Ann Oper Res (2006) 142: 215–241
DOI 10.1007/s10479-006-6169-8

The empirical behavior of sampling methods for stochastic programming

Jeff Linderoth · Alexander Shapiro · Stephen Wright

Abstract We investigate the quality of solutions obtained from sample-average approximations to two-stage stochastic linear programs with recourse. We use a recently developed software tool executing on a computational grid to solve many large instances of these problems, allowing us to obtain high-quality solutions and to verify optimality and near-optimality of the computed solutions in various ways.

Keywords Stochastic linear programming · Recourse · Sample average approximations · Computational grid · Monte Carlo sampling · Optimality gap · Statistical KKT test

Research supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, and by the National Science Foundation under Grant 9726385.

Research supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, and by the National Science Foundation under Grant DMS-0073770.

Research supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, and by the National Science Foundation under Grants 9726385 and 0082065.

J. Linderoth
Industrial and Systems Engineering Department, Lehigh University, 200 West Packer Avenue, Bethlehem, PA 18015;
e-mail: jtl3@lehigh.edu

A. Shapiro (✉)
School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta. GA 30332
e-mail: ashapiro@isye.gatech.edu

S. J. Wright
Computer Sciences Department, University of Wisconsin-Madison, 1210 West Dayton Street. Madison, WI 53706
e-mail: swright@cs.wisc.edu

1. Introduction

Consider the following stochastic programming problem

$$\min_{x \in X} \{f(x) := \mathbb{E}_P[F(x, \xi(\omega))]\}, \tag{1}$$

where $F(x, \xi)$ is a real valued function of two vector variables $x \in \mathbb{R}^n$ and $\xi \in \mathbb{R}^d$, X is a subset of $\mathbb{R}^n$, and $\xi(\omega)$ is viewed as a random vector having probability distribution P, which is assumed to be known. Often we omit the subscript P and simply write $\mathbb{E}[F(x, \xi(\omega))]$. In this paper, we discuss sampling techniques for obtaining and verifying approximate solutions of (1), and we present results for several standard problems from the literature. Some of the ideas presented here are applicable to the general form (1), while others apply specifically to two-stage stochastic linear programs with recourse.

The two-stage stochastic linear program with recourse can be written in standard form as follows:

$$\min_x c^T x + \mathcal{Q}(x) \text{ subject to } Ax = b,\ x \geq 0, \tag{2}$$

where $\mathcal{Q}(x) := \mathbb{E}[Q(x, \xi(\omega))]$ and $Q(x, \xi)$ is the optimal value of the second-stage problem

$$\min_y q^T y \text{ subject to } Tx + Wy = h,\ y \geq 0. \tag{3}$$

Here some or all elements of the data vector $\xi = (q, h, T, W)$ can be random (that is, can depend on ω). In problems with *fixed recourse*, the matrix W is fixed (deterministic). If we define

$$F(x, \xi) := c^T x + Q(x, \xi), \quad X = \{x : Ax = b,\ x \geq 0\},$$

and assume that the optimal value of the second-stage problem (3) is finite for all $x \in X$ and almost every ξ, then problem (2) becomes a particular case of problem (1).

We report computational results in this paper for problems of the form (2), (3) with fixed recourse. We consider problems where the probability distribution of $\xi(\omega)$ has *finite support*; that is, $\xi(\omega)$ has a finite number of possible realizations, called *scenarios* $\{\xi_1, \xi_2, \ldots \xi_K\}$ with respective probabilities $p_k \in (0, 1)$, $k = 1, 2, \ldots, K$. For such problems, the expected value function $\mathcal{Q}(x)$ can be written as the finite sum

$$\mathcal{Q}(x) = \sum_{k=1}^{K} p_k Q(x, \xi_k), \tag{4}$$

where $Q(x, \xi)$ is once again the optimal value of (3) and ξ_k denotes the data vector (q_k, h_k, T_k, W), $k = 1, 2, \ldots, K$. By defining y_k to be the instance of the second-stage variable vector corresponding to scenario k, we can combine (2), (3) into a single linear program as follows:

$$\min_{x,y_1,y_2,\dots,y_K} c^T x + \sum_{k=1}^{K} p_k q_k^T y_k \text{ subject to} \tag{5a}$$

$$Ax = b,\ x \geq 0, \tag{5b}$$

$$W y_k = h_k - T_k x,\ y_k \geq 0,\ k = 1, 2, \dots, K. \tag{5c}$$

In many applications, K is an astronomically large number. This situation can arise when each component of the random data vector ξ has a number of possible values, independent of the other components. The total number of scenarios is then the product of the number of possible values for each component of ξ. For example, if ξ has 100 components, each of which has two possible values, then the total number of scenarios is 2^{100}, or approximately 10^{30}. In such situations, it is not practical to solve (2), (3), or equivalently (5), directly. It is not even possible to evaluate the function $\mathcal{Q}(x)$, for a given x, by using formula (4) in a reasonable amount of time, since the number of second-stage linear programs (3) to be solved is just too large.

We can however use sampling techniques, which consider only randomly selected subsets of the set $\{\xi_1, \xi_2, \dots, \xi_K\}$ to obtain approximate solutions. Sampling techniques can be applied in different ways. One approach uses sampling in an "interior" fashion. Such algorithms aim to solve the original problem (1), but resort to sampling whenever the algorithm requires an (approximate) value of $f(\cdot)$ or subgradient information for $f(\cdot)$ at some point x. Typically, a different sample is used each time function or subgradient information is required. We mention in particular the L-shaped method with embedded sampling of Dantzig and Infanger (1991), the stochastic decomposition method of Higle and Sen (1991, 1996), and stochastic quasi-gradient methods (see Ermoliev, 1988). A second fundamental approach to sampling is an "exterior" approach, in which a sample is selected from $\{\xi_1, \xi_2, \dots, \xi_K\}$ and a corresponding approximation to $f(\cdot)$ is defined from this sample. This approximate objective, known as a *sample-average approximation* of $f(\cdot)$, is then minimized using a deterministic optimization algorithm; no further sampling is performed. This approach is known variously as sample-path optimization (Gürkan, Özge, and Robinson, 1994; Plambeck et al., 1996) or the stochastic counterpart method (Rubinstein and Shapiro, 1990, 1993).

In scenario-based algorithms, each scenario can often be considered independently from other scenarios. This makes such algorithms well suited to implementation on parallel computing platforms, where different processors execute the independent portions of the computation in parallel.

Our computational experiments are performed on an unusual platform—a large federation of nondedicated, heterogeneous, loosely coupled processors, known as a *computational grid* or a *metacomputer* (Foster and Kesselman, 1999; Livny et al., 1997). The grid represents an extremely powerful computing resource that can provide large amounts of computing power over extended time periods, yet is also inexpensive, especially since we use only idle time on many of the workstations and PCs that make up the grid. We needed the computational power of the grid to determine optimal and near-optimal solutions to two-stage stochastic linear programs with recourse, and to test the convergence behavior of sample average approximations of $f(\cdot)$. Zakeri (2000) has also used a computational grid to verify the quality of a given solution to a stochastic programming instance. Other recent use of computational grids to solve large, complex optimization problems have been described by Anstreicher et al. (2002), Goux and Leyffer (2002), and Chen, Ferris, and Linderoth (2001).

The remainder of the paper is structured as follows. Section 2 defines the construction of sample-average approximations using different sampling techniques. In Section 3, we summarize results from Shapiro and Homem-de-Mello (1998) concerning convergence of the minimizer of the sample-average approximation to the exact minimizer of the original problem under certain circumstances. In Section 4, we describe techniques for estimating lower and upper bounds on the optimal value of the problem (1), and for estimating the gap between these bounds. We describe several tests for optimality or near-optimality of a candidate solution in Section 5. In Section 6 we focus on the two-stage stochastic linear programming problem with recourse (2), outlining the software that we used to solve this problem. (A fuller description can be found in Linderoth and Wright (2003).) Section 7 briefly highlights the characteristics of the computational grid we used in this work. Section 8 describes the benchmark problems used in our computational tests, while Section 9 describes and analyzes the results obtained from our tests.

2. Sample-average approximations

Suppose that a random sample $\xi^1, \xi^2, \ldots \xi^N \sim P$ of N (not necessarily independent) realizations of the random vector $\xi(\omega)$ is generated. By replacing $f(\cdot)$ by an approximation based on this sample, we approximate the problem (1) by

$$\min_{x \in X} \left\{ \hat{f}_N(x) := N^{-1} \sum_{i=1}^{N} F(x, \xi^i) \right\}. \tag{6}$$

The function $\hat{f}_N$ is a sample-average approximation (SAA) to the objective f of (1). We denote by $\hat{v}_N$ the optimal value and by $\hat{x}_N$ an optimal solution of (6).

Since the random realizations ξ^i have the same probability distribution P, it follows that $\hat{f}_N(x)$ is an unbiased estimator of $f(x)$, for any x. If we perform Monte Carlo sampling, in which all ξ^i are independent of each other, then Var $[\hat{f}_N(x)] = \sigma^2(x)/N$, where $\sigma^2(x) :=$ Var$[F(x, \xi(\omega))]$.

We also used Latin Hypercube sampling (LHS) to construct the sample average function $\hat{f}_N$. In this technique, initially proposed by McKay, Beckman, and Conover (1979), a sample of size N is constructed by dividing the interval (0, 1) into N subintervals of equal size and picking one number randomly from each subinterval. These N numbers are then shuffled, and the resulting sequence is used to generate random variates for a given distribution, possibly by performing an inverse transform. Assuming that the components of the random vector are independently distributed, the procedure is repeated for each component, yielding a stratified sample of size N for that vector. The sample-average function $\hat{f}_N$ so obtained is an unbiased estimator of f, but its variance Var $[\hat{f}_N(x)]$ often is considerably less than that obtained from Monte Carlo sampling. Further discussion of LHS and its properties can be found in Avramidis and Wilson (1996, Section 1.2.2), Stein (1987), and Owen (1997). In stochastic programming, LHS methodology was used in Diwekar and Kalagnanam (1997) and Bailey, Jensen, and Morton (1999). Some other variance reduction techniques have been described in Higle (1998). Importance sampling variance reduction has been described by Dantzig and Infanger (1991).

