A Tutorial on Stochastic Programming

Alexander Shapiro* and Andy Philpott†

March 21, 2007

1 Introduction

This tutorial is aimed at introducing some basic ideas of stochastic programming. The intended audience of the tutorial is optimization practitioners and researchers who wish to acquaint themselves with the fundamental issues that arise when modeling optimization problems as stochastic programs. The emphasis of the paper is on motivation and intuition rather than technical completeness (although we could not avoid giving some technical details). Since it is not intended to be a historical overview of the subject, relevant references are given in the "Notes" section at the end of the paper, rather than in the text.

Stochastic programming is an approach for modeling optimization problems that involve uncertainty. Whereas deterministic optimization problems are formulated with known parameters, real world problems almost invariably include parameters which are unknown at the time a decision should be made. When the parameters are uncertain, but assumed to lie in some given set of possible values, one might seek a solution that is feasible for all possible parameter choices and optimizes a given objective function. Such an approach might make sense for example when designing a least-weight bridge with steel having a tensile strength that is known only to within some tolerance. Stochastic programming models are similar in style but try to take advantage of the fact that probability distributions governing the data are known or can be estimated. Often these models apply to settings in which decisions are made repeatedly in essentially the same circumstances, and the objective is to come up with a decision that will perform well on average. An example would be designing truck routes for daily milk delivery to customers with random demand. Here probability distributions (e.g., of demand) could be estimated from data that have been collected over time. The goal is to find some policy that is feasible for all (or almost all) the possible parameter realizations and optimizes the expectation of some function of the decisions and the random variables.

*School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332-0205, USA, e-mail: ashapiro@isye.gatech.edu.

†Department of Engineering Science, University of Auckland, Auckland, New Zealand, e-mail: a.philpott@auckland.ac.nz.

Stochastic programming can also be applied in a setting in which a one-off decision must be made. Here an example would be the construction of an investment portfolio to maximize return. Like the milk delivery example, probability distributions of the returns on the financial instruments being considered are assumed to be known, but in the absence of data from future periods, these distributions will have to be elicited from some accompanying model, which in its simplest form might derive solely from the prior beliefs of the decision maker. Another complication in this setting is the choice of objective function: maximizing expected return becomes less justifiable when the decision is to be made once only, and the decision-maker's attitude to risk then becomes important.

The most widely applied and studied stochastic programming models are two-stage (linear) programs. Here the decision maker takes some action in the first stage, after which a random event occurs affecting the outcome of the first-stage decision. A recourse decision can then be made in the second stage that compensates for any bad effects that might have been experienced as a result of the first-stage decision. The optimal policy from such a model is a single first-stage decision and a collection of recourse decisions (a decision rule) defining which second-stage action should be taken in response to each random outcome. As an introductory example we discuss below the classical *inventory model.*

Example 1 (Inventory model) Suppose that a company has to decide an order quantity x of a certain product to satisfy demand d. The cost of ordering is $c > 0$ per unit. If the demand d is bigger than x, then a back order penalty of $b \geq 0$ per unit is incurred. The cost of this is equal to $b(d - x)$ if $d > x$, and is zero otherwise. On the other hand if $d < x$, then a holding cost of $h(x - d) \geq 0$ is incurred. The total cost is then

$$G(x, d) = cx + b[d - x]_+ + h[x - d]_+, \tag{1.1}$$

where $[a]_+$ denotes the maximum of a and 0. We assume that $b > c$, i.e., the back order cost is bigger than the ordering cost. We will treat x and d as continuous (real valued) variables rather than integers. This will simplify the presentation and makes sense in various situations.

The objective is to minimize the total cost $G(x, d)$. Here x is the decision variable and the demand d is a parameter. Therefore, if the demand is known, the corresponding optimization problem can be formulated in the form

$$\min_{x \geq 0} G(x, d). \tag{1.2}$$

The nonnegativity constraint $x \geq 0$ can be removed if a back order policy is allowed. The objective function $G(x, d)$ can be rewritten as

$$G(x, d) = \max\big\{(c - b)x + bd, (c + h)x - hd\big\}, \tag{1.3}$$

which is piecewise linear with a minimum attained at $\bar{x} = d$. That is, if the demand d is known, then (no surprises) the best decision is to order exactly the demand quantity d.

For a numerical instance suppose $c = 1$, $b = 1.5$, and $h = 0.1$. Then

$$G(x,d) = \begin{cases} -0.5x + 1.5d, & \text{if } x < d \\ 1.1x - 0.1d, & \text{if } x \geq d. \end{cases}$$

Let $d = 50$. Then $G(x, 50)$ is the pointwise maximum of the linear functions plotted in Figure 1.

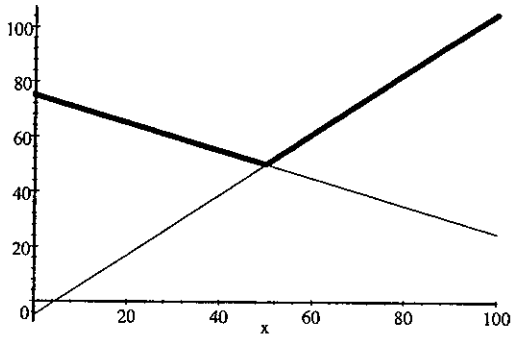

Figure 1: Plot of $G(x, 50)$. Its minimum is at $\bar{x} = 50$

Consider now the case when the ordering decision should be made *before* a realization of the demand becomes known. One possible way to proceed in such situation is to view the demand D as a *random variable* (denoted here by capital D in order to emphasize that it is now viewed as a random variable and to distinguish it from its particular realization d). We assume, further, that the probability distribution of D is *known*. This makes sense in situations where the ordering procedure repeats itself and the distribution of D can be estimated, say, from historical data. Then it makes sense to talk about the expected value, denoted $\mathbb{E}[G(x, D)]$, of the total cost and to write the corresponding optimization problem

$$\min_{x \geq 0} \mathbb{E}[G(x, D)]. \tag{1.4}$$

The above formulation approaches the problem by optimizing (minimizing) the total cost *on average*. What would be a possible justification of such approach? If the process repeats itself then, by the Law of Large Numbers, for a given (fixed) x, the average of the total cost, over many repetitions, will converge with probability one to the expectation $\mathbb{E}[G(x, D)]$. Indeed, in that case a solution of problem (1.4) will be optimal on average.

The above problem gives a simple example of a *recourse action*. At the first stage, before a realization of the demand D is known, one has to make a decision about ordering quantity x. At the second stage after demand D becomes known, it may happen that $d > x$. In that

case the company can meet demand by taking the recourse action of ordering the required quantity $d - x$ at a penalty cost of $b > c$.

The next question is how to solve the optimization problem (1.4). In the present case problem (1.4) can be solved in a closed form. Consider the cumulative distribution function (cdf) $F(z) := \mathsf{Prob}(D \leq z)$ of the random variable D. Note that $F(z) = 0$ for any $z < 0$. This is because the demand cannot be negative. It is possible to show (see the Appendix) that

$$\mathbb{E}[G(x, D)] = b\,\mathbb{E}[D] + (c - b)x + (b + h)\int_0^x F(z)dz. \tag{1.5}$$

Therefore, by taking the derivative, with respect to x, of the right hand side of (1.5) and equating it to zero we obtain that optimal solutions of problem (1.4) are defined by the equation $(b + h)F(x) + c - b = 0$, and hence an optimal solution of problem (1.4) is given by the quantile [1]

$$\bar{x} = F^{-1}(\kappa), \tag{1.6}$$

where $\kappa := \frac{b-c}{b+h}$.

Suppose for the moment that the random variable D has a finitely supported distribution, i.e., it takes values $d_1, ..., d_K$ (called *scenarios*) with respective probabilities $p_1, ..., p_K$. In that case its cdf $F(\cdot)$ is a step function with jumps of size p_k at each d_k, $k = 1, ..., K$. Formula (1.6), for an optimal solution, still holds with the corresponding κ-quantile, coinciding with one of the points d_k, $k = 1, ..., K$. For example, the considered scenarios may represent historical data collected over a period of time. In such case the corresponding cdf is viewed as the empirical cdf giving an approximation (estimation) of the true cdf, and the associated κ-quantile is viewed as the sample estimate of the κ-quantile associated with the true distribution. It is instructive to compare the quantile solution $\bar{x}$ with a solution corresponding to one scenario $d = \bar{d}$, where $\bar{d}$ is, say, the mean (expected value) of D. As it was mentioned earlier, the optimal solution of such (deterministic) problem is $\bar{d}$. The mean $\bar{d}$ can be very different from the κ-quantile $\bar{x}$. It is also worthwhile mentioning that typically sample quantiles are much less sensitive than the sample mean to random perturbations of the empirical data.

