource allocation). A decision maker has an amount of resource that can be allocated over some time horizon. At each discrete point in time, a request for some amount of resource is received. If the request is for more resource than the decision maker has available, then the request has to be rejected. Otherwise, the request can be accepted or rejected. A request must be accepted or rejected as a whole—the decision maker cannot allocate a fraction of the amount of resource requested. Rejected requests cannot be recalled later. If the request is accepted, the amount of resource available to the decision maker is reduced by the amount of resource requested and the decision maker receives an associated reward in return. The amounts of resource and the rewards of future requests are unknown to the decision maker, but the decision maker knows the probability distribution of these. At the end of the time horizon, the decision maker receives a salvage reward for the remaining amount of resource. The objective is to maximize the expected total reward over the time horizon. Problems of this type are encountered in revenue management and the selling of assets such as real estate and vehicles. This resource-allocation problem can be formulated as a dynamic program. The state $S(t)$ is the amount of resource available to the decision maker at the beginning of time period t. The decision $A(t)$ is the rule that will be used for accepting or rejecting requests during time period t. If a request for amount Q of resource with an associated reward R is accepted in time period t, then the single-stage reward is R and the next state is $S(t + 1) = S(t) - Q$. If the request

is rejected, then the next state is $S(t + 1) = S(t)$. It is easy to see that the optimal value function $V^*(s, t)$ is increasing in s and decreasing in t. The following threshold policy, with reward threshold function $x^*(q, s, t) = V^*(s, t + 1) - V^*(s - q, t + 1)$, is optimal. Accept a request for amount Q of resource with an associated reward R if $Q \leq S(t)$ and $R \geq x^*(Q, S(t), t)$, and reject the request otherwise. If each request is for the same amount of resource (say 1 unit of resource), and the salvage reward is concave in the remaining amount of resource, then the optimal value function $V^*(s, t)$ is concave in s and t, and the optimal reward threshold $x^*(1, s, t) = V^*(s, t + 1) - V^*(s - 1, t + 1)$ is decreasing in s and t. These intuitive properties do not hold in general if the requests are for different amounts of resource.

Structural properties of the optimal value functions and optimal policies of dynamic programs have been investigated for many different applications. Some general structural results are given in Serfozo (1976), Topkis (1978), and Heyman and Sobel (1984).

4.3. Infinite Horizon Dynamic Programs

In this section we present dynamic programming models with an infinite time horizon. Although an infinite time horizon is a figment of the imagination, these models often are useful for decision problems with many decision points. Many infinite horizon models also have the desirable feature that there exist stationary deterministic optimal policies. Thus, optimal decisions depend only on the current state of the process and not on the sometimes artificial notion of time, as in finite horizon problems. This characteristic makes optimal policies easier to understand, compute, and implement, which is desirable in applications.

We again assume that $\mathbb{S}$ is countable and r is bounded. Similar results hold in more general cases, subject to regularity conditions. We also assume that the sets $\mathbb{A}(s)$ of feasible decisions depend only on the states s, the transition probabilities $p[s'|s, a]$ depend only on the states s, s', and decisions a, and the rewards $r(s, a)$ depend only on the states s and decisions a, and not on time, as in the finite horizon case.

In this article we focus on dynamic programs with total discounted reward objectives. As illustrated in the example of Section 4.1.7, infinite horizon dynamic programs with other types of objectives, such as long-run average reward objectives, may exhibit undesirable behavior. A proper treatment of dynamic programs with these types of objectives requires more space than we have available here, and therefore we refer the interested reader to the references. Besides, in most practical applications, rewards and costs in the near future are valued more than rewards and costs in the more distant future, and hence total discounted reward objectives are preferred for applications.

4.4. Infinite Horizon Discounted Dynamic Programs

In this section we investigate dynamic programming models for optimization problems with the form

$$\max_{(A(0),A(1),\ldots)} \mathbb{E}\left[\sum_{t=0}^{\infty} \alpha^t r(S(t), A(t))\right] \tag{42}$$

where $\alpha \in (0, 1)$ is a known discount factor. Again, decisions $A(t)$, $t = 0, 1, \ldots$ have to be feasible and may depend only on the information available to the decision maker at each time t, that is, the history $H(t)$ of the process up to time t, and possibly some randomization.