From a computational point of view, one can regard the SAA problem (6) as a stochastic programming problem with a finite number of scenarios $\xi^1, \xi^2, \ldots, \xi^N$ each with equal probability $p_i = N^{-1}$. Therefore, any numerical algorithm suitable for solving the corresponding stochastic problem with a discrete distribution can be applied to the SAA problem.

3. Convergence of the SAA solution to the exact solution set

A fundamental question central to any sampling-based solution approach is how large must the sample size be so that the solution to the sampled instance is a good (or optimal) solution for the true instance. In this section, we review a theory, developed recently in Shapiro and Homem-de-Mello (1998), Kleywegt, Shapiro, and Homem-de-Mello (2001), Shapiro, Homem-de-Mello, and Kim (2002), concerning the rate of convergence of the solutions $\hat{x}_N$ of the SAA problem (6) to the optimal solution set of the original problem (1). This theory suggests that for certain instances, the sample size necessary to obtain excellent solutions is small compared to the size of the whole sample space.

Unless stated otherwise, we make the following assumptions on the problem (1) throughout the remainder of the paper:

(A1) The distribution of $\xi(\omega)$ has finite support, say $\Xi = \{\xi_1, \xi_2, \ldots, \xi_K\}$;
(A2) For every $\xi \in \Xi$, the function $F(\cdot, \xi) : X \to I\!R$ is convex and piecewise linear;
(A3) The set X is polyhedral:
(A4) The set S of optimal solutions of (1) is nonempty and bounded.

Assumption (A1) means that we deal with random data having a finite, although possibly very large, number of realizations (scenarios). In the case of two-stage linear programs of the form (2), (3) we have that $F(x, \xi) < +\infty$ if the corresponding second stage problem is feasible. We also have that $F(\cdot, \xi)$ is convex and piecewise linear, and hence $F(\cdot, \xi)$ is bounded from below if $F(x, \xi) > -\infty$ for at least one x. Therefore, assumption (A2) is satisfied by stochastic two-stage linear programs provided that the recourse is relatively complete, that is, for every $x \in X$ and $\xi \in \Xi$, the second stage problem (3) is feasible. For the problems used in our experiments, the set X is defined by linear constraints, and hence is polyhedral, and consequently assumption (A3) holds. For the problems considered, the final assumption (A4) is satisfied as well. Assumptions (A1) and (A2) imply that the expected value function $f(\cdot)$ is finite valued, piecewise linear and convex on X. These assumptions together with (A3) imply that the set S is convex and polyhedral.

For $\varepsilon \geq 0$ we denote by S_ε the set of all ε-optimal solutions of (1), that is, the set of points $\bar{x} \in X$ such that $f(\bar{x}) \leq v^* + \varepsilon$, where v^* is the optimal objective value in (1). In particular, for $\varepsilon = 0$ the set S_ε coincides with the set S of optimal solutions of (1).

Under assumptions (A1)–(A4), we have that for each $\varepsilon \geq 0$ there exist constants $C_\varepsilon > 0$ and $\beta_\varepsilon > 0$ such that the inequality

$$1 - P(\hat{x}_N \in S_\varepsilon) \leq C_\varepsilon e^{-\beta_\varepsilon N} \tag{7}$$

holds for all $N \in \mathbb{N}$. Note that the choice of the optimal solution $\hat{x}_N$ of (6) here is arbitrary, so that the probability $P(\hat{x}_N \in S_\varepsilon)$ actually refers to the probability that the set of optimal solutions of the SAA problem (6) is included in the set S_ε. It follows from (7) that

$$\limsup_{N \to \infty} \frac{1}{N} \log[1 - P(\hat{x}_N \in S_\varepsilon)] \leq -\beta_\varepsilon. \tag{8}$$

In other words, the probability that $\hat{x}_N$ is an ε-optimal solution of the true (expected value) problem (1) approaches 1 exponentially fast in N. The above result (7), and hence (8), is a consequence of the upper bound of Cramér's Large Deviation Theorem and the assumed polyhedral structure of the problem (cf., Shapiro and Homem-de-Mello 2000, Theorem 3.2 and Kleywegt, Shapiro, and Homem-de-Mello 2001, Section 2.2). By using the lower bound of Cramér's Large Deviation Theorem, it is possible to show that a limit in the left hand side of (8) is attained, so that "lim sup" can be replaced by "lim".

The estimates (7) and (8) hold in particular for $\varepsilon = 0$, that is, for $S_\varepsilon = S$. Of course, the probability $P(\hat{x}_N \in S_\varepsilon)$ increases with ε, and hence for larger values of ε the exponential constant β_ε is larger. In the case in which the solution to (1) is a singleton (that is, $S = \{x_0\}$ for some $x_0 \in X$), there exists $\beta > 0$ such that

$$\lim_{N\to\infty} \frac{1}{N} \log[1 - P(\hat{x}_N = x_0)] = -\beta. \tag{9}$$

We can refine the expression (9) by quantifying the conditioning of the singleton solution. Denote by $T_X(x_0)$ the tangent cone to the feasible set X at x_0 and by $F'_\omega(x_0, d)$ and $f'(x_0, d)$ the directional derivatives of $F(\cdot, \omega)$ and $f(\cdot)$, respectively, at x in the direction d. Because of the assumed polyhedral structure of problem (1) and since the optimal solution x_0 is unique, we have by the theory of linear programming that it is *sharp*, that is

$$f'(x_0, d) > 0, \quad \forall d \in T_X(x_0)\backslash\{0\}. \tag{10}$$

Furthermore, from Shapiro and Homem-de-Mello (2000), we have the following property: There exists a *finite* set $\Delta \subset T_X(x_0)\backslash\{0\}$ of directions, independent of the sample, such that if $\hat{f}'_N(x_0, d) > 0$ for every $d \in \Delta$, then $\hat{x}_N = x_0$. We call

$$\kappa := \max_{d\in\Delta} = \frac{\mathrm{Var}[F'_\omega(x_0, d)]}{[f'(x_0, d)]^2} \tag{11}$$

the *condition number* of the true problem (1). The above definition is motivated by the result which states that the sample size N required to achieve a given probability of the event "$\hat{x}_N = x_0$" is roughly proportional to the condition number κ. Specifically, for large N, we have from Shapiro, Homem-de-Mello, and Kim (2002, Theorem 1) that the probability of this event can approximated as follows:

$$P(\hat{x}_N = x_0) \approx 1 - \frac{\nu e^{-N/(2\kappa)}}{\sqrt{4\pi N/(2\kappa)}}, \tag{12}$$

where $\nu \geq 1$ is a small constant depending on the problem. It follows by comparing this expression with (9) that for large κ, the exponential constant β is small, and therefore that one needs a large sample to attain a given probability of the event "$\hat{x}_N = x_0$". Such problems can be be viewed as ill conditioned. Note that for a given d, the expression in the right hand side of (11) is the ratio of the variance of $F'_\omega(x_0, d)$ to its squared expected value.

It is impractical to try to compute the numerical value of the condition number κ. (For one thing, it depends on the optimal solution x_0 which, of course, is not known a priori.) Rather, the value of the above analysis is conceptual. It indicates that instances that possess a sharp optimal solution will be well conditioned and can be solved exactly with a small sample size. In other problems, the objective function is flat near the minimizer. These cases correspond to a very large value of κ from (11). The value of κ may even be infinite, since the directional derivative of $f(\cdot)$ may be zero in some directions. (This happens if the set of optimal solutions of the true problem is not a singleton.) For these instances, the sample size necessary to obtain a good solution to the true instance could be quite large. One of the goals of this paper is to test the convergence behavior of the sample average approximation on a number of real stochastic programming instances to empirically verify if this conceptual analysis extends to practice.

4. Estimating the optimal value

Although the results discussed in the previous section indicate a high probability of identifying an exact optimal solution of (1) under certain circumstances, they have little to say about identifying the optimal objective value v^*. Even when N is large and $\hat{x}_N$ is identical to x^* for different samples of size N, the corresponding optimal objective values $\hat{v}_N$ of (6) will in general be different for each sample. Further, the expected value of $\hat{v}_N$ is an underestimate of v^*, as we discuss in Section 4.1.

Sampling methodology can be used to obtain estimates of upper and lower bounds on the optimal value v^* of the true problem (1). In this section, we give details on obtaining these estimates (with approximate confidence intervals).

4.1. *Lower Bound Estimates*

Denote by v^* and $\hat{v}_N$ the optimal values of the true (1) and SAA (6) problems, respectively. It is well known, and can be easily shown, that

$$\mathbb{E}[\hat{v}_N] \leq v^*. \tag{13}$$

The expected value $\mathbb{E}[\hat{v}_N]$ can be estimated as follows. Generate M independent samples. $\xi^{1,j}, \ldots, \xi^{N,j}$, $j = 1, \ldots, M$, each of size N, and solve the corresponding SAA problems

$$\min_{x \in X} \left\{ \hat{f}_N^j(x) := N^{-1} \sum_{i=1}^{N} F(x, \xi^{i,j}) \right\}, \tag{14}$$

for each $j = 1, \ldots, M$. Let $\hat{v}_N^j$ be the optimal value of problem (14), and compute

$$L_{N,M} := \frac{1}{M} \sum_{j=1}^{M} \hat{v}_N^j. \tag{15}$$

The estimate $L_{N,M}$ is an unbiased estimate of $\mathbb{E}[\hat{v}_N]$. Because of the property (13), it provides a statistical lower bound for the true optimal value v^*.