In most real settings, closed-form solutions for stochastic programming problems such as (1.4) are rarely available. In the case of finitely many scenarios it is possible to model the stochastic program as a deterministic optimization problem, by writing the expected value $\mathbb{E}[G(x, D)]$ as the weighted sum:

$$\mathbb{E}[G(x, D)] = \sum_{k=1}^{K} p_k G(x, d_k). \tag{1.7}$$

[1] Recall that, for $\kappa \in (0, 1)$, the (left side) κ-quantile of the cumulative distribution function $F(\cdot)$ is defined as $F^{-1}(\kappa) := \inf\{t : F(t) \geq \kappa\}$. In a similar way the right side κ-quantile is defined as $\sup\{t : F(t) \leq \kappa\}$. The set of optimal solutions of problem (1.4) is the (closed) interval with end points given by the respective left and right side κ-quantiles.

The deterministic formulation (1.2) corresponds to *one* scenario d taken with probability one. By using the representation (1.3), we can write problem (1.2) as the linear programming problem

$$\begin{array}{ll} \min\limits_{x,t} & t \\ \text{s.t.} & t \geq (c-b)x + bd, \\ & t \geq (c+h)x - hd, \\ & x \geq 0. \end{array} \tag{1.8}$$

Indeed, for fixed x, the optimal value of (1.8) is equal to $\max\{(c-b)x+bd, (c+h)x-hd\}$, which is equal to $G(x,d)$. Similarly, the expected value problem (1.4), with scenarios $d_1, ..., d_K$, can be written as the linear programming problem:

$$\begin{array}{ll} \min\limits_{x,t_1,...,t_K} & \sum_{k=1}^{K} p_k t_k \\ \text{s.t.} & t_k \geq (c-b)x + bd_k, \ k = 1, ..., K, \\ & t_k \geq (c+h)x - hd_k, \ k = 1, ..., K, \\ & x \geq 0. \end{array} \tag{1.9}$$

The tractability of linear programming problems makes approximation by scenarios an attractive approach for attacking problems like (1.4). In the next section we will investigate the convergence of the solution of such a scenario approximation as K becomes large. It is also worth noting here the "almost separable" structure of problem (1.9). For fixed x, problem (1.9) separates into the sum of optimal values of problems of the form (1.8) with $d = d_k$. Such decomposable structure is typical for two-stage linear stochastic programming problems.

We digress briefly here to compare the exact solution to (1.4) with the scenario solution for the numerical values $c = 1.0$, $b = 1.5$, and $h = 0.1$. Suppose that D has a uniform distribution on the interval $[0, 100]$. Then for any $x \in [0, 100]$,

$$\begin{aligned} \mathbb{E}[G(x,D)] &= b\,\mathbb{E}[D] + (c-b)x + (b+h)\int_0^x F(z)dz \\ &= 75 - 0.5x + 0.008x^2 \end{aligned}$$

as shown in Figure 2.

Suppose the demand is approximated by a finitely supported distribution with two equally likely scenarios $d_1 = 20$, $d_2 = 80$. Then

$$\mathbb{E}[G(x,D)] = \frac{1}{2}\sum_{k=1}^{2} G(x, d_k),$$

which is shown plotted in Figure 3 in comparison with the plot for uniformly distributed D. Figure 3 also shows the same construction with three scenarios $d_1 = 20$, $d_2 = 50$, $d_3 = 80$, with respective probabilities $\frac{2}{5}, \frac{1}{5}, \frac{2}{5}$. This illustrates how more scenarios in general yield a

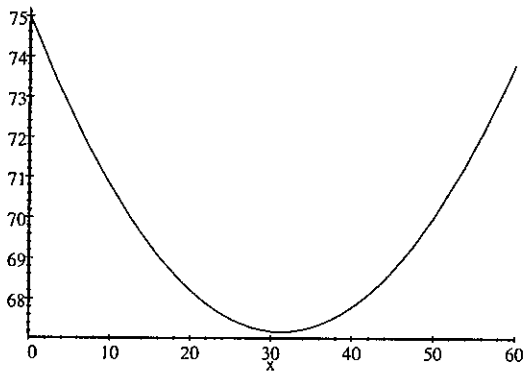

Figure 2: Plot of $\mathbb{E}[G(x, D)]$. Its minimum is at $\bar{x} = 31.25$.

better approximation to the objective function (although in this case this has not made the approximate solution any better). The plot for three scenarios also illustrates the fact that if the scenarios are based on conditional expectations of D over respective intervals $[0, 40]$, $(40, 60]$, and $(60, 100]$ (with corresponding probabilities) and $G(x, D)$ is convex in D, then the approximation gives a lower bounding approximation to $\mathbb{E}[G(x, D)]$ by virtue of Jensen's inequality. □

This example raises several questions. First, how should we approximate the random variable by one with a finitely-supported probability distribution? Second, how should we solve the resulting (approximate) optimization problem? Third, how can we gauge the quality of the approximate solution? In the next section we will discuss these issues in the context of two-stage linear stochastic programming problems. One natural generalization of the two-stage model, which we discuss in section 3, extends it to many stages. Here each stage consists of a decision followed by a set of observations of the uncertain parameters which are gradually revealed over time. In this context stochastic programming is closely related to decision analysis, optimization of discrete event simulations, stochastic control theory, Markov decision processes, and dynamic programming. Another class of stochastic programming models seeks to safeguard the solution obtained against very bad outcomes. Some classical approaches to deal with this are mean-variance optimization, and so-called chance constraints. We briefly discuss these in section 4, and explore the relationship with their modern counterparts, coherent risk measures and robust optimization.

2 Two-stage stochastic programming

In this section we discuss the two-stage stochastic programming approach to optimization under uncertainty. The basic idea of two-stage stochastic programming is that (optimal)

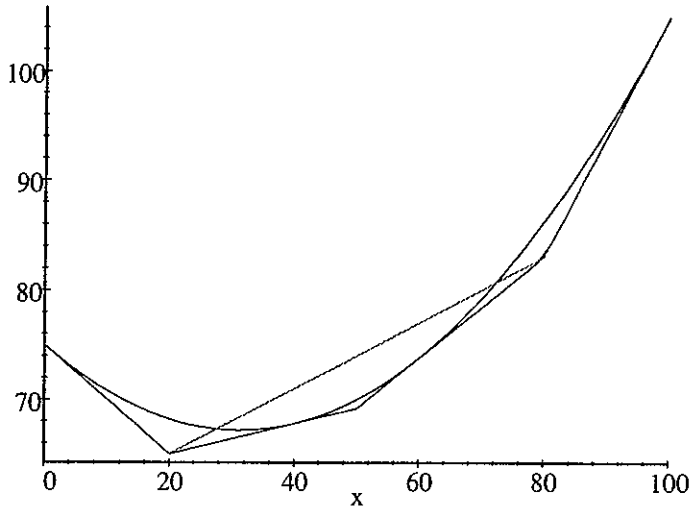

Figure 3: Scenario approximations of $\mathbb{E}[G(x, D)]$.

decisions should be based on data available at the time the decisions are made and should not depend on future observations. The classical two-stage linear stochastic programming problems can be formulated as

$$\min_{x \in X} \{g(x) := c^{\mathsf{T}} x + \mathbb{E}[Q(x, \xi)]\}, \tag{2.1}$$

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

$$\min_{y} q^{\mathsf{T}} y \text{ subject to } Tx + Wy \leq h. \tag{2.2}$$

Here $x \in \mathbb{R}^n$ is the first-stage decision vector, X is a polyhedral set, defined by a finite number of linear constraints, $y \in \mathbb{R}^m$ is the second-stage decision vector and $\xi = (q, T, W, h)$ contains the data of the second-stage problem. In this formulation, at the first stage we have to make a "here-and-now" decision x before the realization of the uncertain data ξ, viewed as a random vector, is known. At the second stage, after a realization of ξ becomes available, we optimize our behavior by solving an appropriate optimization problem.

At the first stage we optimize (minimize) the cost $c^{\mathsf{T}} x$ of the first-stage decision plus the expected cost of the (optimal) second-stage decision. We can view the second-stage problem simply as an optimization problem which describes our supposedly optimal behavior when the uncertain data is revealed, or we can consider its solution as a recourse action where the term Wy compensates for a possible inconsistency of the system $Tx \leq h$ and $q^{\mathsf{T}} y$ is the cost of this recourse action.

The considered two-stage problem is *linear* since the objective functions and the constraints are linear. Conceptually this is not essential and one can consider more general two-stage stochastic programs. For example, if the first-stage problem is integer (combinatorial), its feasible set X could be discrete (finite).