4.4.1. Optimality Results

From the stationary properties of the feasible sets, transition probabilities, and rewards, one would expect that it should be sufficient to consider stationary deterministic policies. This can be shown to be true for infinite horizon discounted problems of the form (42).

The value function V^π of a stationary policy π is defined by

$$V^\pi(s) \equiv \mathbb{E}^\pi\left[\sum_{t=0}^{\infty} \alpha^t r(S(t), A(t)) \middle| S(0) = s\right] \tag{43}$$

Then, because it is sufficient to consider stationary deterministic policies, the optimal value function V^* is given by

$$V^*(s) \equiv \sup_{\pi \in \Pi^{SD}} V^\pi(s) \tag{44}$$

Again motivated by the stationary input data, it is intuitive, and can be shown to be true, that the value function V^π of a stationary policy π satisfies an equation similar to (37) for the finite horizon case, that is,

$$V^{\pi}(s) = r(s, \pi(s)) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, \pi(s)] V^{\pi}(s') \tag{45}$$

Similarly, the optimal value function V^* satisfies the following optimality equation:

$$V^*(s) = \sup_{a \in \mathbb{A}(s)} \left\{ r(s, a) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, a] V^*(s') \right\} \tag{46}$$

4.4.2. Infinite Horizon Algorithms

Solving an infinite horizon discounted dynamic program usually involves computing V^*. An optimal policy $\pi^* \in \Pi^{SD}$ or an ε-optimal policy $\pi^*_\varepsilon \in \Pi^{SD}$ can then be obtained, as shown in this section.

Unlike the finite horizon case, V^* is not computed directly using backward induction. An approach that is often used is to compute a sequence of approximating functions V_i, $i = 0, 1, 2, \ldots$, such that $V_i \to V^*$ as $i \to \infty$.

Approximating value functions provide good policies, as motivated by the following result. Suppose V^* is approximated by $\hat{V}$ such that $\|V^* - \hat{V}\|_\infty \leq \varepsilon$. Consider any policy $\hat{\pi} \in \Pi^{SD}$ such that

$$r(s, \hat{\pi}(s)) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, \hat{\pi}(s)] \hat{V}(s') + \delta \geq \sup_{a \in \mathbb{A}(s)} \left\{ r(s, a) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, a] \hat{V}(s') \right\}$$

for all $s \in \mathbb{S}$, that is, decision $\hat{\pi}(s)$ is within δ of the optimal decision using approximating function $\hat{V}$ on the right-hand side of the optimality equation (46). Then

$$V^{\hat{\pi}}(s) \geq V^*(s) - \frac{2\alpha\varepsilon + \delta}{1 - \alpha} \tag{47}$$

for all $s \in \mathbb{S}$, that is, policy $\hat{\pi}$ has value function within $(2\alpha\varepsilon + \delta)/(1 - \alpha)$ of the optimal value function.

4.4.2.1. Value Iteration One algorithm based on a sequence of approximating functions V_i is called value iteration, or successive approximation. The iterates V_i of value iteration correspond to the value function $V^*(s, T + 1 - i)$ of the finite horizon dynamic program with the same problem parameters. Specifically, starting with initial approximation $V_0(s) = 0 = V^*(s, T + 1)$ for all s, the ith approximating function $V_i(s)$ is the same as the value function $V^*(s, T + 1 - i)$ of the corresponding finite horizon dynamic program, that is, the value function for time $T + 1 - i$ that is obtained after i steps of the backward induction algorithm.

Value iteration algorithm

0. Choose initial approximation V_0 and stopping tolerance ε. Set $i \leftarrow 0$.

1. For each $s \in \mathbb{S}$, compute

$$V_{i+1}(s) = \sup_{a \in \mathbb{A}(s)} \left\{ r(s, a) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, a] V_i(s') \right\} \tag{48}$$

2. If $\|V_{i+1} - V_i\|_\infty < (1 - \alpha)\varepsilon/2\alpha$, then go to step 3. Otherwise, set $i \leftarrow i + 1$ and go to step 1.