When the M batches $\xi^{1,j}, \xi^{2,j}, \ldots, \xi^{N,j}$, $j = 1, \ldots, M$, are i.i.d. (although the elements *within* each batch do not need to be i.i.d.; they can for instance be obtained by LHS), we have by the Central Limit Theorem that

$$\sqrt{M}[L_{N,M} - \mathbb{E}(\hat{v}_N)] \Rightarrow N\big(0, \sigma_L^2\big), \quad \text{as } M \to \infty, \tag{16}$$

where $\sigma_L^2 = \text{Var}[\hat{v}_N]$ and "$\Rightarrow$" denotes convergence in distribution. The sample variance estimator of σ_L^2 is

$$s_L^2(M) := \frac{1}{M-1}\sum_{j=1}^{M}\left(\hat{v}_N^j - L_{N,M}\right)^2. \tag{17}$$

Defining z_α to satisfy $P\{N(0,1) \leq z_\alpha\} = 1-\alpha$, and replacing σ_L by $s_L(M)$, we can obtain an approximate $(1-\alpha)$-confidence interval for $\mathbb{E}[\hat{v}_N]$ to be

$$\left[L_{N,M} - \frac{z_{\alpha/2}s_L(M)}{\sqrt{M}},\, L_{N,M} + \frac{z_{\alpha/2}s_L(M)}{\sqrt{M}}\right]. \tag{18}$$

Note that the above confidence interval is based on the asymptotic result (16), and therefore is approximate. For small values of M, one can use $t_{\alpha/2,M-1}$ critical values instead of $z_{\alpha/2}$, which will produce slightly bigger confidence intervals.

The idea of the statistical lower bound (15), and statistical upper bounds discussed in the next section, was introduced by Norkin, Pflug, and Ruszczyński (1998) and developed further by Mak, Morton, and Wood (1999).

4.2. *Upper Bound Estimates*

An upper bound can be obtained by noting that for any $\hat{x} \in X$, we have immediately from (1) that $f(\hat{x}) \geq v^*$. Hence, by choosing $\hat{x}$ to be a near-optimal solution (possibly obtained by solving an SAA problem (6)), and by using some unbiased estimator of $f(\hat{x})$, we can obtain an estimate of an upper bound for v^*. In particular, we can generate T independent batches of samples of size $\bar{N}$, denoted by $\xi^{1,j}, \xi^{2,j}, \ldots .\xi^{\bar{N},j}$, $j = 1, 2, \ldots, T$, where each batch has the unbiased property, namely

$$\mathbb{E}\left[\hat{f}_{\bar{N}}^j(x) := \bar{N}^{-1}\sum_{i=1}^{\bar{N}} F(x, \xi^{i,j})\right] = f(x), \quad \text{for all } x \in X. \tag{19}$$

We can then use the average value defined by

$$U_{\bar{N},T}(\hat{x}) := T^{-1}\sum_{j=1}^{T}\hat{f}_{\bar{N}}^j(\hat{x}), \tag{20}$$

as an estimate of $f(\hat{x})$. By applying the Central Limit Theorem again, we have that

$$\sqrt{T}[U_{\bar{N},T}(\hat{x}) - f(\hat{x})] \Rightarrow N\big(0, \sigma_U^2(\hat{x})\big), \quad \text{as } T \to \infty, \tag{21}$$

where $\sigma_U^2(\hat{x}) := \text{Var}\left[\hat{f}_{\bar{N}}(\hat{x})\right]$. We can estimate $\sigma_U^2(\hat{x})$ by the sample variance estimator $s_U^2(\hat{x}; T)$ defined by

$$s_U^2(\hat{x}; T) := \frac{1}{T-1}\sum_{j=1}^{T}\left[\hat{f}_{\bar{N}}^j(\hat{x}) - U_{\bar{N},T}(\hat{x})\right]^2. \tag{22}$$

By replacing $\sigma_U^2(\hat{x})$ with $s_U^2(\hat{x}; T)$, we can proceed as above to obtain a $(1-\alpha)$-confidence interval for $f(\hat{x})$:

$$\left[U_{\bar{N},T}(\hat{x}) - \frac{z_{\alpha/2} s_U(\hat{x}; T)}{\sqrt{T}}, U_{\bar{N},T}(\hat{x}) + \frac{z_{\alpha/2} s_U(\hat{x}; T)}{\sqrt{T}} \right]. \tag{23}$$

4.3. *Estimating a Bound on the Gap*

For a given feasible solution $\hat{x} \in X$ we may wish to estimate the optimality gap $f(\hat{x}) - v^*$. Consider the difference

$$\text{Gap}_{N,M,\bar{N},T}(\hat{x}) := U_{\bar{N},T}(\hat{x}) - L_{N,M}, \tag{24}$$

between the upper and lower bound estimates, defined in (20) and (15), respectively. By the Law of Large Numbers we have that $U_{\bar{N},T}(\hat{x})$ tends to $f(\hat{x})$ with probability one as $\bar{N}T$ tends to ∞, and $L_{N,M}$ tends to $\mathbb{E}[\hat{v}_N]$ with probability one as M tends to ∞. Moreover, under the assumptions (A1), (A2) and (A4), $\mathbb{E}[\hat{v}_N]$ tends to v^* as N tends to ∞. It follows that $\text{Gap}_{N,M,\bar{N},T}(\hat{x})$ tends to $f(\hat{x}) - v^*$ with probability one as N, M, $\bar{N}$, and T all tend to ∞. Of course, if $\hat{x}$ is not an optimal solution, then $f(\hat{x}) - v^*$ is strictly positive. Three factors contribute to the error in the statistical estimator $\text{Gap}_{N,M,\bar{N},T}(\hat{x})$ of the gap $f(\hat{x}) - v^*$. These factors are

(i) variance of $U_{\bar{N},T}(\hat{x})$;
(ii) variance of $L_{N,M}$;
(iii) bias $v^* - \mathbb{E}[\hat{v}_N]$.

Note that $U_{\bar{N},T}(\hat{x})$ and $L_{N,M}$ are unbiased estimators of $f(\hat{x})$ and $\mathbb{E}[\hat{v}_N]$, respectively. Variances of these estimators can be estimated from the generated samples and may be reduced by increasing the respective sample sizes $\bar{N}$, M and T. We have here that $\text{Gap}_{N,M,\bar{N},T}(\hat{x})$ is an unbiased estimator of $f(\hat{x}) - \mathbb{E}[\hat{v}_N]$, and that

$$f(\hat{x}) - \mathbb{E}[\hat{v}_N] \geq f(\hat{x}) - v^*.$$

That is, $\text{Gap}_{N,M,\bar{N},T}(\hat{x})$ overestimates the true gap $f(\hat{x}) - v^*$, and has bias $v^* - \mathbb{E}[\hat{v}_N]$.

The bias constitutes the most serious problem in estimating $f(\hat{x}) - v^*$. We have by the Central Limit Theorem that $N^{1/2}(\hat{f}_N(x) - f(x))$ converges in distribution to $Y(x)$, where each $Y(x)$ is a random variable having normal distribution with mean zero and the same covariance structure as $F(x, \xi(\omega))$. Furthermore, it is known that $N^{1/2}(\hat{v}_N - v^*)$ converges in distribution to $\min_{x \in S} Y(x)$ (see Shapiro, 1991). Under mild additional conditions, we have that the expected value of $N^{1/2}(\hat{v}_N - v^*)$ converges to the expected value of $\min_{x \in S} Y(x)$. Although each $Y(x)$ has mean (expected value) zero, the mean of the minimum of these random variables could be negative and would tend to be smaller for problems with a larger set S of optimal solutions. For a finite sample size N, one may expect that problems having a large set of nearly optimal solutions will exhibit a similar behavior. That is, for ill conditioned problems, this bias may be relatively large and tends to zero at a rate of $O(N^{-1/2})$. The bias can be reduced by increasing the sample size N of the corresponding SAA problems or by sampling more intelligently (by using LHS, say). Of course, an increase in N leads to a larger problem instance to be solved, while increases in $\bar{N}$, M and T to reduce components (i) and (ii) of the error lead only to more instances of the same size to be solved. (See

Kleywegt, Shapiro, and Homem-de-Mello 2001, Section 3.3] for a further discussion of the bias problem.)

5. Testing for (Approximate) optimality

5.1. *Testing for a Unique Solution with Repeated Independent Samples*

Perhaps the simplest scheme for optimality verification is to ascertain whether a phenomenon predicted by the results discussed in Section 3 occurs; that is, to see if the solutions of (6) for a number of different samples of a given size N yield identical solutions $\hat{x}_N$. For stochastic two-stage linear programs with unique optimal solution, we would expect to see the same minimizer $\hat{x}_N$ for almost all samples, for sufficiently large N. Although this test will not in general yield coincident values of $\hat{x}_N$ if the solution set S is not a singleton, or if the problem is poorly conditioned and does not allow identification of the true solution for any reasonable value of N, we can still sometimes use the minimizers $\hat{x}_N$ to learn about the dimensionality of S and its diameter.

Our implementation of this test is as follows. We choose N, and solve a single SAA problem (6) for this N, terminating the algorithm when an upper bound on the estimated relative decrease in the objective function attainable at the next iteration falls below a very tight tolerance. We use the solution $\hat{x}_N$ so calculated as a starting point for solving a number (say, 10) of SAAs for the same N, with different samples, again using a tight convergence tolerance. If all these SAAs terminate without moving away from the starting point $\hat{x}_N$, we conclude that it is likely that $\hat{x}_N$ coincides with x^*. If not, we repeat the entire process for a larger value of N.