Let us take a closer look at the above two-stage problem. Its formulation involves the assumption that the second-stage data[2] ξ can be modelled as a *random* (not just uncertain) vector with a *known* probability distribution. As mentioned in the inventory example this would be justified in situations where the problem is solved repeatedly under random conditions which do not significantly change over the considered period of time. In such situations one may reliably estimate the required probability distribution and the optimization *on average* could be justified by the Law of Large Numbers.

The other basic question is whether the formulated problem can be solved numerically. In that respect the standard approach is to assume that random vector ξ has a finite number of possible realizations, called *scenarios*, say $\xi_1, ..., \xi_K$, with respective (positive) probabilities $p_1, ..., p_K$. Then the expectation can be written as the summation

$$\mathbb{E}[Q(x,\xi)] = \sum_{k=1}^{K} p_k Q(x,\xi_k), \tag{2.3}$$

and, moreover, the two-stage problem (2.1)–(2.2) can be formulated as one large linear programming problem:

$$\begin{array}{ll} \min\limits_{x,y_1,...,y_K} & c^{\mathsf{T}}x + \sum_{k=1}^{K} q_k^{\mathsf{T}} y_k \\ \text{s.t.} & x \in X,\ T_k x + W_k y_k \leq h_k,\ k = 1, ..., K. \end{array} \tag{2.4}$$

In the above formulation (2.4) we make one copy y_k, of the second stage decision vector, for every scenario $\xi_k = (q_k, T_k, W_k, h_k)$. By solving (2.4) we obtain an optimal solution $\bar{x}$ of the first-stage problem and optimal solutions $\bar{y}_k$ of the second-stage problem for each scenario ξ_k, $k = 1, ..., K$. Given $\bar{x}$, each $\bar{y}_k$ gives an optimal second-stage decision corresponding to a realization $\xi = \xi_k$ of the respective scenario.

Example 2 (Inventory model) Recall the inventory model with K scenarios. This can be written as

$$\begin{array}{ll} \min\limits_{x,t_1,...,t_K} & 0x + \sum_{k=1}^{K} p_k t_k \\ \text{s.t.} & (c-b)x - t_k \leq -bd_k,\ k = 1, ..., K, \\ & (c+h)x - t_k \leq hd_k,\ k = 1, ..., K, \\ & x \geq 0. \end{array} \tag{2.5}$$

The first-stage decision is x, and the second-stage decisions are t_k, $k = 1, ..., K$. □

[2]We use the same notation ξ to denote a random vector and its particular realization. Which one of these two meanings will be used in a particular situation will be clear from the context.

When ξ has an infinite (or very large) number of possible realizations the standard approach is then to represent this distribution by scenarios. As mentioned above this approach raises three questions, namely:

1. how to construct scenarios;

2. how to solve the linear programming problem (2.4);

3. how to measure quality of the obtained solutions with respect to the 'true' optimum.

The answers to these questions are of course not independent: for example, the number of scenarios constructed will affect the tractability of (2.4). We now proceed to address the first and third of these questions in the next subsections. The second question is not given much attention here, but a brief discussion of it is given in the Notes section at the end of the paper.

2.1 Scenario construction

In practice it might be possible to construct scenarios by eliciting experts' opinions on the future. We would like the number of constructed scenarios to be relatively modest so that the obtained linear programming problem can be solved with reasonable computational effort. It is often claimed that a solution that is optimal using only a few scenarios provides more adaptable plans than one that assumes a single scenario only. In some cases such a claim could be verified by a simulation. In theory, we would want some measure of guarantee that an obtained solution solves the original problem with reasonable accuracy. We also would like to point out that typically in applications only the *first* stage optimal solution $\bar{x}$ has a practical value since almost certainly a "true" realization of the random (uncertain) data will be different from the set of constructed (generated) scenarios.

A natural question for a modeler to ask is: *"what is a reasonable number of scenarios to choose so that an obtained solution will be close to the 'true' optimum of the original problem"*. First, it is clear that this question supposes the existence of some good methodology to construct scenarios - a poor procedure may never get close to the optimum, no matter how many are used. If one seeks to construct scenarios, say by using experts, then it quickly becomes clear that, as the number of random parameters increases, some automated procedure to do this will become necessary. For example, suppose that the components of the random vector $\xi \in \mathbb{R}^d$ are independent of each other and we construct scenarios by assigning to each component three possible values, by classifying future realizations as low, medium and high. Then the total number of scenarios is $K = 3^d$. Such *exponential* growth of the number of scenarios makes model development using expert opinion very difficult already for reasonably sized d (say $d = 25$).

Even assuming an automated scenario generation process, with $K = 3^{25} \approx 10^{12}$, say, the corresponding linear program (2.4) becomes too large to be tractable. The situation becomes even worse if we model ξ_i as random variables with continuous distributions. To

make a reasonable discretization of a (one dimensional) random variable ξ_i we would need considerably more than 3 points. At this point the modeler is faced with two somewhat contradictory goals. On one hand, the number of employed scenarios should be computationally manageable. Note that reducing the number of scenarios by a factor of, say, 10 will not help much if the true number of scenarios is $K = 10^{12}$, say. On the other hand, the constructed approximation should provide a reasonably accurate solution of the true problem. A possible approach to reconcile these two contradictory goals is by randomization. That is, scenarios could be generated by Monte-Carlo sampling techniques.

Before we discuss the latter approach, it is important to mention boundedness and feasibility. The function $Q(x, \xi)$ is defined as the optimal value of the second-stage problem (2.2). It may happen that for some feasible $x \in X$ and a scenario ξ_k, the problem (2.2) is unbounded from below, i.e., $Q(x, \xi_k) = -\infty$. This is a somewhat pathological and unrealistic situation meaning that for such feasible x one can improve with a positive probability the second-stage cost indefinitely. One should make sure at the modeling stage that this does not happen.

Another, more serious, problem is if for some $x \in X$ and scenario ξ_k, the system $T_k x + W_k y \leq h_k$ is incompatible, i.e., the second-stage problem is infeasible. In that case the standard practice is to set $Q(x, \xi_k) = +\infty$. That is, we impose an infinite penalty if the second-stage problem is infeasible and such x cannot be an optimal solution of the first-stage problem. This will make sense if such a scenario results in a catastrophic event. It is said that the problem has *relatively complete recourse* if such infeasibility does not happen, i.e., for every $x \in X$ and every scenario ξ_k, the second-stage problem is feasible. It is always possible to make the second-stage problem feasible (i.e., exhibit *complete recourse*) by changing it to

$$\min_{y,t} q^{\mathsf{T}} y + \gamma t \ \text{ subject to } \ Tx + Wy - te \leq h, \ t \geq 0, \tag{2.6}$$

where $\gamma > 0$ is a chosen constant and e is a vector of an appropriate dimension with all its coordinates equal to one. In this formulation the penalty for a possible infeasibility is controlled by the parameter γ.

2.2 Monte Carlo techniques

We now turn our attention to approaches that reduce the scenario set to a manageable size by using Monte Carlo simulation. That is, suppose that the total number of scenarios is very large or even infinite. Suppose, further, that we can generate a sample $\xi^1, ..., \xi^N$ of N replications of the random vector ξ. By this we mean that each ξ^j, $j = 1, ..., N$, has the same probability distribution as ξ. Moreover, if ξ^j are distributed independently of each other, it is said that the sample is iid (independent identically distributed). Given a sample, we can approximate the expectation function $q(x) = \mathbb{E}[Q(x, \xi)]$ by the average

$$\hat{q}_N(x) = N^{-1} \sum_{j=1}^{N} Q(x, \xi^j),$$

and consequently the "true" (expectation) problem (2.1) by the problem

$$\min_{x \in X} \left\{ \hat{g}_N(x) := c^T x + \frac{1}{N} \sum_{j=1}^{N} Q(x, \xi^j) \right\}. \tag{2.7}$$

This rather simple idea was suggested by different authors under different names. In recent literature it has became known as the *sample average approximation* (SAA) method. Note that for a generated sample $\xi^1, ..., \xi^N$, the SAA problem (2.7) is of the same form as the two-stage problem (2.1)–(2.2) with the scenarios ξ^j, $j = 1, ..., N$, each taken with the same probability $p_j = 1/N$. Note also that the SAA method is not an algorithm, one still has to solve the obtained SAA problem by an appropriate numerical procedure.