3. For each $s \in \mathbb{S}$, choose a decision

$$\pi^*_\varepsilon(s) \in \arg\max_{a \in \mathbb{A}(s)} \left\{ r(s, a) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, a] V_{i+1}(s') \right\}$$

if the maximum on the right-hand side is attained. Otherwise, for any chosen $\delta > 0$, choose a decision $\pi^*_\delta(s)$ such that

$$r(s, \pi^*_\delta(s)) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, \pi^*_\delta(s)] V_{i+1}(s') + (1 - \alpha)\delta > \sup_{a \in \mathbb{A}(s)} \left\{ r(s, a) + \alpha \sum_{s' \in \mathbb{S}} p[s'|s, a] V_{i+1}(s') \right\}$$

It can be shown, using the contraction property provided by the discount factor α, that $V_i \to V^*$ as $i \to \infty$ for any initial approximation V_0. Also, the convergence is geometric with rate α. Specifically,

for any V_0, $\|V_i - V^*\|_\infty \leq \alpha^i \|V_0 - V^*\|_\infty$. That implies that the convergence rate is faster if the discount factor α is smaller.

When the value iteration algorithm stops, the final approximation V_{i+1} satisfies $\|V_{i+1} - V^*\|_\infty < \varepsilon/2$. Furthermore, the chosen policy π_ε^* is an ε-optimal policy, and the chosen policy π_δ^* is an $(\varepsilon + \delta)$-optimal policy.

There are several versions of the value iteration algorithm. One example is Gauss–Seidel value iteration, which uses the most up-to-date approximation $V_{i+1}(s)$ on the right-hand side of (48) as soon as it becomes available, instead of using the previous approximation $V_i(s')$ as shown in (48). Gauss–Seidel value iteration has the same convergence properties and performance guarantees given above, but in practice it usually converges faster.

There are several other algorithms for solving infinite horizon discounted dynamic programs. One of these is policy iteration, which computes a sequence of policies π_i, and their value functions V^{π_i}. Another algorithm is modified policy iteration, which is a generalization of both value iteration and policy iteration. With correct choice of algorithm parameters, modified policy iteration often performs much better than value iteration and policy iteration. There are also several variations on these algorithms, obtained with different choices of algorithm control methods, such as adaptive control, as well as parallel versions. Most books on dynamic programming in the References discuss one or more of these algorithms.

4.5. Approximation Methods

For many interesting applications the state space $\mathcal{S}$ is too big for any of the algorithms discussed so far to be used. This is usually due to the ''curse of dimensionality''—the phenomenon that the number of states grows exponentially in the number of dimensions of the state space. When the state space is too large, not only is the computational effort required by these algorithms excessive, but storing the value function and policy values for each state is impossible with current technology.

Recall that solving a dynamic program usually involves using (38) in the finite horizon case or (46) in the infinite horizon case to compute the optimal value function V^*, and an optimal policy π^*. To accomplish this, the following major computational tasks are performed:

1. Estimation of the optimal value function V^* on the right-hand side of (38) or (46).
2. Estimation of the expected value on the right-hand side of (38) or (46). For many applications, this is a high-dimensional integral that requires a lot of computational effort to compute accurately.
3. The maximization problem on the right hand side of (38) or (46) has to be solved to determine the optimal decision for each state. This maximization problem may be easy or hard, depending on the application. The first part of this article discusses several methods for solving such stochastic optimization problems.

Approximation methods usually involve approaches to perform one or more of these computational tasks efficiently, sometimes by sacrificing optimality.

For many applications, the state space is uncountable and the transition and cost functions are too complex for closed form solutions to be obtained. To compute solutions for such problems, the state space is often discretized. Discretization methods and convergence results are discussed in Wong (1970a), Fox (1973), Bertsekas (1975), Chow and Tsitsiklis (1991), and Kushner and Dupuis (1992).

For many other applications, such as queueing systems, the state space is countably infinite. Computing solutions for such problems usually involves solving smaller dynamic programs with finite state spaces, often obtained by truncating the state space of the original DP, and then using the solutions of the smaller DPs to obtain good solutions for the original DP. Such approaches and their convergence are discussed in Fox (1971), White (1980a, b, 1982), White (1982), Cavazos-Cadena (1986), Van Dijk (1991), and Sennott (1997).