5.2. *Comparing several points with repeated independent samples*

Given two candidates $x^0, x^1 \in X$ for an optimal solution of the problem (1), we can validate whether one of these points has a significantly smaller value of the objective function by using the statistical (paired) t-test. We generate T independent batches of samples of size $\bar{N}$, denoted by $\xi^{1,j}, \xi^{2,j}, \ldots, \xi^{\bar{N},j}$, $j = 1, 2, \ldots, T$, where each batch has the unbiased property (19). We have from this property that

$$\mathbb{E}\left[\hat{f}^j_{\bar{N}}(x^1) - \hat{f}^j_{\bar{N}}(x^0)\right] = \mathbb{E}\left[\bar{N}^{-1}\sum_{i=1}^{\bar{N}}\left(F(x^1,\xi^{i,j}) - F(x^0,\xi^{i,j})\right)\right] = f(x^1) - f(x^0),$$

so the average value defined by

$$D_{\bar{N},T}(x^0,x^1) := T^{-1}\sum_{j=1}^{T}\left[\hat{f}^j_{\bar{N}}(x^1) - \hat{f}^j_{\bar{N}}(x^0)\right] \tag{25}$$

can be used as an estimate of $f(x^1) - f(x^0)$. By applying the Central Limit Theorem again, we have that

$$\sqrt{T}\left[D_{\bar{N},T}(x^0,x^1) - (f(x^1) - f(x^0))\right] \Rightarrow N\left(0, \sigma_D^2(x^0,x^1)\right), \quad \text{as } T \to \infty, \tag{26}$$

where $\sigma_D^2(x^0, x^1) := \mathrm{Var}[\hat{f}_{\bar{N}}(x^1) - \hat{f}_{\bar{N}}(x^0)]$. We can use the sample variance estimator $s_D^2(x^0, x^1; T)$ defined by

$$s_D^2(x^0, x^1; T) := \frac{1}{T-1} \sum_{j=1}^{T} \left[\hat{f}_{\bar{N}}^j(x^1) - \hat{f}_{\bar{N}}^j(x^0) - D_{\bar{N},T}(x^0, x^1) \right]^2$$

to construct a $(1-\alpha)$-confidence interval for $f(x^1) - f(x^0)$, as follows:

$$\left[D_{\bar{N},T}(x^0, x^1) - \frac{z_{\alpha/2} s_D(x^0, x^1; T)}{\sqrt{T}}, D_{\bar{N},T}(x^0, x^1) + \frac{z_{\alpha/2} s_D(x^0, x^1; T)}{\sqrt{T}} \right]. \tag{27}$$

Mak, Morton, and Wood (1999) use a common-random numbers technique like that described above in connection with estimating the optimality gap defined in Section 4.3.

5.3. *Testing KKT conditions I*

In discussing the tests of optimality in this section, we assume that the feasible set X for the problem (1) and the SAA problem (6) is defined by linear constraints, that is,

$$X := \left\{ x \in \mathbb{R}^n : a_i^T x + b_i \leq 0,\ i \in \mathcal{I} \right\}, \tag{28}$$

where $\mathcal{I}$ is a finite index set. It follows that X is polyhedral.

Another test for near-optimality of a candidate solution to (1) is based on statistical verification of the first-order Karush-Kuhn-Tucker (KKT) optimality conditions, using subgradients from the sample-average approximations (6) to build up an approximation to the subdifferential of $f(\cdot)$ in the vicinity of the candidate solution $\hat{x}$. Higle and Sen (1996, Section 3) discuss a test that is related to the one described below. It differs in that it does not attempt to calculate explicitly the distance between the subdifferential of f and the normal cone to X at $\hat{x}$, but rather uses a kind of sampling technique to estimate this distance.

We discuss first the situation in which $f(\cdot)$ is differentiable at $\hat{x}$: that is, its subdifferential $\partial f(\hat{x})$, at $\hat{x}$, is a singleton. In the case of two-stage linear stochastic programming the (optimal value) function $Q(\cdot, \xi)$ is convex, and since it is assumed that the corresponding expected value function is real valued, the subdifferential of $f(x)$ can be taken inside the expectation operator (see, e.g., Ioffe and Tihomirov, 1979, Theorem 4, page 381). It follows that $\partial f(\hat{x})$ is a singleton if $\partial Q(\hat{x}, \xi(\omega))$ is a singleton with probability one. This typically holds if the involved probability distributions are continuous. On the other hand if the number of scenarios is finite, then the expected value function is piecewise linear and cannot be differentiable everywhere except in the trivial case in which it is affine. Nevertheless, it may still be differentiable at the considered point $\hat{x}$.

Assuming that $F(\cdot, \xi(\omega))$ is differentiable at a given point x with probability one, we can (under mild regularity conditions and, in particular, if $F(\cdot, \xi(\omega))$ is convex and $f(\cdot)$ real valued) take the derivatives inside the expected value, that is,

$$\nabla f(x) = \mathbb{E}[\nabla_x F(x, \xi(\omega))]. \tag{29}$$

The following first-order optimality conditions hold at an optimal solution x^* of the true problem (1). There exist Lagrange multipliers $\lambda_i \geq 0$, $i \in \mathcal{I}(x^*)$, such that

$$0 \in \partial f(x^*) + \sum_{i \in \mathcal{I}(x^*)} \lambda_i a_i, \tag{30}$$

where

$$\mathcal{I}(x^*) := \left\{ i \in \mathcal{I} : a_i^T x^* + b_i = 0 \right\}$$

is the index set of constraints active at x^*. In particular, if $f(\cdot)$ is differentiable at x^*, that is, $\partial f(x^*) = \{\nabla f(x^*)\}$ is a singleton, then the conditions (30) can be written in the familiar KKT form:

$$\nabla f(x^*) + \sum_{i \in \mathcal{I}(x^*)} \lambda_i a_i = 0. \tag{31}$$

We can also write (31) in the form

$$-\nabla f(x^*) \in N_X(x^*), \tag{32}$$

where $N_X(x^*)$, the normal cone to X at x^*, can be written as follows:

$$N_X(x^*) = \left\{ \sum_{i \in \mathcal{I}(x^*)} \lambda_i a_i : \lambda_i \geq 0, \quad i \in \mathcal{I}(x^*) \right\}. \tag{33}$$

Given a candidate solution $\hat{x}$, we propose in our KKT test to estimate the distance

$$\delta(\hat{x}) := \mathrm{dist}(-\nabla f(\hat{x}), N_X(\hat{x})) \ ,$$

where dist (a, C) denotes the distance from point a to set C. The KKT conditions hold at $\hat{x}$ if and only if $\delta(\hat{x}) = 0$.

Assuming that equation (29) holds at the point $\hat{x}$, the gradient $\nabla \hat{f}_N(\hat{x})$ of the sample-average function provides an unbiased and consistent estimator of $\nabla f(\hat{x})$. It follows that the quantity $\hat{\delta}_N(\hat{x})$ defined by

$$\hat{\delta}_N(\hat{x}) := \mathrm{dist}(-\nabla \hat{f}_N(\hat{x}), N_X(\hat{x})) \tag{34}$$

gives a consistent estimator of $\delta(\hat{x})$. Moreover, the covariance matrix of $\nabla \hat{f}_N(\hat{x})$ can be estimated, allowing a confidence region for $\nabla f(\hat{x})$ to be constructed, and therefore allowing a statistical validation of the KKT conditions. This test is discussed in detail in Shapiro and Homem-de-Mello (1998).

Consider now the case in which the function $f(\cdot)$ in (1) is convex but is not differentiable everywhere, as typically happens in the two-stage linear problem with recourse when the distribution is discrete. We can write the first-order conditions (30) as follows:

$$\exists \gamma \in \partial f(x^*) \text{ such that } -\gamma \in N_X(x^*). \tag{35}$$

Defining

$$\Delta(\hat{x}) := \inf_{\gamma\in\partial f(\hat{x})} \operatorname{dist}(-\gamma, N_X(\hat{x})) , \tag{36}$$

we have that $\triangle(\hat{x}) = 0$ if and only if $\hat{x}$ satisfies the conditions (30). Similarly to (34), we estimate $\Delta(\hat{x})$ by $\hat{\Delta}_N(\hat{x})$ defined as follows:

$$\hat{\Delta}_N(\hat{x}) := \inf_{\gamma\in\partial f(\hat{x})} \operatorname{dist}(-\gamma, N_X(\hat{x})). \tag{37}$$

It is known that the Hausdorff distance between $\partial f(x)$ and its sample-average estimate $\partial \hat{f}_N(x)$ converges with probability 1 to zero, and moreover the convergence is uniform in x on any compact subset of $\mathbb{R}^n$ if the distribution of $\xi(\omega)$ has a finite support (Shapiro and Homem-de-Mello, 2000, Lemma 2.4). In this sense, $\partial \hat{f}_N(x)$ provides a consistent estimator of $\partial f(x)$, and it follows that $\hat{\Delta}_N(\hat{x})$ is a consistent estimator of $\Delta(\hat{x})$.

It is, of course, impractical to calculate the entire subdifferential $\partial \hat{f}_N(\hat{x})$. However, when$f$ is convex and piecewise linear (as in the case of two-stage linear programs with recourse), then in the neighborhood of any point $\hat{x}$ there exist a finite number of points $x^1, x^2, \ldots, x^\ell$ such that $\mathcal{Q}$ is differentiable at these points and $\partial f(\hat{x}) = \operatorname{conv}\{\nabla f(x^1), \nabla f(x^2), \ldots, \nabla f(x^\ell)\}$. This observation suggests the following test.

(i) Generate M points x^j, $j = 1, 2, \ldots, M$, randomly distributed in a small neighborhood of $\hat{x}$.
(ii) Calculate an estimate $\hat{\gamma}_j$ of a subgradient of$f$ at each point x^j. This can be done either by using a common sample for all subgradients, that is, $\hat{\gamma}_j \in \partial \hat{f}_N(x^j)$, where $\hat{f}_N$ is defined by the same sample for each argument x^j; or by using a different sample for each point x^j.
(iii) Calculate the minimal distance of the convex hull of $\hat{\gamma}_j$, $j = 1, 2, \ldots, M$, to $-N_X(\hat{x})$, given by the optimal value $\hat{\delta}_N^M(\hat{x})$ of the following problem:

$$\min_\alpha \operatorname{dist}\left(-\textstyle\sum_{j=1}^M \alpha_j \hat{\gamma}_j, N_X(\hat{x})\right), \text{ subject to} \tag{38}$$

$$\sum_{j=1}^M \alpha_j = 1, \quad \alpha_j \geq 0, \quad j = 1, 2, \ldots, M.$$

The convex hull of $\hat{\gamma}_j$, $j = 1, 2, \ldots, M$. converges (in the Hausdorff metric) to $\partial f(\hat{x})$ with probability 1 as N and M both tend to infinity. It follows that (38) yields a consistent estimate of $\delta(\hat{x})$.