Example 3 (Inventory model continued) We can illustrate the SAA method on the instance of the inventory example discussed in section 1. Recall that

$$G(x, D) = \begin{cases} -0.5x + 1.5D, & \text{if } x < D \\ 1.1x - 0.1D, & \text{if } x \geq D, \end{cases}$$

where D has a uniform distribution on $[0, 100]$. In Figure 4, we show SAA approximations for three random samples, two with $N = 5$ (ξ^j =15, 60, 72, 78, 82, and ξ^j = 24, 24, 32, 41, 62), and one with $N = 10$ (ξ^j = 8, 10, 21, 47, 62, 63, 75, 78, 79, 83). These samples were randomly generated from the uniform $[0, 100]$ distribution and rounded to integers for the sake of simplicity of presentation. The true cost function is shown in bold. □

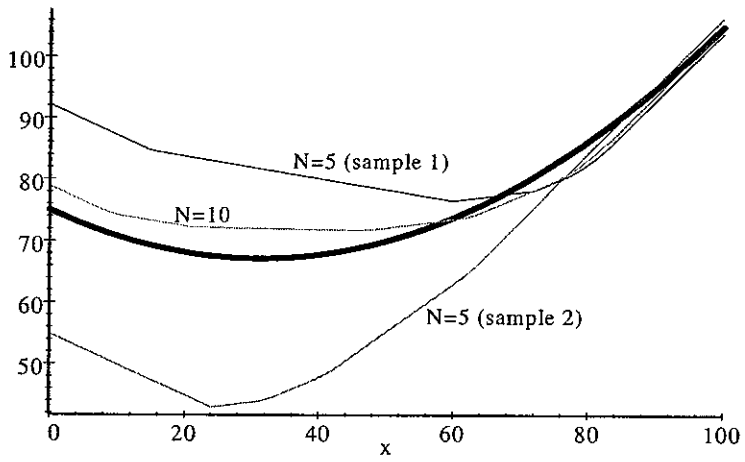

Figure 4: SAA approximations of $\mathbb{E}[G(x, D)]$.

We now can return to the question above: "how large should we choose the sample size N (i.e., how many scenarios should be generated) in order for the SAA problem (2.7) to give a reasonably accurate approximation of the true problem (2.1)". Monte Carlo simulation methods are notorious for slow convergence. For a fixed $x \in X$ and an iid sample, the variance of the sample average $\hat{q}_N(x)$ is equal to $\mathrm{Var}[Q(x,\xi)]/N$. This implies that, in stochastic terms, the sample average converges to the corresponding expectation at a rate of $O_p(N^{-1/2})$. That is, in order to improve the accuracy by one digit, we need to increase the sample size by the factor of 100.

It should be clearly understood that it is not advantageous to use Monte Carlo techniques when the number d of random variables is small. For $d \leq 2$ it would be much better to use a uniform discretization grid for numerical calculations of the required integrals. This is illustrated by the example above. For $d \leq 10$ quasi-Monte Carlo methods usually work better than Monte Carlo techniques[3]. The point is, however, that in principle it is not possible to evaluate the required expectation with a high accuracy by Monte Carlo (or any other) techniques in multivariate cases. It is also possible to show theoretically that the numerical complexity of two-stage linear programming problems grows fast with an increase of the number of random variables and such problems cannot be solved with a high accuracy, like say 10^{-4}, as we expect in deterministic optimization.

The good news about Monte Carlo techniques is that the accuracy of the sample average approximation does not depend on the number of scenarios (which can be infinite), but depends only on the variance of $Q(x,\xi)$. It is remarkable and somewhat surprising that the SAA method turns out to be quite efficient for solving some classes of two-stage stochastic programming problems. It was shown theoretically and verified in numerical experiments that with a manageable sample size, the SAA method (and its variants) solves the "true" problem with a reasonable accuracy (say 1% or 2%) provided the following conditions are met:

(i) it is possible to generate a sample of realizations of the random vector ξ,

(ii) for moderate values of the sample size it is possible to efficiently solve the obtained SAA problem,

(iii) the true problem has relatively complete recourse,

(iv) variability of the second-stage (optimal value) function is not "too large".

[3]Theoretical bounds for the error of numerical integration by quasi-Monte Carlo methods are proportional to $(\log N)^d N^{-1}$, i.e., are of order $O\left((\log N)^d N^{-1}\right)$, with the proportionality constant A_d depending on d. For small d it is "almost" the same as of order $O(N^{-1})$, which of course is better than $O_p(N^{-1/2})$. However, the theoretical constant A_d grows superexponentially with increase of d. Therefore, for larger values of d one often needs a very large sample size N for quasi-Monte Carlo methods to become advantageous. It is beyond the scope of this paper to discuss these issues in detail. The interested reader may be referred to [22], for example.

Usually the requirement (i) does not pose a serious problem in stochastic programming applications, and there are standard techniques and software for generating random samples. Also modern computers coupled with good algorithms can solve linear programming problems with millions of variables (see the Notes section at the end of the tutorial).

Condition (iii), on the other hand, requires a few words of discussion. If for some feasible $x \in X$ the second-stage problem is infeasible with a positive probability (it does not matter how small this positive probability is) then the expected value of $Q(x, \xi)$, and hence its variability, is infinite. If the probability p of such an event (scenario) is very small, say $p = 10^{-6}$, then it is very difficult, or may even be impossible, to verify this by sampling. For example, one would need an iid sample of size N significantly bigger than p^{-1} to reproduce such a scenario in the sample with a reasonable probability.

The sample average approximation problems that arise from the above process are large-scale programming problems. Since our purpose in this tutorial is to introduce stochastic programming models and approaches, we point to the literature for a further reading on that subject in the "Notes" section.

2.3 Evaluating candidate solutions

Given a feasible point $\hat{x} \in X$, possibly obtained by solving a SAA problem, a practical problem is how to evaluate the quality of this point viewed as a candidate for solving the true problem. Since the point $\hat{x}$ is feasible, we clearly have that $g(\hat{x}) \geq v^*$, where $v^* = \min_{x \in X} g(x)$ is the optimal value of the true problem. The quality of $\hat{x}$ can be measured by the optimality gap

$$\mathsf{gap}(\hat{x}) := g(\hat{x}) - v^*.$$

We outline now a statistical procedure for estimating this optimality gap. The true value $g(\hat{x})$ can be estimated by Monte Carlo sampling. That is, an iid random sample ξ^j, $j = 1, ..., N'$, of ξ is generated and $g(\hat{x})$ is estimated by the corresponding sample average $\hat{g}_{N'}(\hat{x}) = c^{\mathsf{T}}\hat{x} + \hat{q}_{N'}(\hat{x})$. At the same time the sample variance

$$\hat{\sigma}^2_{N'}(\hat{x}) := \frac{1}{N'(N'-1)} \sum_{j=1}^{N'} \left[Q(\hat{x}, \xi^j) - \hat{q}_{N'}(\hat{x}) \right]^2, \tag{2.8}$$

of $\hat{q}_{N'}(\hat{x})$, is calculated. Note that it is feasible here to use a relatively large sample size N' since calculation of the required values $Q(\hat{x}, \xi^j)$ involves solving only individual second-stage problems. Then

$$U_{N'}(\hat{x}) := \hat{g}_{N'}(\hat{x}) + z_\alpha \hat{\sigma}_{N'}(\hat{x}) \tag{2.9}$$

gives an approximate $100(1-\alpha)\%$ confidence upper bound for $g(\hat{x})$. This bound is justified by the Central Limit Theorem with the critical value $z_\alpha = \Phi^{-1}(1-\alpha)$, where $\Phi(z)$ is the cdf

of the standard normal distribution. For example, for $\alpha = 5\%$ we have that $z_\alpha \approx 1.64$, and for $\alpha = 1\%$ we have that $z_\alpha \approx 2.33$.

In order to calculate a lower bound for v^* we proceed as follows. Denote by $\hat{v}_N$ the optimal value of the SAA problem based on a sample of size N. Note that $\hat{v}_N$ is a function of the (random) sample and hence is random. To obtain a lower bound for v^* observe that $\mathbb{E}[\hat{g}_N(x)] = g(x)$, i.e., the sample average $\hat{g}_N(x)$ is an unbiased estimator of the expectation $g(x)$. We also have that for any $x \in X$ the inequality $\hat{g}_N(x) \geq \inf_{x' \in X} \hat{g}_N(x')$ holds, so for any $x \in X$, we have

$$g(x) = \mathbb{E}\left[\hat{g}_N(x)\right] \geq \mathbb{E}\left[\inf_{x' \in X} \hat{g}_N(x')\right] = \mathbb{E}\left[\hat{v}_N\right].$$

By taking the minimum over $x \in X$ of the left hand side of the above inequality we obtain that $v^* \geq \mathbb{E}[\hat{v}_N]$.