Even if the state space is not infinite, the number of states may be very large. A natural approach is to aggregate states, usually by collecting similar states into subsets, and then to solve a related DP with the aggregated state space. Aggregation and aggregation/disaggregation methods are discussed in Mendelssohn (1982), Chatelin (1984), Schweitzer et al. (1985), Bean et al. (1987), and Bertsekas and Castanon (1989).

Another natural approach for dealing with a large-scale DP is to decompose the DP into smaller related DPs, which are easier to solve, and then to use the solutions of the smaller DPs to obtain a good solution for the original DP. Decomposition methods are presented in Wong (1970b), Collins and Lew (1970), Courtois (1977), and Kleywegt et al. (1999).

Some general state space-reduction methods that include many of the methods mentioned above are analyzed in Whitt (1978, 1979a, b), Hinderer (1976, 1978), Hinderer and Hübner (1977), and Haurie and L'Ecuyer (1986). Surveys are given in Morin (1978) and Rogers et al. (1991).

Another natural and quite different approach for dealing with DPs with large state spaces is to approximate the optimal value function V^* with an approximating function $\hat{V}$. It was shown in Section

4.4.2 that good approximations $\hat{V}$ to the optimal value function V^* lead to good policies $\hat{\pi}$. Polynomial approximations, often using orthogonal polynomials such as Legendre and Chebychev polynomials, have been suggested by Bellman and Dreyfus (1959), Chang (1966), and Schweitzer and Seidman (1985). Approximations using splines have been suggested by Daniel (1976), and approximations using regression splines by Chen et al. (1999). Estimation of the parameters of approximating functions for infinite horizon discounted DPs have been studied in Tsitsiklis and Van Roy (1996), Van Roy and Tsitsiklis (1996), and Bertsekas and Tsitsiklis (1996). Some of this work was motivated by methods proposed for reinforcement learning; see Sutton and Barto (1998) for an overview.

REFERENCES

Albritton, M., Shapiro, A., and Spearman, M. L. (1999), ''Finite Capacity Production Planning with Random Demand and Limited Information,'' Preprint.

Beale, E. M. L. (1955), ''On Minimizing a Convex Function Subject to Linear Inequalities,'' *Journal of the Royal Statistical Society, Series B,* Vol. 17, pp. 173—184.

Bean, J. C., Birge, J. R., and Smith, R. L. (1987), ''Aggregation in Dynamic Programming,'' *Operations Research,* Vol. 35, pp. 215–220.

Bellman, R., and Dreyfus, S. (1959), ''Functional Approximations and Dynamic Programming,'' *Mathematical Tables and Other Aids to Computation,* Vol. 13, pp. 247–251.

Bellman, R. E. (1957), *Dynamic Programming,* Princeton University Press, Princeton, NJ.

Bellman, R. E. (1961), *Adaptive Control Processes: A Guided Tour,* Princeton University Press, Princeton, NJ.

Bellman, R. E., and Dreyfus, S. (1962), *Applied Dynamic Programming,* Princeton University Press, Princeton, NJ.

Benveniste, A., Métivier, M., and Priouret, P. (1990), *Adaptive Algorithms and Stochastic Approximations,* Springer, Berlin.

Bertsekas, D. P. (1975), ''Convergence of Discretization Procedures in Dynamic Programming,'' *IEEE Transactions on Automatic Control,* Vol. AC-20, pp. 415–419.

Bertsekas, D. P. (1995), *Dynamic Programming and Optimal Control,* Athena Scientific, Belmont, MA.

Bertsekas, D. P., and Castanon, D. A. (1989), ''Adaptive Aggregation Methods for Infinite Horizon Dynamic Programming,'' *IEEE Transactions on Automatic Control,* Vol. AC-34, pp. 589–598.

Bertsekas, D. P., and Shreve, S. E. (1978), *Stochastic Optimal Control: The Discrete Time Case,* Academic Press, New York.

Bertsekas, D. P., and Tsitsiklis, J. N. (1996), *Neuro-Dynamic Programming,* Athena Scientific, Belmont, MA.

Birge, J. R., and Louveaux, F. (1997), *Introduction to Stochastic Programming,* Springer Series in Operations Research, Springer, New York.