Note that the convex hull of $\hat{\gamma}_j$, $j = 1, 2, \ldots, M$, actually converges to the convex hull of the set $\cup_{\|x-\hat{x}\|\leq\epsilon}\partial f(x)$, where $\epsilon > 0$ defines a small neighborhood of $\hat{x}$. Although, since $f(\cdot)$ is piecewise linear, for ϵ sufficiently small this set will coincide with $\partial f(\hat{x})$, we should in practice choose ϵ to be a value corresponding to the error we are prepared to tolerate in $\hat{x}$. That is, if we wish only to verify that $\hat{x}$ is within ϵ of an exact solution, it suffices to use this value of ϵ in the test.

5.4. *Testing KKT conditions II*

Here we consider a variant of the test proposed in the previous section. As above, the aim is to estimate the quantity $\Delta(\hat{x})$ defined by (36). Rather than building up an estimate of the subgradient by taking randomly distributed points in a small neighborhood of $\hat{x}$, we use different samples of size N to generate different elements of the true subdifferential $\partial f(\hat{x})$. The approach is as follows.

(i) For given N and M, choose M samples of size N that is, $\{\xi^{i,j}, \xi^{2,j}, \ldots, \xi^{N,j}\}$, $j = 1, 2, \ldots, M$.
(ii) For each $j = 1, 2, \ldots, M$, and defining $\hat{f}_N^j$ as in (14), calculate an element $\hat{\gamma}_j \in \partial \hat{f}_N^j(\hat{x})$.
(iii) Calculate the minimal distance of the convex hull of $\hat{\gamma}_j$, $j = 1, 2, \ldots, M$, to $-N_X(\hat{x})$ by solving (38).

Note that this test is "stricter" than the test of the previous section, because it constructs an approximation to $\partial f(\hat{x})$ by sampling subgradients at the fixed point $\hat{x}$. Consequently, unless $\partial f(\hat{x})$ is a singleton, there is no guarantee that the convex hull of $\hat{\gamma}_j$, $j = 1, 2, \ldots, M$, converges (in the Hausdorff metric) to $\partial f(\hat{x})$ with probability one as N and M tend to infinity. Hence, in general, this test is not a consistent KKT test.

6. An algorithm for two-stage stochastic linear programs with recourse

In this section, we focus our attention on two-stage stochastic linear programs with recourse over a discrete scenario space, which are the subjects of our computational experiments. We discuss briefly the algorithm used in the experiments, referring the reader to Linderoth and Wright (2003) for further details.

The problem we consider is (2), with second-stage problems defined by (3), with fixed recourse, and with a finite number K of scenarios. We thus have $\mathcal{Q}$ defined by (4), where ξ_k defines the data vector (q_k, h_k, T_k, W), $k = 1, 2, \ldots, K$. The optimality conditions for each second-stage problem (3) with $\xi = \xi_k$ are that there exists a vector π_k such that

$$0 \le q_k - W^T \pi_k \perp y_k \ge 0,\ W y_k = h_k - T_k \hat{x}.$$

(The symbol $\perp$ represents a complementarity condition, which we can also state as requiring that the inner product of these two nonnegative vectors must be zero.) It can be shown that

$$-T_k^T \pi_k \in \partial Q(x, \xi_k).$$

From Rockafellar (1970, Theorem 23.8), we have that

$$c + \sum_{k=1}^{K} p_k \partial Q(x, \xi_k) = c + \partial \mathcal{Q}(x),$$

and therefore

$$c - \sum_{k=1}^{K} p_k T_k^T \pi_k \in c + \partial \mathcal{Q}(x). \tag{39}$$

Since the evaluation of each $Q(x, \xi_k)$ requires solution of (3) and since π_k is simply the dual solution of this linear program, it follows that we can obtain an element of the subdifferential for the objective function in (2) for little more than the cost of evaluating the function alone.

Our tests require the solution of several SAA problems (6), which have the same form as described above except that we work with a subset of N (instead of K) scenarios, and the probabilities $p_k = 1/N$.

The algorithm described in Linderoth and Wright (2003) uses subgradient information to construct a lower bounding piecewise-linear approximation to the objective function in (2). Candidate iterations are generated by finding the minimizer of this model subject to the constraints $Ax = b$, $x \geq 0$ from (2) and a trust-region constraint of the form $\|x - x^I\|_\infty \leq \Delta$, where x^I is the "incumbent" (roughly speaking, the best point found so far), and Δ is a trust-region radius. This bundle-trust-region subproblem can be formulated as a linear program and therefore solved efficiently. Convergence to a solution (existence of a solution is specified in the assumption (A4)) is assured by means of rules for adjusting the value of Δ, for accepting candidate iterates as the new incumbent or rejecting them, and for managing the subgradient information. The algorithm is terminated when an upper bound on the relative improvement in the objective function attainable at the next iteration falls below a specified tolerance.

7. Computational grids

The algorithm described in Section 6 is implemented to run on a computational grid built using Condor (Livny et al., 1997). The Condor system manages distributively owned collections of workstations known as *Condor pools*. A unique and powerful feature of Condor is that each machine's owner specifies the conditions under which jobs from other users are allowed to run on his machine. The default policy is to terminate a Condor job when a workstation's owner begins using the machine, and migrate it to some other machine. Under this policy, Condor jobs use only compute cycles that would have otherwise been wasted. Because of the minimal intrusion of the Condor system, workstation owners often are quite willing to contribute their machines to Condor pools, and large pools can be built. Table 1 gives summary statistics about the resources that made up the computational grid used in many of the experiments of Section 9.

The pool of workers typically grows and shrinks dynamically during the computation, as some workers become available to the job and others are reclaimed by their owners or by other Condor users. It is important for algorithms using this platform to be able to handle the dynamism of the environment, the possibility that the computations on some workers may never be completed, and the high latencies in transferring information between master and workers. To help implement our algorithm on the Condor platform, we used the runtime support library MW (Goux et al., 2000, 2001), which supports parallel computations of the master-worker type on the computational grid.

Table 1 Computational Grid used for Experiments

Number	Type of Machine	Location
414	Intel/Linux PC	Argonne
579	Intel/Linux PC	Wisconsin
31	Intel/Solaris PC	Wisconsin
133	Sun/Solaris Workstation	Wisconsin
512	Intel/Linux PC	New Mexico

Table 2 Test Problem Dimensions

Name	Application	Scenarios	First-Stage Size	Second-Stage Size
20term	Vehicle Assignment	1.1×10^{12}	(3, 64)	(124, 764)
gbd	Aircraft Allocation	6.5×10^{5}	(4, 17)	(5, 10)
LandS	Electricity Planning	10^{6}	(2, 4)	(7, 12)
ssn	Telecom Network Design	10^{70}	(1, 89)	(175, 706)
storm	Cargo Flight Scheduling	6×10^{81}	(185, 121)	(528, 1259)

A typical computation in this environment proceeds as follows: A master processor (typically the user's workstation or a powerful server) maintains a queue of tasks which it wishes the worker processors to solve, and a list of currently available workers. MW manages these queues and takes responsibility for assigning tasks to available workers. When a worker completes its task, it returns the results to the master, which performs some computation in response, possibly generating more tasks for the task queue.

Parallelism in the algorithmic approach of Section 6 arises in two places. First, solution of the second-stage linear programs can be carried out independently for each scenario ξ_k. Typically, we group these scenarios into "chunks" and assign the solution of each chunk to a worker processor. The master processor manages the function and subgradient information returned by the workers. The second source of parallelism arises from considering more than one candidate iterate at a time. We maintain a "basket" of possible new incumbents; at any given time, tasks for evaluating the objective function for each iterate in the basket are being performed on the workers. When evaluation of the function for a point in the basket is completed, the master decides whether or not to make it the new incumbent and solves a trust-region subproblem to generate a new candidate. Consideration of more than one candidate at a time makes the algorithm less "synchronous," making it less sensitive to slow or unreliable workers.

For the statistical tests that do not require the solution of a master problem, the batches or scenarios can be processed completely independently of each other. No centralized control is needed, except to gather statistics and results of the runs. Such solution approaches lend themselves extremely well to computing platforms where the processors themselves are diverse and loosely coupled, such as the computational grid used in our experiments.

8. Test problems

Here we briefly describe the test problems used in our experiments. All are two-stage stochastic linear programs with recourse obtained from the literature, though in some cases we refined the scenario distribution to increase the total number of scenarios. The input data sets we used, in SMPS format, can be obtained from the following site:

`www.cs.wisc.edu/~swright/stochastic/sampling/`

8.1. *20term*

This problem is described by Mak, Morton, and Wood (1999). It is a model of a motor freight carrier's operations, in which the first-stage variables are positions of a fleet of vehicles at the beginning of a day, and the second-stage variables move the fleet through a network to satisfy point-to-point demands for shipments (with penalties for unsatisfied demands) and to finish the day with the same fleet configuration as at the start.

8.2. *gdb*

This problem is derived from the aircraft allocation problem described by Dantzig (1963, Chapter 28). Aircraft of different types are to be allocated to routes in a way that maximizes profit under uncertain demand. Besides the cost of operating the aircraft, there are costs associated with bumping passengers when the demand for seats outstrips the capacity. In this model, there are four types of aircraft flying on five routes, and the first-stage variables are the number of aircraft of each type allocated to each route. (Since three of the type-route pairs are infeasible, there are 17 variables in all.) The first-stage constraints are bounds on the available number of aircraft of each type. The second-stage variables indicate the number of carried passengers and the number of bumped passengers on each of the five routes, and the five second-stage constraints are demand balance equations for the five routes. The demands (the right-hand sides of the five demand balance equations) are random. In the original model, there are a total of only 750 scenarios. By refining the possible demand states (to allow between 13 and 17 states on each route), we obtain a model with 646, 425 scenarios.