We can estimate $\mathbb{E}[\hat{v}_N]$ by solving the SAA problems several times and averaging the calculated optimal values. That is, SAA problems based on independently generated samples, each of size N, are solved to optimality M times. Let $\hat{v}_N^1, ..., \hat{v}_N^M$ be the computed optimal values of these SAA problems. Then

$$\bar{v}_{N,M} := \frac{1}{M} \sum_{j=1}^{M} \hat{v}_N^j \tag{2.10}$$

is an unbiased estimator of $\mathbb{E}[\hat{v}_N]$. Since the samples, and hence $\hat{v}_N^1, ..., \hat{v}_N^M$, are independent, we can estimate the variance of $\bar{v}_{N,M}$ by

$$\hat{\sigma}^2_{N,M} := \frac{1}{M(M-1)} \sum_{j=1}^{M} \left(\hat{v}_N^j - \bar{v}_{N,M}\right)^2. \tag{2.11}$$

An approximate $100(1-\alpha)\%$ lower bound for $\mathbb{E}[\hat{v}_N]$ is then given by

$$L_{N,M} := \bar{v}_{N,M} - t_{\alpha,\nu} \hat{\sigma}_{N,M}, \tag{2.12}$$

where $\nu = M - 1$ and $t_{\alpha,\nu}$ is the α-critical value of the t-distribution with ν degrees of freedom. In applications it often suffices to use small values of M, say $M = 5$ or $M = 10$. For a small number of degrees of freedom, the critical value $t_{\alpha,\nu}$ is slightly bigger than the corresponding standard normal critical value z_α, and $t_{\alpha,\nu}$ quickly approaches z_α as ν increases. Since $v^* \geq \mathbb{E}[\hat{v}_N]$, we have that $L_{N,M}$ gives a valid lower statistical bound for v^* as well. Consequently

$$\widehat{\mathsf{gap}}(\hat{x}) := U_{N'}(\hat{x}) - L_{N,M} \tag{2.13}$$

gives a statistically valid (with confidence at least $1 - 2\alpha$) bound on the true $\mathsf{gap}(\hat{x})$. It can be noted that the lower bound $L_{N,M}$ is somewhat conservative since the bias $v^* - \mathbb{E}[\hat{v}_N]$ of the SAA estimator of the true optimal value could be significant for not "too large" values of the sample size N.

Example 4 (Inventory model continued) We can illustrate this procedure on the instance of the inventory example (example 1) discussed in section 1. The following derivations are given for illustration purposes only. As discussed above, in the case of one dimensional random data it is much better to use a uniform discretization grid on the interval $[0, 100]$ rather than a uniformly distributed random sample. Recall

$$G(x,d) = \begin{cases} -0.5x + 1.5d, & \text{if } x < d \\ 1.1x - 0.1d, & \text{if } x \geq d. \end{cases}$$

and for $x \in [0, 100]$

$$\mathbb{E}[G(x,D)] = 75 - 0.5x + 0.008x^2$$

which has a minimum of $v^* = 67.19$ at $\bar{x} = 31.25$. (We round all real numbers to two decimal places in this example.) Suppose we have a candidate solution $\hat{x} = 40$. Then it is easy to compute the true value of this candidate:

$$\mathbb{E}[G(40,D)] = 67.80.$$

To obtain statistical bounds on this value, we sample from D, to give $d_1, d_2, \ldots, d_{N'}$. Consider the following (ordered) random sample of {1, 11, 26, 26, 36, 45, 45, 46, 50, 54, 56, 56, 59, 62, 70, 98} (again this random sample is rounded to integers). With $N' = 16$, this gives an estimated cost of

$$\begin{aligned} \hat{g}_{N'}(\hat{x}) &= \frac{1}{16}\sum_{k=1}^{16} G(40, d^k) \\ &= \frac{1}{16}\left(\sum_{k=1}^{5}\left(44 - 0.1d^k\right) + \sum_{k=6}^{16}\left(-20 + 1.5d^k\right)\right) \\ &= 59.47, \end{aligned}$$

and $\hat{\sigma}^2_{N'}(\hat{x}) = 31.13$. Then an approximate 95% confidence upper bound for $\mathbb{E}[G(40,D)]$ is

$$\begin{aligned} U_{N'}(\hat{x}) &= \hat{g}_{N'}(\hat{x}) + z_{0.05}\hat{\sigma}_{N'}(\hat{x}) \\ &= 59.47 + 1.64\sqrt{31.13} \\ &= 68.65. \end{aligned}$$

Computing a statistical lower bound is a bit more involved. Here we have solved (offline) $M = 9$ SAA problems (each with $N = 5$) to give $\hat{v}_N = 56.39$, 35.26, 69.53, 77.97, 54.87, 42.95, 68.52, 61.99, 78.93. The sample variance is $\hat{\sigma}^2_{N,M} = 24.80$, and $t_{0.05,8} = 1.86$, which gives

$$\begin{aligned} L_{N,M} &= \bar{v}_{N,M} - t_{\alpha,\nu}\hat{\sigma}_{N,M} \\ &= 60.71 - 1.86\sqrt{24.80} \\ &= 51.45 \end{aligned}$$

as an approximately 95% confidence lower bound for $\mathbb{E}[\hat{v}_N]$. It follows that 68.65-51.45=17.20 is an approximate 90% confidence bound on the true value of $\mathsf{gap}(\hat{x})$. To try and reduce this, one might first try to increase N to a value larger than 5. □

3 Multi-stage stochastic programming

The stochastic programming models that we have discussed so far have been static in the sense that we made a (supposedly optimal) decision at one point in time, while accounting for possible recourse actions after all uncertainty has been resolved. There are many situations where one is faced with problems where decisions should be made sequentially at certain periods of time based on information available at each time period. Such multi-stage stochastic programming problems can be viewed as an extension of two-stage programming to a multi-stage setting. In order to demonstrate some basic ideas let us discuss an extension of the inventory model (example 1) to a multi-stage setting.

Example 5 (Inventory model continued) Suppose now that the company has a planning horizon of T periods of time. We view demand D_t as a random process indexed by the time $t = 1, ..., T$. In the beginning, at $t = 1$, there is (known) inventory level y_1. At each period $t = 1, ..., T$ the company first observes the current inventory level y_t and then places an order to replenish the inventory level to x_t. This results in order quantity $x_t - y_t$ which clearly should be nonnegative, i.e., $x_t \geq y_t$. After the inventory is replenished, demand d_t is realized and hence the next inventory level, at the beginning of period $t + 1$, becomes $y_{t+1} = x_t - d_t$. The total cost incurred in period t is

$$c_t(x_t - y_t) + b_t[d_t - x_t]_+ + h_t[x_t - d_t]_+, \tag{3.1}$$

where c_t, b_t, h_t are the ordering cost, backorder penalty cost and holding cost per unit, respectively, at time t. The objective is to minimize the expected value of the total cost over the planning horizon. This can be written as the following optimization problem

$$\begin{array}{ll} \min\limits_{x_t \geq y_t} & \sum\limits_{t=1}^{T} \mathbb{E}\big\{c_t(x_t - y_t) + b_t[D_t - x_t]_+ + h_t[x_t - D_t]_+\big\} \\ \text{s.t.} & y_{t+1} = x_t - D_t,\ t = 1, ..., T-1. \end{array} \tag{3.2}$$

For $T = 1$ the above problem (3.2) essentially is the same as the (static) problem (1.4) (the only difference is the assumption here of the initial inventory level y_1). However, for $T > 1$ the situation is more subtle. It is not even clear what is the exact meaning of the formulation (3.2). There are several equivalent ways to give precise meaning to the above problem. One possible way is to write equations describing dynamics of the corresponding optimization process. That is what we discuss next.

Consider the demand process D_t, $t = 1, ..., T$. We denote by $D_{[t]} = (D_1, ..., D_t)$ the history of the demand process up to time t, and by $d_{[t]} = (d_1, ..., d_t)$ its particular realization. At each period (stage) t, our decision about the inventory level x_t should depend only on

information available at the time of the decision, i.e., on an observed realization $d_{[t-1]}$ of the demand process, and not on future observations. This principle is called the *nonanticipativity* constraint. We assume, however, that the probability distribution of the demand process is known. That is, the conditional probability distribution of D_t, given $D_{[t-1]} = d_{[t-1]}$, is assumed to be known.