Cavazos-Cadena, R. (1986), ''Finite-State Approximations for Denumerable State Discounted Markov Deci sion Processes,'' *Applied Mathematics and Optimization,* Vol. 14, pp. 1–26.

Chang, C. S. (1966), ''Discrete-Sample Curve Fitting Using Chebyshev Polynomials and the Approximate Determination of Optimal Trajectories via Dynamic Programming,'' *IEEE Transactions on Automatic Control,* Vol. AC-11, pp. 116–118.

Chatelin, F. (1984), ''Iterative Aggregation/Disaggregation Methods,'' in *Mathematical Computer Performance and Reliability,* G. Iazeolla, P. J. Courtois, and A. Hordijk, Eds., Elsevier, Science Publishers Amsterdam, pp. 199–207.

Chen, V. C. P., Ruppert, D., and Shoemaker, C. A. (1999), ''Applying Experimental Design and Regression Splines to High-Dimensional Continuous-State Stochastic Dynamic Programming,'' *Operations Research,* Vol. 47, pp. 38–53.

Chong, E. K. P., and Ramadge, P. J. (1992), ''Convergence of Recursive Optimization Algorithms Using Infinitesimal Perturbation Analysis Estimates,'' *Discrete Event Dynamic Systems: Theory and Applications,* Vol. 1, pp. 339–372.

Chow, C. S., and Tsitsiklis, J. N. (1991), ''An Optimal One-Way Multigrid Algorithm for Discrete-Time Stochastic Control,'' *IEEE Transactions on Automatic Control,* Vol. AC-36, pp. 898–914.

Collins, D. C., and Lew, A. (1970), ''A Dimensional Approximation in Dynamic Programming by Structural Decomposition,'' *Journal of Mathematical Analysis and Applications,* Vol. 30, pp. 375–384.

Courtois, P. J. (1977), *Decomposability: Queueing and Computer System Applications,* Academic Press, New York.

Daniel, J. W. (1976), "Splines and Efficiency in Dynamic Programming," *Journal of Mathematical Analysis and Applications,* Vol. 54, pp. 402–407.

Dantzig, G. B. (1955), "Linear Programming under Uncertainty," *Management Science,* Vol. 1, pp. 197–206.

Denardo, E. V. (1982), *Dynamic Programming Models and Applications,* Prentice-Hall, Englewood Cliffs, NJ.

Fox, B. L. (1971), "Finite-State Approximations to Denumerable-State Dynamic Programs," *Journal of Mathematical Analysis and Applications,* Vol. 34, pp. 665–670.

Fox, B. L. (1973), "Discretizing Dynamic Programs," *Journal of Optimization Theory and Applications,* Vol. 11, pp. 228–234.

Glasserman, P. (1991), *Gradient Estimation via Perturbation Analysis,* Kluwer, Norwell, MA.

Glynn, P. W. (1990), "Likelihood Ratio Gradient Estimation for Stochastic Systems," *Communications of the ACM,* Vol. 33, pp. 75–84.

Haurie, A., and L'Ecuyer, P. (1986), "Approximation and Bounds in Discrete Event Dynamic Programming," *IEEE Transactions on Automatic Control,* Vol. AC-31, pp. 227–235.

Heyman, D. P., and Sobel, M. J. (1984), *Stochastic Models in Operations Research,* Vol. 2, McGraw-Hill, New York.

Hinderer, K. (1970), *Foundations of Non-stationary Dynamic Programming with Discrete Time Parameter.* Springer, Berlin.

Hinderer, K. (1976), "Estimates for Finite-Stage Dynamic Programs," *Journal of Mathematical Analysis and Applications,* Vol. 55, pp. 207–238.

Hinderer, K. (1978), "On Approximate Solutions of Finite-Stage Dynamic Programs," in *Dynamic Programmming and Its Applications,* M. L. Puterman, Ed., Academic Press, New York, pp. 289–317.

Hinderer, K., and Hübner, G. (1977), "On Exact and Approximate Solutions of Unstructured Finite-Stage Dynamic Programs," in *Markov Decision Theory: Proceedings of the Advanced Seminar on Markov Decision Theory* (Amsterdam, September 13–17, 1976), H. C. Tijms and J. Wessels, Eds., Mathematisch Centrum, Amsterdam. pp. 57–76.