8.3. *LandS*

Our test problem LandS is a modification of a simple problem in electrical investment planning presented by Louveaux and Smeers (1988). The four first stage variables represent capacities of different new technologies, and the 12 second-stage variables represent production of each of three different modes of electricity from each of the four technologies technologies. First-stage constraints represent minimum total capacity and budget restrictions. The second-stage constraints include capacity constraints for each of the four technologies, and three demand constraints, which have the form

$$\sum_{i=1}^{4} y_{ij} \ge \xi_j,\ j = 1, 2, 3,$$

where ξ_1, ξ_2, and ξ_3 are the (possibly uncertain) demands. All other data is deterministic.

In [25], ξ_2 and ξ_3 are fixed at 3 and 2, respectively, while ξ_1 takes on three possible values. The total number of scenarios is therefore 3. We modified the problem to allow 100 different values for each of these random parameters, namely,

$$\xi_j = .04(k-1),\ \ k = 1, 2, \ldots, 100,\ \ j = 1, 2, 3,$$

each with probability .01. Since we assume independence of these parameters, the total number of scenarios is 10^6, each with equal probability 10^{-6}.

8.4. *SSN*

The SSN problem of Sen, Doverspike, and Cosares (1994) arises in telecommunications network design. The owner of the network sells private-line services between pairs of nodes in the network. When a demand for service is received, a route between the two nodes with sufficient bandwidth must be identified for the time period in question. If no such route is available, the demand cannot be satisfied and the sale is lost. The optimization problem is to decide where to add capacity to the network to minimize the expected rate of unmet requests. Since the demands are random (the scenarios are based on historical demand patterns), this is a stochastic optimization problem.

The formulation in Sen, Doverspike, and Cosares (1994) is as follows:

$$\min \mathcal{Q}(x) \text{ subject to } e^T x \leq b, \; x \geq 0, \tag{40}$$

where x is the vector of capacities to be added to the arcs of the network, e is the vector whose components are all 1, b (the budget) is the total amount of capacity to be added, and $\mathcal{Q}(x)$ is the expected cost of adding capacity in the manner indicated by x. We have $\mathcal{Q}(x) = \mathbb{E}_p Q(x, d(\omega))$, where

$$Q(x, d) := \min_{f,s} \{e^T s : Af \leq x + c, Ef + s = d(\omega), (f, s) \geq 0\}. \tag{41}$$

Here, $d = d(\omega)$ is the random vector of demands; e is the vector whose components are all 1; the columns of A are incidence vectors that describe the topology of the network; f is the vector whose components are the number of connections between each node pair that use a certain route; c is the vector of current capacities; s is the vector of unsatisfied demands for each request; and the rows of E have the effect of summing up the total flow between a given node pair, over all possible routes between the nodes in that pair.

In the data set for SSN, there are total of 89 arcs and 86 point-to-point pairs; that is, the dimension of x is 89 and of $d(\omega)$ is 86. Each component of $d(\omega)$ is an independent random variable with a known discrete distribution. Specifically, there are between three and seven possible values for each component of $d(\omega)$, giving a total of approximately 10^{70} possible complete demand scenarios.

8.5. *Storm*

This problem is based on a cargo flight scheduling application described by Mulvey and Ruszczyński (1995, p. 486 et seq.). The aim is to plan cargo-carrying flights over a set of routes in a network, where the amounts of cargo are uncertain. Each first-stage variable represents the number of aircraft of a certain type assigned to a certain route. The first-stage equations represent minimum frequencies for flights on each route and balance equations for arrival and departure of aircraft of each type at the nodes in the network. Second-stage variables include the amounts of cargo to be shipped between nodes of the network, the amounts of undelivered cargo on each route, and the unused capacity on each leg of each route. Second-stage constraints include demand constraints and balance equations. Only the demands (the right-hand sides of the 118 demand constraints) are random. Mulvey and Ruszczyński (1995) generate scenarios from a uniform distribution of $\pm 20\%$ around the basic demand forecast. In our data set, we used five scenarios for each demand, namely, the multiples .8, .9. 1.0, 1.1 and 1.2 of the basic demand. Each scenario was assigned a probability of 0.2. Since the demand scenarios were allowed to vary independently, our model has a total of 5^{118} scenarios, or approximately 6×10^{81}.

9. Computational results

9.1. *Optimal value estimates*

Summaries of our experiments to determine upper and lower bounds on the optimal objective values are shown in Tables 3 and 4. Our experimental procedure is as described in Sections 4.1 and 4.2. For lower bound estimates, we calculated values of $\hat{v}_N^j$ (see (14)) for $j = 1, 2, \ldots, M$, with M between 7 and 10, and $N = 50, 100, 500, 1000, 5000$. The 95% confidence interval (18) for $\mathbb{E}[\hat{v}_N]$ is tabulated for each problem and for each sample size N in Tables 3 and 4. Monte Carlo (MC) was used to select the samples in Tables 3, while Latin Hypercube sampling (LHS) was used in Table 4.

These tables also show upper bound estimates. For the computed minimizer $\hat{x}_N^j$ in each of the lower-bound trials $j = 1, 2, \ldots, M$, we estimated of $f(\hat{x}_N^j)$ by sampling $T = 50$

Table 3 Lower and Upper Bound Estimates for v^*, Monte Carlo Sampling

Problem	N	$E\hat{v}_N$ (95% conf. int.)	Best $f(\hat{x}_N^j)$ (95% conf. int.)
20term	50	253361.33 ± 944.06	254317.96 ± 19.89
20term	100	254024.89 ± 800.88	254304.54 ± 21.20
20term	500	254324.33 ± 194.51	254320.39 ± 27.36
20term	1000	254307.22 ± 234.04	254333.83 ± 18.85
20term	5000	254340.78 ± 85.99	254341.36 ± 20.32
gbd	50	1678.62 ± 66.73	1655.86 ± 1.34
gbd	100	1595.24 ± 42.41	1656.35 ± 1.19
gbd	500	1649.66 ± 13.60	1654.90 ± 1.46
gbd	1000	1653.50 ± 12.32	1655.70 ± 1.49
gbd	5000	1653.13 ± 4.37	1656.40 ± 1.31
LandS	50	227.19 ± 4.03	225.71 ± 0.12
LandS	100	226.39 ± 3.99	225.55 ± 0.12
LandS	500	226.02 ± 1.43	225.61 ± 0.12
LandS	1000	225.96 ± 0.76	225.70 ± 0.13
LandS	5000	225.72 ± 0.52	225.70 ± 0.12
ssn	50	4.11 ± 1.23	12.68 ± 0.05
ssn	100	7.66 ± 1.31	11.20 ± 0.05
ssn	500	8.54 ± 0.34	10.28 ± 0.04
ssn	1000	9.31 ± 0.23	10.09 ± 0.03
ssn	5000	9.98 ± 0.21	9.86 ± 0.05
storm	50	15506271.7 ± 22043.4	15499092.17 ± 845.26
storm	100	15482549.9 ± 19213.8	15499056.00 ± 623.30
storm	500	15498139.8 ± 4152.8	15498468.02 ± 684.65
storm	1000	15500879.4 ± 4847.1	15498893.02 ± 695.90
storm	5000	15498121.3 ± 1879.0	15498646.89 ± 696.08

Table 4 Lower and Upper Bound Estimates for v^*, Latin Hypercube Sampling

Problem	N	$E\hat{v}_N$ (95% conf. int.)	Best $f(\hat{x}_N^j)$ (95% conf. int.)
20term	50	254307.57 ± 371.80	254328.69 ± 4.73
20term	100	254387.00 ± 252.13	254312.50 ± 5.77
20term	500	254296.43 ± 117.95	254315.82 ± 4.59
20term	1000	254294.00 ± 95.22	254310.33 ± 5.23
20term	5000	254298.57 ± 38.74	254311.55 ± 5.56
gbd	50	1644.21 ± 10.71	1655.628 ± 0.00
gbd	100	1655.62 ± 0.00	1655.628 ± 0.00
gbd	500	1655.62 ± 0.00	1655.628 ± 0.00
gbd	1000	1655.62 ± 0.00	1655.628 ± 0.00
gbd	5000	1655.62 ± 0.00	1655.628 ± 0.00
LandS	50	222.59 ± 2.75	225.647 ± 0.004
LandS	100	225.57 ± 0.16	225.630 ± 0.004
LandS	500	225.65 ± 0.05	225.628 ± 0.004
LandS	1000	225.64 ± 0.03	225.633 ± 0.005
LandS	5000	225.62 ± 0.02	225.624 ± 0.005
ssn	50	10.10 ± 0.81	11.380 ± 0.023
ssn	100	8.90 ± 0.36	10.542 ± 0.021
ssn	500	9.87 ± 0.22	10.069 ± 0.026
ssn	1000	9.83 ± 0.29	9.996 ± 0.025
ssn	5000	9.84 ± 0.10	9.913 ± 0.022
storm	50	15497683.7 ± 1078.8	15498722.14 ± 17.97
storm	100	15499255.3 ± 1011.7	15498739.40 ± 18.34
storm	500	15498661.4 ± 280.8	15498733.67 ± 17.71
storm	1000	15498598.1 ± 148.5	15498735.73 ± 19.93
storm	5000	15498657.8 ± 73.9	15498739.41 ± 19.11

batches of $\bar{N} = 20000$ scenarios, thereby deriving an estimate $U_{\bar{N},T}(\hat{x}_N^j)$ from the formula (20). We then selected the point $\hat{x}_N^j$ with the lowest value of $U_{\bar{N},T}(\hat{x}_N^j)$, and generated a new, independent set of $T = 50$ batches of $\bar{N} = 20000$ scenarios, to obtain a new estimate $U_{\bar{N},T}(\hat{x}_N^j)$ and a 95% confidence interval from (23). These are the intervals reported in the last columns of Tables 3 and 4.

Further details of these experiments are given in the appendix of the technical report (Linderoth, Shapiro, and Wright, 2002), which is an expanded version of this paper.