At the last stage $t = T$ we need to solve the problem:

$$\min_{x_T} c_T(x_T - y_T) + \mathbb{E}\left\{b_T[D_T - x_T]_+ + h_T[x_T - D_T]_+ \middle| D_{[T-1]} = d_{[T-1]}\right\} \quad \text{s.t. } x_T \geq y_T. \tag{3.3}$$

The expectation in (3.3) is taken conditional on realization $d_{[T-1]}$ of the demand process prior to the considered time T. The optimal value (and the set of optimal solutions) of problem (3.3) depends on y_T and $d_{[T-1]}$, and is denoted $V_T(y_T, d_{[T-1]})$. At stage $t = T-1$ we solve the problem

$$\begin{aligned} \min_{x_{T-1}} \quad & c_{T-1}(x_{T-1} - y_{T-1}) + \mathbb{E}\big\{b_{T-1}[D_{T-1} - x_{T-1}]_+ \\ & +h_{T-1}[x_{T-1} - D_{T-1}]_+ + V_T\left(x_{T-1} - D_{T-1}, D_{[T-1]}\right) \big| D_{[T-2]} = d_{[T-2]}\big\} \\ \text{s.t.} \quad & x_{T-1} \geq y_{T-1}. \end{aligned} \tag{3.4}$$

Its optimal value is denoted $V_{T-1}(y_{T-1}, d_{[T-2]})$. And so on, going backwards in time we write the following *dynamic programming* equations

$$\begin{aligned} V_t(y_t, d_{[t-1]}) = \min_{x_t \geq y_t} \quad & c_t(x_t - y_t) + \mathbb{E}\big\{b_t[D_t - x_t]_+ \\ & +h_t[x_t - D_t]_+ + V_{t+1}\left(x_t - D_t, D_{[t]}\right) \big| D_{[t-1]} = d_{[t-1]}\big\}, \end{aligned} \tag{3.5}$$

$t = T-1, ..., 2$. Finally, at the first stage we need to solve the problem

$$V_1(y_1) = \min_{x_1 \geq y_1} c_1(x_1 - y_1) + \mathbb{E}\big\{b_1[D_1 - x_1]_+ + h_1[x_1 - D_1]_+ + V_2\left(x_1 - D_1, D_1\right)\big\}. \tag{3.6}$$

Let us take a closer look at the above dynamic process. We need to understand how the dynamic programming equations (3.4)–(3.6) could be solved and what is the meaning of an obtained solution. Starting with the last stage $t = T$, we need to calculate the value functions $V_t(y_t, d_{[t-1]})$ going backwards in time. In the present case the value functions cannot be calculated in a closed form and should be approximated numerically. For a generally distributed demand process this could be very difficult or even impossible. The situation is simplified dramatically if we assume that the process D_t is *stagewise independent*, that is, if D_t is independent of $D_{[t-1]}$, $t = 2, ..., T$. Then the conditional expectations in equations (3.3)–(3.5) become the corresponding unconditional expectations, and consequently value functions $V_t(y_t)$ do not depend on demand realizations and become functions of the respective univariate variables y_t only. In that case by using discretizations of y_t and the (one-dimensional) distribution of D_t, these value functions can be calculated with a high accuracy in a recursive way.

Suppose now that somehow we can solve the dynamic programming equations (3.4)–(3.6). Let $\bar{x}_t$ be a corresponding optimal solution, i.e., $\bar{x}_T$ is an optimal solution of (3.3), $\bar{x}_t$ is an optimal solution of the right-hand side of (3.5) for $t = T-1, ..., 2$, and $\bar{x}_1$ is an optimal solution of (3.6). We see that $\bar{x}_t$ is a function of y_t and $d_{[t-1]}$, for $t = 2, ..., T$, while the first-stage (optimal) decision $\bar{x}_1$ is independent of the data. Under the assumption of stagewise independence, $\bar{x}_t = \bar{x}_t(y_t)$ becomes a function of y_t alone. Note that y_t, in itself, is a function of $d_{[t-1]} = (d_1, ..., d_{t-1})$ and decisions $(x_1, ..., x_{t-1})$. Therefore we may think about a sequence of possible decisions $x_t = x_t(d_{[t-1]})$, $t = 1, ..., T$, as functions of realizations of the demand process available at the time of the decision (with the convention that x_1 is independent of the data). Such a sequence of decisions $x_t(d_{[t-1]})$ is called a *policy*. That is, a policy is a rule which specifies our decisions, at every stage based on available information, for any possible realization of the demand process. By its definition any policy $x_t = x_t(d_{[t-1]})$ satisfies the nonanticipativity constraint. A policy is said to be *feasible* if it satisfies the involved constraints with probability one (w.p.1). In the present case a policy is feasible if $x_t \geq y_t$, $t = 1, ..., T$, for almost every realization of the demand process.

We can now formulate the optimization problem (3.2) as the problem of minimizing the expectation in (3.2) with respect to all feasible policies. An optimal solution of such a problem will give us an optimal policy. We have that a policy $\bar{x}_t$ is optimal if and only if it is given by optimal solutions of the respective dynamic programming equations. Note again that in the present case under the assumption of stagewise independence, an optimal policy $\bar{x}_t = \bar{x}_t(y_t)$ is a function of y_t alone. Moreover, in that case it is possible to give the following characterization of the optimal policy. Let x_t^* be an (unconstrained) minimizer of

$$c_t x_t + \mathbb{E}\big\{b_t[D_t - x_t]_+ + h_t[x_t - D_t]_+ + V_{t+1}\left(x_t - D_t\right)\big\}, \quad t = T, ..., 1, \tag{3.7}$$

with the convention that $V_{T+1}(\cdot) = 0$. Then by using convexity of the value functions it is not difficult to show that $\bar{x}_t = \max\{y_t, x_t^*\}$ is an optimal policy. Such policy is called the *basestock* policy. A similar result holds without the assumption of stagewise independence, but then critical values x_t^* depend on realizations of the demand process up to time $t-1$.

As mentioned above, if the stagewise independence condition holds, then each value function $V_t(y_t)$ depends solely on the univariate variable y_t. Therefore in that case we can accurately represent $V_t(\cdot)$ by a discretization, i.e., by specifying its values at a finite number of points on the real line. Consequently, the corresponding dynamic programming equations can be accurately solved recursively going backwards in time. The effectiveness of this approach starts to change dramatically with an increase in the number of variables on which the value functions depend. In the present case this may happen if the demand process is dependent across time. The discretization approach may still work with two, maybe three, or even four, variables. It is out of the question, however, to use such a discretization approach with more than, say, 10 variables. This is the so-called "curse of dimensionality", the term coined by Bellman. Observe that solving the dynamic programming equations recursively under stagewise independence generates a policy that might be more general than we require for the solution to (3.2). That is, the dynamic programming equations will define $\bar{x}_t(y_t)$ for all possible values of y_t, even though some of these values cannot be attained from y_1 by any

combination of demand outcomes $(d_1, ..., d_{t-1})$ and decisions $(x_1, ..., x_{t-1})$.

Stochastic programming tries to approach the problem in a different way by using a discretization of the random process $D_1, ..., D_T$ in the form of a scenario tree. That is, N_1 possible realizations of the (random) demand D_1 are generated. Next, conditional on every realization of D_1, several (say the same number N_2) realizations of D_2 are generated, and so on. In that way $N := N_1 \times ... \times N_T$ different sample paths of the random process $D_1, ..., D_T$ are constructed. Each sample path is viewed as a scenario of a possible realization of the underline random process. There is a large literature of how such scenario trees could (and should) be constructed in a reasonable and meaningful way. One possible approach is to use Monte Carlo techniques to generate scenarios by conditional sampling. This leads to an extension of the SAA method to a multi-stage setting. Note that the total number of scenarios is the product of the numbers N_t of constructed realizations at each stage. This suggests an explosion of the number of scenarios with increase of the number of stages. In order to handle an obtained optimization problem numerically one needs to reduce the number of scenarios to a manageable level. Often this results in taking $N_t = 1$ from a certain stage on. This means that from this stage on the nonanticipativity constraint is relaxed.

In contrast to dynamic programming, stochastic programming does not seek $V_t(y_t)$ for all values of t and y_t, but focuses on computing $V_1(y_1)$ and the first-stage optimal decision $\bar{x}_1$ for the known inventory level y_1. This makes stochastic programming more akin to a decision-tree analysis than dynamic programming (although unlike decision analysis stochastic programming requires the probability distributions to be independent of decisions). A possible advantage of this approach over dynamic programming is that it might mitigate the curse of dimensionality in situations where the states visited as the dynamic process unfolds (or the *dimension* of the state space) are constrained by the currently observed state y_1 (whatever the actions might be). In general, however, the number of possible states that might be visited grows explosively with the number of stages, and so multi-stage stochastic programs are tractable only in special cases. □

4 Risk averse optimization

In this section we discuss the problem of *risk*. We begin by recalling the example of the two-stage inventory problem.