Hiriart-Urruty, J. B., and Lemarechal, C. (1993), *Convex Analysis and Minimization Algorithms,* Springer, Berlin.

Ho, Y. C., and Cao, X. R. (1991), *Perturbation Analysis of Discrete Event Dynamic Systems,* Kluwer, Norwell, MA.

Kall, P., and Wallace, S. W. (1994), *Stochastic Programming,* John Wiley & Sons, Chichester.

Klein Haneveld, W. K., and Van der Vlerk, M. H. (1999), "Stochastic Integer Programming: General Models and Algorithms," *Annals of Operations Research,* Vol. 85, pp. 39–57.

Kleywegt, A. J., and Shapiro, A. (1999), "The Sample Average Approximation Method for Sto chastic Discrete Optimization," Preprint, available at Stochastic Programming E-Print Series, http://dochost.rz.hu-berlin.de/speps/.

Kleywegt, A. J., Nori, V. S., and Savelsbergh, M. W. P. (1999), "The Stochastic Inventory Routing Problem with Direct Deliveries," Technical Report TLI99-01, The Logistics Institute, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.

Kushner, H. J., and Clark, D. S. (1978), *Stochastic Approximation Methods for Constrained and Unconstrained Systems,* Springer, Berlin.

Kushner, H. J., and Dupuis, P. (1992), *Numerical Methods for Stochastic Control Problems in Continuous Time,* Springer, New York.

L'Ecuyer, P., and Glynn, P. W. (1994), "Stochastic Optimization by Simulation: Convergence Proofs for the GI/G/1 Queue in Steady-State," *Management Science,* Vol. 11, pp. 1562–1578.

Mendelssohn, R. (1982), "An Iterative Aggregation Procedure for Markov Decision Processes," *Operations Research,* Vol. 30, pp. 62–73.

Morin, T. (1978), "Computational Advances in Dynamic Programming," in *Dynamic Programmming and its Applications,* M. L. Puterman, Ed., Academic Press, New York. pp. 53–90.

Nemhauser, G. L. (1966), *Introduction to Dynamic Programming,* John Wiley & Sons, New York.

Norkin, V. I., Pflug, G. C., and Ruszczyński, A. (1998), "A Branch and Bound Method for Stochastic Global Optimization," *Mathematical Programming,* Vol. 83, pp. 425–450.

Puterman, M. L. (1994), *Markov Decision Processes,* John Wiley & Sons, New York.

Robbins, H., and Monro, S. (1951), "On a Stochastic Approximation Method," *Annals of Mathematical Statistics,* Vol. 22, pp. 400–407.

Robinson, S. M. (1996), "Analysis of Sample-Path Optimization," *Mathematics of Operations Research,* Vol. 21, pp. 513–528.

Rogers, D. F., Plante, R. D., Wong, R. T., and Evans, J. R. (1991), "Aggregation and Disaggregation Techniques and Methodology in Optimization," *Operations Research,* Vol. 39, pp. 553–582.

Ross, S. M. (1970), *Applied Probability Models with Optimization Applications,* Dover, New York.

Ross, S. M. (1983), *Introduction to Stochastic Dynamic Programming,* Academic Press, New York.

Rubinstein, R. Y. and Shapiro, A. (1990), "Optimization of Static Simulation Models by the Score Function Method," *Mathematics and Computers in Simulation,* Vol. 32, pp. 373–392.

Rubinstein, R. Y., and Shapiro, A. (1993), *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method,* John Wiley & Sons, Chichester.

Ruppert, D. (1991), "Stochastic Approximation," in *Handbook of Sequential Analysis,* B. K. Ghosh and P. K. Sen, Eds., Marcel Dekker, New York, pp. 503–529.

Schultz, R., Stougie, L., and Van der Vlerk, M. H. (1998), "Solving Stochastic Programs with Integer Recourse by Enumeration: A Framework Using Gröbner Basis Reductions," *Mathematical Programming,* Vol. 83, pp. 229–252.

Schweitzer, P. J., and Seidman, A. (1985), "Generalized Polynomial Approximations in Markovian Decision Processes," *Journal of Mathematical Analysis and Applications,* Vol. 110, pp. 568–582.