Several comments about these results are in order. The use of LHS produced significant improvements over MC sampling on all problems; in fact, the improvements were dramatic for all problems except SSN. The bias $v^* - \mathbb{E}[\hat{v}_N]$, which can be estimated by comparing the corresponding entries in the last two columns of Tables 3 and 4, is significantly better with the LHS scheme of Table 4. For the two smaller problems gbd and LandS, the results from Table 4 strongly suggest that the optimal solution value is identified exactly; we discuss this issue further in Section 9.2. For 20term and storm, the relative gap appears small, with storm showing a remarkable improvement in the variance of the lower bound estimate $\hat{v}_N$ with the use of LHS. For 20term, gbd, LandS, and storm, the upper bound estimate is remarkably unaffected by the value of N, suggesting that the approximate solutions $\hat{x}_N^j$ obtained even for sample sizes $N = 100$ are of similar quality to this obtained for $N = 5000$. This fact is particularly notable in the case of storm, a problem with a huge number of scenarios.

The conditioning of SSN is significantly worse than than of the other problems. A sample size of at least $N = 500$ was required to identify approximate solutions of reasonable quality, and the quality continues to improve as N is increased to 1000 and 5000. LHS produces improvements in both the optimality gap and the variance of the upper- and lower-bound estimates, and our results indicate strongly that the true optimal value is about 9.90.

These experiments were run over a number of weeks on the computational grid detailed in Table 1. Our goal was not to maximize the efficiency of these runs (which is a difficult task, in any case, on a dynamic computational platform such as this one). Nevertheless, a few words on the computational effort are in order. The number of processors used at any one time was typically fewer than 100. The wall clock time required to solve a single SAA problem ranged from less than one minute (for gbd with $N = 50$) to around 30–45 minutes (for SSN with $N = 5000$). (Recall that 7–10 such problems are solved per line of Tables 3 and 4.) For each line in the last column of Table 3. $T\bar{N} = 10^6$ second-stage linear programs were solved. Each such instance typically required 25–45 processors and 15–30 minutes of wall clock time. The upper bound estimates using Latin Hypercube samples (shown in the last column of Table 4) required considerably more effort. Rather than generate all the scenarios and pass them to the workers (which would have required too much communication), we send only a random seed for each batch to each worker, and require the worker to generate the full set of scenarios, stopping once it has calculated its own "slice" of scenarios. Hence, the runtimes are dominated by sampling computations rather than by optimization calculations. To compute each line of the last column in Table 4, about 80–120 processors and 0.5–3 hours of wall clock time were typically required,

9.2. *Convergence of approximate solution*

Taking the approximate solutions $\hat{x}_N^j$, $j = 1, 2, \ldots, M$ in the Latin Hypercube experiments with $N = 5000$, we calculated pairwise distances between six of these solutions, for the problems 20term, gbd, LandS, SSN, and storm. The resulting "distance matrices" are shown in Tables 5–9. Table 6 confirms that we appear to have identified an exact solution for gbd; that is, $\hat{x}_N^j \equiv x^*$, where $\{x^*\} = S$ is the unique solution of (2), (3), (4). For 20term and SSN, Tables 5 and 8 suggest an "ill conditioned" problem—one in which a small change to the data results in a large change to the solution set S. For these problems, it is quite possible that S is a set with a fairly large diameter. It is even more likely that S lies in a shallow basin; that is, there is a large, feasible neighborhood of S within which the value of f is not much different from the optimal value v^*. For the storm problem, on the other hand, Table 9 indicates that the approximate solutions $\hat{x}_N^j$, $j = 1, 2, \ldots, M$ are quite close together for $N = 5000$, suggesting that they may be converging to a unique minimizer x^* that will be identified at a larger sample size N. We report below on further investigations of this issue, but note here that the result is somewhat surprising; given the very large number of scenarios for this problem we might have expected it to display similar symptoms of ill conditioning to 20term or SSN.

We performed the repeated sampling test of Section 5.1 on the gbd, LandS, and storm problems. For gbd and LandS, we solved an SAA with $N = 5000$ (using LHS), then solved 10 more different instances with different samples of the same size, with a very tight convergence

Table 5 Distance Matrix for Six SAAs of 20term with $N = 5000$, Latin Hypercube Sampling

0.000	16.1	26.5	296	278	263
16.1	0.000	20.3	297	279	264
26.5	20.3	0.000	291	273	259
296	297	291	0.000	49.2	67.0
278	279	273	49.2	0.000	31.9
263	264	259	67.0	31.9	0.000

Table 6 Distance Matrix for Six SAAs of gbd with $N = 5000$, Latin Hypercube Sampling

0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00

Table 7 Distance Matrix for Six SAAs of LandS with $N = 5000$, Latin Hypercube Sampling

0.000	0.0566	0.0980	0.0566	0.0566	0.0800
0.0566	0.000	0.0980	0.000	0.000	0.0566
0.0980	0.0980	0.000	0.0980	0.0980	0.1497
0.0566	0.000	0.0980	0.000	0.000	0.0566
0.0566	0.000	0.0980	0.000	0.000	0.0566
0.0800	0.0566	0.1497	0.0566	0.0566	0.000

Table 8 Distance Matrix for Six SAAs of SSN with $N = 5000$, Latin Hypercube Sampling

0.00	19.9	13.3	22.0	16.9	21.8
19.9	0.00	21.6	27.3	23.0	18.1
13.3	21.6	0.00	22.4	19.5	22.3
22.0	27.3	22.4	0.00	26.4	30.6
16.9	23.0	19.5	26.4	0.00	26.7
21.8	18.1	22.3	30.6	26.7	0.00

tolerance of 10^{-12}. The results are as expected from the observations above; all 10 additional trials terminated without stepping away from their starting point, effectively confirming that all instances have the same solution $\hat{x}_N$.

For storm, we tried repeated sampling with sample sizes $N = 20000$ and $N = 50000$ (with LHS), but did not obtain convergence of the solution of the SAA to an identical value. Close examination of the solutions, however, suggested a solution set S with a very specific structure. In the 10 trials for $N = 20000$, the 10 solutions were essentially identical in all but one of their 121 components. The exception was the 112th component, which varied between 1.46003333 and 1.47593333. In the 10 trials for $N = 50000$, the 10 solutions were again identical in all components except the 112th, which took values between 1.46123333 and 1.47210000. It seems likely that the condition number κ as defined by (11) is very large (possibly infinite), yet this problem is quite well behaved, in that for small ε, the ε-optimal solution set S_ε is close to being a short line segment, rather than a large multidimensional basin.

Table 9 Distance Matrix for Six SAAs of storm with $N = 5000$, Latin Hypercube Sampling

0.000	0.0187	0.0037	0.0105	0.0147	0.0069
0.0187	0.000	0.0153	0.0095	0.0078	0.0124
0.0037	0.0153	0.000	0.0069	0.0122	0.0033
0.0105	0.0095	0.0069	0.000	0.0097	0.0036
0.0147	0.0078	0.0122	0.0097	0.000	0.0105
0.0069	0.0124	0.0033	0.0036	0.0105	0.000

For SSN, we tested with two SAAs at each sample size $N = 10000$, 20000, and 40000, all using LHS, but found that the two solutions obtained for each value of N differed. We conclude that either the solution set S for SSN is not a singleton, or else that the values of N we tried were not large enough to identify the unique solution x^*.

9.3. *Comparison of several points*

We performed a comparison of several candidate solution points for the SSN problem, using the technique described in Section 5.2. We compared three points. The first point x^0 was obtained by solving an SAA with sample size $N = 250000$, generated using LHS. To obtain the second point x^1, we generated an SAA with a *different* sample of size $N = 250000$, and applied our trust-region algorithm starting from the first point x^0, with initial trust-region radius 10^{-3}. The point x^1 was the first point accepted by this algorithm as having a lower value of $\hat{f}_N$ for this sample. For the point x^2, we used the same SAA as was used to obtain x^1, but this time applied our algorithm with an initial trust-region radius of 1. The point x^2 was the first point accepted by the algorithm. (At the time it was generated, the trust-region radius had been reduced to .4 by the algorithm.)

We applied the test of Section 5.2 $\bar{N} = 25000$ (samples generated by LHS) and $T = 2000$ to obtain confidence intervals for the function values at x^0, x^1, and x^2, and the gaps $f(x^1) - f(x^0)$ and $f(x^2) - f(x^0)$. Results are shown in Table 10. They show that x^0 almost certainly has the lowest function value of the three. In this sense, the second SAA (which was used to obtain x^1 and x^2) appears to give a less accurate approximation to the true function f in the vicinity of these three points than does the first SAA (for which x^0 was the minimizer). It also confirms the impression gained from Table 4 that the optimal value x^* for SSN is very close to 9.90. We can also conclude that the solution set for SSN is flat in the neighborhood of the optimal solution.

Table 10 Confidence Intervals for Function Values and Gaps for SSN at Three Points

Quantity	Estimate (95% conf. int.)
$f(x^0)$	$9.9016607 \pm .0035396$
$f(x^1)$	$9.9016731 \pm .0035396$
$f(x^2)$	$9.9043870 \pm .0035302$
$f(x^1) - f(x^0)$	$1.24118 \times 10^{-5} \pm 2.910 \times 10^{-7}$
$f(x^2) - f(x^0)$	$2.72648 \times 10^{-3} \pm 4.96724 \times 10^{-5}$

Table 11 Discrepancy in KKT Conditions for 20term Using Test of Section 5.3

M	$\hat{\delta}_N^M(\hat{x})$	λ
50	1065	$(-111.9,0,0)$
500	53.08	$(-100, -7 \times 10^{-12}, 0)$
1000	10.89	$(-100, -1 \times 10^{-11}, 0)$
2000	3.53	$(-100, -8 \times 10^{-14}, 0)$

9.4. *KKT tests for near-optimality*

We now report on the KKT tests described in Sections 5.3 and 5.4, applied to the problems 20term and SSN.