Example 6 (Inventory model continued) Recall that

$$G(x, D) = \begin{cases} (c - b)x + bD, & \text{if } x < D, \\ (c + h)x - hD, & \text{if } x \geq D. \end{cases}$$

We have already observed that for a particular realization of the demand D, the cost $G(\bar{x}, D)$ can be quite different from the optimal-on-average cost $\mathbb{E}[G(\bar{x}, D)]$. Therefore a natural question is whether we can control the risk of the cost $G(x, D)$ to be not "too high". For example, for a chosen value (threshold) $\tau > 0$, we may add to problem (1.4) the constraint

A popular traditional approach to controlling risk in optimization is to try to reduce variability of the (random) cost $G(x,\xi)$, and hence to make it closer to its average (expectation) $\mathbb{E}[G(x,\xi)]$. In classical statistics, variability of a random variable Z is usually measured by its variance or standard deviation. This suggests adding the constraint $\mathrm{Var}[G(x,\xi)] \leq \vartheta$ to problem (1.4), for some chosen constant $\vartheta > 0$. Note that this constraint is equivalent to the corresponding constraint $\sqrt{\mathrm{Var}[G(x,\xi)]} \leq \sqrt{\vartheta}$ for the standard deviation of the total cost.

There are, however, several problems with this approach. Firstly, constraining either the variance or standard deviation of $G(x,\xi)$ means that the obtained constraint set is not guaranteed to be convex unless $G(x,\xi)$ is linear in x. Secondly variance and standard deviation are both symmetrical measures of variability, and, in the minimization case, we are not concerned if the cost is small; we just don't want it to be too large. This motivates the use of asymmetrical measures of variability (like the coherent risk measure $\rho(Z) = \mathbb{E}[Z] + \gamma \mathbb{D}_\kappa[Z]$) which appropriately penalize large values of the cost $G(x,\xi)$. The following example shows that the symmetry of variance and standard deviation is problematic even if the cost function is linear.

Example 9 Suppose that the space $\Xi = \{\xi_1, \xi_2\}$ consists of two points with associated probabilities p and $1-p$, for some $p \in (0,1)$. Consider dispersion measure $\mathbb{D}[Z]$, defined on the space of functions (random variables) $Z : \Xi \to \mathbb{R}$, either of the form

$$\mathbb{D}[Z] := \sqrt{\mathrm{Var}[Z]} \text{ or } \mathbb{D}[Z] := \mathrm{Var}[Z],$$

and the corresponding $\rho(Z) := \mathbb{E}[Z] + \lambda \mathbb{D}[Z]$. Consider functions $Z_1, Z_2 : \Xi \to \mathbb{R}$ defined by $Z_1(\xi_1) = -a$ and $Z_1(\xi_2) = 0$, where a is some positive number, and $Z_2(\xi_1) = Z_2(\xi_2) = 0$. Now, for $\mathbb{D}[Z] = \sqrt{\mathrm{Var}[Z]}$, we have that $\rho(Z_2) = 0$ and $\rho(Z_1) = -pa + \lambda a \sqrt{p(1-p)}$. It follows that for any $\lambda > 0$ and $p < (1+\lambda^{-2})^{-1}$ we have that $\rho(Z_1) > \rho(Z_2)$. Similarly, for $\mathbb{D}[Z] = \mathrm{Var}[Z]$ we have that $\rho(Z_1) > \rho(Z_2)$ if $a > \lambda^{-1}$ and $p < [1 - (\lambda a)^{-1}]^{-1}$. That is, although Z_2 dominates Z_1 in the sense that $Z_2(\xi) \geq Z_1(\xi)$ for *every* possible realization of $\xi \in \Xi$, we have here that $\rho(Z_1) > \rho(Z_2)$, i.e., $\rho(\cdot)$ is *not monotone*.

Consider now the optimization problem

$$\min_{x \in \mathbb{R}^2} \rho[G(x,\xi)] \text{ s.t. } x_1 + x_2 = 1,\ x_1 \geq 0,\ x_2 \geq 0, \tag{4.25}$$

with $G(x,\xi) = x_1 Z_1(\xi) + x_2 Z_2(\xi)$. Let $\bar{x} := (1,0)$ and $x^* := (0,1)$. We have here that $G(x,\xi) = x_1 Z_1(\xi)$, and hence $G(\bar{x},\xi)$ is dominated by $G(x,\xi)$ for *any* feasible point x of problem (4.25) and any realization $\xi \in \Xi$. And yet $\bar{x}$ is not an optimal solution of the corresponding optimization (minimization) problem since $\rho[G(\bar{x},\xi)] = \rho[Z_1]$ is greater than $\rho[G(x^*,\xi)] = \rho(Z_2)$.□

5 Notes

The inventory model, introduced in example 1 (also called the Newsvendor Problem) is classical. This model, and its multistage extensions, are discussed extensively in Zipkin [37], to which the interested reader is referred for a further reading on that topic.

The concept of two-stage linear stochastic programming with recourse was introduced in Beale [4] and Dantzig [8]. Although two-stage linear stochastic programs are often regarded as the classical stochastic programming modeling paradigm, the discipline of stochastic programming has grown and broadened to cover a wide range of models and solution approaches. Applications are widespread, from finance to fisheries management. There is a number of books and monographs where theory and applications of stochastic programming is discussed. In that respect we can mention, for example, monographs [6, 26, 30, 36]. Chance constraints problems were introduced in Charnes, Cooper and Symonds [7]. For a thorough discussion and development of that concept we may refer to Prékopa [26]. An introductory treatment is available in the on-line tutorial by Henrion [12].

Questions of how to construct scenarios and to measure quality of obtained solutions have been studied extensively in the stochastic programming literature. A way of reducing the number of scenarios by certain aggregations techniques are discussed in Heitsch and Römisch [11] and Pflug [25], for example. The approach to statistical validation of a candidate solution, discussed in section 2.3, was suggested in Norkin, Pflug and Ruszczyński [23], and developed in Mak, Morton and Wood [17]. For a more recent work in that direction see Bayraksan and Morton [3].

The first mathematical discussion of risk aversion (using utility functions) is often attributed to Daniel Bernouilli [5]. The concept of utility was formally defined and expounded by von Neumann and Morgenstern [21]. The idea of using a mean-variance approach to stochastic optimization goes back to Markowitz [18]. Coherent risk measures were first introduced by Artzner *et al* [2]. A discussion of drawbacks of using variance or standard deviation as measures of dispersion in stochastic programming can be found in Takriti and Ahmed [33], for example. The approach of using CVaR for approximating chance constraints is due to Rockafellar and Uryasev [27]. We follow Nemirovski and Shapiro [20] to show that the CVaR approximation is the best convex approximation of a chance constraint. It is also suggested in [20] to use the exponential function $\psi(z) = e^z$, instead of the piecewise linear function $[1 + z]_+$, for constructing convex conservative approximations, which has the potential advantage of treating the obtained constraints analytically.

The term "sample average approximation" method was coined in Kleywegt et al [15], although this approach was used long before that paper under various names. Statistical properties of the SAA method are discussed in Shapiro [31] and complexity of two and multi-stage stochastic programming in Shapiro and Nemirovski [32]. From a deterministic point of view, complexity of two-stage linear stochastic programming is discussed in Dyer and Stougie [9]. Rates of convergence of Monte Carlo and Quasi-Monte Carlo estimates of the expected values are discussed in Niederreiter [22]. Numerical experiments with the SAA method are reported in [16, 17, 35], for example.

The sample average approximation problems that arise from the sampling process of two (and multi) stage linear stochastic programs are large-scale linear programming problems. These can be attacked directly using commercial optimization software which is widely available (see e.g. [10]). As the problems grow in size (with the number of scenarios) they become too large to solve using general-purpose codes. Of course for astronomically large numbers of

scenarios we cannot hope to solve any problem exactly, but at the boundary of tractability of general purpose codes we can exploit the special structure inherent in the formulation (2.4) to solve problems in a reasonable amount of time.

The mathematical programming literature on exploiting structure to solve large-scale linear programs is extensive. Since our purpose in this tutorial is to introduce stochastic programming models and approaches to readers, we give below only a brief (and selective) overview of this body of work. A more comprehensive picture is available in the monographs [6, 30, 14].

The formulation (2.4) has a structure that lends itself to the decomposition techniques developed to solve large-scale linear programming problems. These include variants of Benders decomposition (also called the L-shaped method [34]), Lagrangian and augmented Lagrangian decomposition ([28],[19]) interior-point methods [29], and operator splitting [24].

The SAA approach to solving stochastic linear programs may require the solution of a sequence of problems with increasing numbers of scenarios (e.g. if the error in the solution from the current SAA is deemed to be too large). In this case it makes sense to use information obtained in the solution of the problem to guide the solution to a larger SAA problem. This is the approach adopted by so-called *internal* sampling methods, e.g., the *stochastic decomposition* method developed by Higle and Sen [13].

When some or all of the decision variables must take integer values, the formulation (2.4) becomes an integer linear program, which in general lacks the convexity properties to enable decomposition techniques. To overcome this, stochastic integer programs are attacked using a variety of techniques that combine advances in large-scale integer programming with the special structure that arises in applications. A more comprehensive discussion of this is provided in the online tutorial by Ahmed [1].