Schweitzer, P. J., Puterman, M. L., and Kindle, K. W. (1985), "Iterative Aggregation-Disaggregation Procedures for Discounted Semi-Markov Reward Processes," *Operations Research,* Vol. 33, pp. 589–605.

Sennott, L. I. (1997), "The Computation of Average Optimal Policies in Denumerable State Markov Decision Chains," *Advances in Applied Probability,* Vol. 29, pp. 114–137.

Sennott, L. I. (1999), *Stochastic Dynamic Programming and the Control of Queueing Systems,* John Wiley & Sons, New York.

Serfozo, R. F. (1976), Monotone Optimal Policies for Markov Decision Processes. *Mathematical Programming Study,* Vol. 6, pp. 202–215.

Shapiro, A. (1996), "Simulation-Based Optimization: Convergence Analysis and Statistical Inference," *Stochastic Models,* Vol. 12, pp. 425–454.

Shapiro, A., and Homem-de-Mello, T. (1998), "A Simulation-Based Approach to Two-Stage Stochastic Programming with Recourse," *Mathematical Programming,* Vol. 81, pp. 301–325.

Shapiro, A., and Homem-de-Mello, T. (1999), "On the Rate of Convergence of Optimal Solutions of Monte Carlo Approximations of Stochastic Programs," *SIAM Journal on Optimization,* Vol. 11, No. 1, pp. 70–86, 2000.

Sutton, R. S., and Barto, A. G. (1998), *Reinforcement Learning: An Introduction,* MIT Press, Cambridge, MA.

Topkis, D. M. (1978), "Minimizing a Submodular Function on a Lattice," *Operations Research,* Vol. 26, pp. 305–321.

Tsitsiklis, J. N., and Van Roy, B. (1996), "Feature-Based Methods for Large-Scale Dynamic Programming," *Machine Learning,* Vol. 22, pp. 59–94.

Van Dijk, N. (1991), "On Truncations and Perturbations of Markov Decision Problems with an Application to Queueing Network Overflow Control," *Annals of Operations Research,* Vol. 29, pp. 515–536.

Van Roy, B., and Tsitsiklis, J. N. (1996), "Stable Linear Approximations to Dynamic Programming for Stochastic Control Problems with Local Transitions," *Advances in Neural Information Processing Systems 8,* MIT Press, Cambridge, MA, pp. 1045–1051.

Van Slyke, R., and Wets, R. J. B. (1969), "L-Shaped Linear Programs with Application to Optimal Control and Stochastic Programming," *SIAM Journal of Applied Mathematics,* Vol. 17, pp. 638–663.

White, D. J. (1980a), "Finite-State Approximations for Denumerable-State Infinite-Horizon Discounted Markov Decision Processes: The Method of Successive Approximations," in *Recent Developments in Markov Decision Processes,* R. Hatley, L. C. Thomas, and D. J. White, Eds., Academic Press, New York, pp. 57–72.

White, D. J. (1980b), "Finite-State Approximations for Denumerable-State Infinite-Horizon Discounted Markov Decision Processes," *Journal of Mathematical Analysis and Applications,* Vol. 74, pp. 292–295.

White, D. J. (1982), "Finite-State Approximations for Denumerable-State Infinite Horizon Discounted Markov Decision Processes with Unbounded Rewards," *Journal of Mathematical Analysis and Applications,* Vol. 86, pp. 292–306.

Whitt, W. (1978), "Approximations of Dynamic Programs, I," *Mathematics of Operations Research,* Vol. 3, pp. 231–243.

Whitt, W. (1979a), "A-Priori Bounds for Approximations of Markov Programs," *Journal of Mathematical Analysis and Applications,* Vol. 71, pp. 297–302.

Whitt, W. (1979b), "Approximations of Dynamic Programs, II," *Mathematics of Operations Research,* Vol. 4, pp. 179–185.

Wong, P. J. (1970a), "An Approach to Reducing the Computing Time for Dynamic Programming," *Operations Research,* Vol. 18, pp. 181–185.

Wong, P. J. (1970b), "A New Decomposition Procedure for Dynamic Programming," *Operations Research,* Vol. 18, pp. 119–131.