It should be clearly understood that simple examples given in this paper are for illustration purposes only. It could be dangerous to make general conclusions based on such simple examples. Our intuition based on an analysis of one-dimensional models could be quite misleading in higher dimensional cases.

6 Appendix

6.1 Derivation of (1.5)

Recall

$$G(x, D) = cx + b\max\{D - x, 0\} + h\max\{x - D, 0\}. \tag{6.1}$$

Let $g(x) = \mathbb{E}[G(x, D)]$. It is a convex continuous function. For $x \geq 0$ we have

$$g(x) = g(0) + \int_0^x g'(z)dz,$$

where at nondifferentiable points the derivative $g'(z)$ is understood as the right hand side derivative. Since $D \geq 0$ we have that $g(0) = b\mathbb{E}[D]$. Also observe that

$$\frac{d}{dx}\mathbb{E}[\max\{D - x, 0\}] = \mathsf{Prob}(D \geq x)$$

and

$$\frac{d}{dx}\mathbb{E}[\max\{x - D, 0\}] = \mathsf{Prob}(D \leq x)$$

so

$$\begin{array}{rl} g'(z) & = c + \frac{d}{dz}\mathbb{E}\left[b\max\{D - z, 0\} + h\max\{z - D, 0\}\right] \\ & = c - b\mathsf{Prob}(D \geq z) + h\mathsf{Prob}(D \leq z) \\ & = c - b(1 - F(z)) + hF(z) \\ & = c - b + (b + h)F(z). \end{array}$$

Thus

$$\mathbb{E}[G(x, D)] = b\mathbb{E}[D] + (c - b)x + (b + h)\int_0^x F(z)dz. \tag{6.2}$$

6.2 Derivation of best convex approximation to chance constraints.

Let $\psi : \mathbb{R} \to \mathbb{R}$ be a nonnegative valued, nondecreasing, convex function such that $\psi(z) \geq \mathbb{1}(z)$ for all $z \in \mathbb{R}$, and $\psi(0) = 1$. Since ψ is convex, it is supported, at 0, by a linear function, i.e., there is $a \in \mathbb{R}$ such that $\psi(z) \geq 1 + az$ for all $z \in \mathbb{R}$. Moreover, since $\psi(z)$ is nondecreasing, it follows that $a \geq 0$. If $a = 0$, then $\psi(z) \geq 1$ of course. So suppose that $a > 0$. Then since $\psi(z)$ is nonnegative, we have that $\psi(z) \geq \max\{0, 1 + az\} = [1 + az]_+$ for all z. Note now that the function ψ is defined up to a scaling, i.e., the basic inequality (4.10) is invariant with respect to rescaling $t \mapsto at$. It follows that $\psi(z) := [1 + z]_+$ is a minimal value (and hence a best) choice of such function ψ.

Acknowledgements

The authors would like to thank David Morton, Maarten van der Vlerk, and Bernardo Pagnoncelli for helpful comments on an earlier version of this tutorial.

References

[1] Ahmed, S., Introduction to stochastic integer programming, http://www.stoprog.org.

[2] Artzner, P. Delbaen, F., Eber, J.-M. and Heath, D., Coherent measures of risk, *Mathematical Finance*, **9**, 203–228 (1999).

[3] Bayraksan, G. and Morton, D., Assessing solution quality in stochastic programs, *Mathematical Programming*, **108**, 495–514 (2006).

[4] Beale, E.M.L., On minimizing a convex function subject to linear inequalities, *Journal of the Royal Statistical Society, Series B*, **17**, 173–184 (1955).

[5] Bernouilli, D., Exposition of a new theory on the measurement of risk, 1738. (Translated by L. Sommer in *Econometrica* **22** (1): 22-36, 1954)

[6] Birge, J.R. and Louveaux, F., *Introduction to Stochastic Programming*, Springer, 1997.

[7] Charnes, A., Cooper, W.W. and G.H. Symonds, Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil, *Management Science*, **4**, 235–263 (1958).

[8] Dantzig, G.B., Linear programming under uncertainty, *Management Science*, **1**, 197–206 (1955).

[9] Dyer, M. and Stougie, L., Computational complexity of stochastic programming problems, *Mathematical Programming*, **106**, 423–432 (2006).

[10] Fourer, R., Linear programming frequently asked questions, http://www-unix.mcs.anl.gov/otc/Guide/faq/linear-programming-faq.html (2000).

[11] Heitsch, H. and Römisch,W. Scenario reduction algorithms in stochastic programming, *Computational Optimization and Applications*, **24**, 187-206 (2003).

[12] Henrion, R., Introduction to chance-constrained programming, http://www.stoprog.org.

[13] Higle, J.L. and Sen, S. *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming*, Kluwer Academic Publishers, Dordrecht, 1996.

[14] Kall, P. and Meyer, J., *Stochastic Linear Programming, Models, Theory, and Computation.* Springer, 2005.

[15] Kleywegt, A. J., Shapiro, A. and Homem-de-Mello, T., The sample average approximation method for stochastic discrete optimization, *SIAM J. Optimization*, **12**, 479-502 (2001).

[16] Linderoth, J., Shapiro, A. and Wright, S., The empirical behavior of sampling methods for stochastic programming, *Annals of Operations Research*, **142**, 215-241 (2006).

[17] Mak, W.K., Morton, D.P. and Wood, R.K., Monte Carlo bounding techniques for determining solution quality in stochastic programs, *Operations Research Letters*, **24**, 47–56 (1999).

[18] Markowitz, H.M., Portfolio selection, *Journal of Finance*, **7**, 77–91 (1952).

[19] Mulvey, J.M. and Ruszczyński, A., A new scenario decomposition method for large-scale stochastic optimization, *Operations Research*, **43**, 477–490 (1995).

[20] Nemirovski, A. and Shapiro, A., Convex approximations of chance constrained programs, *SIAM J. Optimization*, **17**, 969-996 (2006).

[21] von Neumann, J. and Morgenstern, O., Theory of Games and Economic Behavior, 1944.

[22] Niederreiter, H., *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia, 1992.

[23] Norkin, V.I., Pflug, G.Ch. and Ruszczyński, A., A branch and bound method for stochastic global optimization. *Mathematical Programming*, **83**, 425–450 (1998).

[24] Pennanen, T. and Kallio, M. A splitting method for stochastic programs, *Annals of Operations Research*, **142**, 259-268 (2006).

[25] Pflug, G.Ch., Scenario tree generation for multiperiod financial optimization by optimal discretization. *Mathematical Programming B*, **89**, 251-271 (2001).

[26] Prékopa, A., *Stochastic Programming*, Kluwer, Dordrecht, Boston, 1995.

[27] Rockafellar, R.T. and Uryasev, S.P., Optimization of conditional value-at-risk, *The Journal of Risk*, **2**, 21-41 (2000).

[28] Rockafellar, R.T. and Wets, R.J-B., Scenarios and policy aggregation in optimization under uncertainty, *Mathematics of Operations Research*, **16**, 119-147 (1991).

[29] Ruszczyński, A., Interior point methods in stochastic programming, IIASA Working paper WP-93-8, (1993).

[30] Ruszczyński, A. and Shapiro, A., (Eds.), *Stochastic Programming*, Handbook in OR & MS, Vol. 10, North-Holland Publishing Company, Amsterdam, 2003.

[31] Shapiro, A., Monte Carlo sampling methods, in: Ruszczyński, A. and Shapiro, A., (Eds.), *Stochastic Programming*, Handbook in OR & MS, Vol. 10, North-Holland Publishing Company, Amsterdam, 2003.

[32] Shapiro, A. and Nemirovski, A., On complexity of stochastic programming problems, in: *Continuous Optimization: Current Trends and Applications*, pp. 111-144, V. Jeyakumar and A.M. Rubinov (Eds.), Springer, 2005.

[33] Takriti, S. and Ahmed, S., On robust optimization of two-stage systems, *Mathematical Programming*, **99**, 109-126 (2004).

[34] Van Slyke, R. and Wets, R.J-B., L-shaped linear programs with applications to optimal control and stochastic programming, *SIAM J. on Appl. Math.*, **17**, 638-663 (1969).

[35] Verweij, B., Ahmed, S., Kleywegt, A.J., Nemhauser, G. and Shapiro, A., The sample average approximation method applied to stochastic routing problems: a computational study, *Computational Optimization and Applications*, **24**, 289–333 (2003).

[36] Ziemba, W.T. and Wallace, S.W., Applications of Stochastic Programming, SIAM, 2006.

[37] Zipkin, P.H. *Foundations of Inventory Management*, McGraw-Hill, 2000.