For 20term, we set $\hat{x}$ to be a computed solution of an SAA with $N = 100000$. The two first-stage equality constraints were active at this point, of course, but the single inequality constraint was inactive. We calculated approximations to the distances $\hat{\delta}_N^M(\hat{x})$ defined in (38) using an ℓ_1 norm, since this allowed us to formulate the distance calculation as a linear program, which we solved with a simple AMPL model (Fourer, Gay, and Kernighan, 2003). For the test of Section 5.3, we generated each of the M subgradients $\hat{\gamma}_i$, $i = 1, 2, \ldots, M$, at points x in a neighborhood of radius 10^{-2} around $\hat{x}$. We used MC sampling with $N = 100000$ to generate the subgradients, using a different sample for each point.

Results are shown for different values of M in Table 11. The final column contains the values of λ, the multiplier vector for the active constraints at the nearest point in $N_X(\hat{x})$ to the approximate subdifferential of f. The third component of λ is uniformly zero since the third constraint is inactive, while the first two components are clearly converging to -100 and 0, respectively. The distance to the subdifferential approximation appears to converge toward zero as more gradients are gathered, as we would expect, since the convex hull covers the subdifferential more fully as the number of gradients increases.

Although useful in verifying near-optimality, it is clear that this test is not a practical termination test for an algorithm, because the time required to compute enough gradients to verify optimality is too great.

For SSN, we performed the test described in Section 5.4 to the point x^0 discussed in Section 9.3. A subgradient was generated at each of the $T = 2000$ SAAs used in the experiment of Section 9.3, where each SAA has $\bar{N} = 25000$ and is obtained by LHS. The single inequality constraint in (40) is active at this solution, as are a number of the lower bounds of zero on the components of x. We performed this test using different values M for the size of the subgradient bundle. In each case we chose simply the first M of the 2000 subgradients in the list.

Results are shown in Table 12. The second column indicates the distance between the convex hull of the subgradient bundle and the normal cone of the constraints, and the third

Table 12 Discrepancy in KKT Conditions for SSN Using Test of Section 5.4

M	$\hat{\delta}_N^M(\hat{x})$	λ
5	.254847	.050000
20	.250329	.050000
250	.238522	.049774
1000	.237175	.049733
2000	.236541	.049878

column indicates the Lagrange multiplier for the inequality constraint $e^T x \leq b$ in (40). We see that the distance does not shrink to zero as M increases, indicating either that the point x^0 is some distance from optimality, or that the convex hull of the subgradient bundle is not converging to $\partial f(x^0)$. We conjecture that the latter is the case, since we have good evidence from the other tests that x^0 is close to optimal, and we know in general that Hausdorff convergence of the subgradient bundle to $\partial f(x^0)$ need not occur. Our experience with this test indicates that it is not valuable.

10. Conclusions

We have described experiments with sample-average approximations to five two-stage stochastic linear programs with recourse. We obtained confidence intervals for upper and lower bounds on the optimal value of these five problems, by solving several SAA instances for different sample sizes, using two different sampling techniques. We then investigated the optimality of various candidate solutions using a number of tests motivated by recently developed theory, including tests that are applicable to problems with a singleton solution set and tests based on satisfaction of the Karush-Kuhn-Tucker conditions for optimality. We found that the five test problem varied widely in difficulty, in that good approximations to the true optimal solutions could be obtained with small sample sizes in some cases, while larger sample sizes were needed in others.

The main finding of this study is that in all examined problems it was possible to compute a good solution, with a proven accuracy, of the true problem with a reasonable computational effort. This is especially notable since the last two problems (ssn and storm) were large, both in the number of random variables and the number of scenarios. As it is predicted by the theory, ill conditioned problems are more difficult to solve. For such problems a larger sample is required and the bias problem of the lower statistical bound is more serious. In that respect we found that the ssn problem is ill conditioned with a large set of optimal or nearly optimal solutions. Yet with a sample size of $N = 1000$ it was possible to solve this problem with a proven accuracy of about 0.5% of the relative error. It was also interesting to observe that even very large problems can be well conditioned and that a small sample is sufficient to solve such problems exactly.

All these experiments were made possible by the availability of the ubiquitous computing power of the computational grid, and by software tools such as Condor and MW that enable us to run practical computations on the grid. All these ingredients, along with an algorithm that runs effectively on the grid, were necessary in building the software tool ATR (described comprehensively in Linderoth and Wright (2003), which was needed to solve the multitude of instances required by this study.

References

Anstreicher, K., N. Brixius, J.-P. Goux, and J.T. Linderoth. (2002). "Solving Large Quadratic Assignment Problems on Computational Grids." *Mathematical Programming* 91(3), 563–588.

Avramidis, A.N. and J.R. Wilson. (1996). "Integrated Variance Reduction Strategies for Simulation." *Operations Research* 44, 327–346.

Bailey, T.G., P. Jensen, and D.P. Morton. (1999). "Response Surface Analysis of Two-Stage Stochastic Linear Programming With Recourse." *Naval Research Logistics* 46, 753–778.

Chen, Q., M.C. Ferris, and J.T. Linderoth. (2001). "Fatcop 2.0: Advanced Features in an Opportunistic Mixed Integer Programming Solver." *Annals of Operations Research* 103, 17–32.

Dantzig, G. and G. Infanger. (1991). “Large-Scale Stochastic Linear Programs—Importance Sampling and Benders’ Decomposition.” In C. Brezinski, and U. Kulisch (eds.), *Computational and Applied Mathematics I (Dublin, 1991)*, pp. 111–120. North-Holland, Amsterdam.

Dantzig, G.B. (1963). *Linear Programming and Extensions.* Princeton University Press, Princeton. New Jersey.

Diwekar, U.M. and J.R. Kalagnanam. (1997). “An Efficient Sampling Technique for Optimization Under Uncertainty.” *American Institute of Chemical Engineers Journal*, 43.

Ermoliev, Y. (1988). “Stochastic Quasigradient Methods.” In Y. Ermoliev, and R.J.-B. Wets (eds.), *Numerical techniques for stochastic optimization problems.* Springer-Verlag, Berlin.

Foster, I. and C. Kesselman. (1999). *The Grid: Blueprint for a New Computing Infrastructure.* Morgan-Kaufmann. Chapter 1 “Grids in Context” by Larry Smarr and chapter 13 “High-Throughput Resource Management” by Miron Livny and Rajesh Raman.

Fourer, R., D.M. Gay, and B.W. Kernighan. (2003). *AMPL: A Modeling Language for Mathematical Programming.* Thomson Learning, Brooks/Cole, Duxbury Press, Pacific Grove. CA. second edition.

Goux, J.-P., S. Kulkarni, J.T. Linderoth, and M. Yoder. (2000). “An Enabling Framework for Master-Worker Applications on the Computational Grid.” In *Proceedings of the Ninth IEEE Symposium on High Performance Distributed Computing (HPDC9)*, pp. 43–50. Pittsburgh, Pennsylvania.

Goux, J.-P., S. Kulkarni, J.T. Linderoth, and M. Yoder. (2001). “Master-Worker: An Enabling Framework for Master-Worker Applications on the Computational Grid.” *Cluster Computing* 4, 63–70.

Goux, J.-P. and S. Leyffer. (2002). “Solving Large MINLPs on Computational Grids.” *Optimization and Engineering* 3, 327–346.

Gürkan, G., A.Y. Özge, and S.M. Robinson. (1994). “Sample-Path Optimization in Simulation.” In *Proceedings of the Winter Simulation Conference*, pp. 247–254.

Higle, J.L. (1998). “Variance Reduction and Objective Function Evaluation in Stochastic Linear Programs.” *INFORMS Journal on Computing* 10(2), 236–247.

Higle, J.L. and S. Sen. (1991). “Stochastic Decomposition: An Algorithm for Two-Stage Linear Programs with Recourse.” *Mathematics of Operations Research* 16, 650–669.

Higle, J.L. and S. Sen. (1996). “Duality and Statistical Tests of Optimality for Two-Stage Stochastic Programs.” *Mathematical Programming* 75, 257–275.

Higle, J.L. and S. Sen. (1996). *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming.* Kluwer Academic Publishers, Boston, MA.

Infanger, G. (1994). *Planning Under Uncertainty: Solving Large Scale Stochastic Linear Programs.* Boyd and Fraser Publishing Company.

loffe, A.D. and V.M. Tihomirov. (1979). *Theory of Extremal Problems.* North-Holland, Amsterdam.

Kleywegt, A., A. Shapiro, and T. Homem-de-Mello. (2001). “The Sample Average Approximation Method for Stochastic Discrete Optimization.” *SIAM Journal on Optimization* 12, 479–502.

Linderoth, J.T., A. Shapiro, and S.J. Wright. (2002). “Emipirical Behavior of Sampling Methods for Stochastic Programming.” Optimization Technical Report 02–01, Computer Sciences Department, University of Wisconsin-Madison, January 2002. Revised April, 2003.

Linderoth, J.T. and S.J. Wright. (2003). “Decomposition Algorithms for Stochastic Programming on a Computational Grid.” *Computational Optimization and Applications* 24, 207–250.

Livny, M., J. Basney, R. Raman, and T. Tannenbaum. (1997). “Mechanisms for High Throughput Computing.” *SPEEDUP*, 11. Available from http://www.cs.wisc.edu/condor/doc/htc_mech.ps.

Louveaux, F.V. and Y. Smeers. (1988). “Optimal Investments for Electricity Generation: A Stochastic Model and a Test Problem.” In Y. Ermoliev, and R.J.-B. Wets (eds.), *Numerical techniques for stochastic optimization problems*, pp. 445–452. Springer-Verlag, Berlin.

Mak, W.K., D.P. Morton, and R.K. Wood. (1999). “Monte Carlo Bounding Techniques for Determining Solution Quality in Stochastic Programs.” *Operations Research Letters* 24, 47–56.

McKay, M.D., R.J. Beckman, and W.J. Conover. (1979). “A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code.” *Technometrics* 21, 239–245.

Mulvey, J.M. and A. Ruszczyński. (1995). “A New Scenario Decomposition Method for Large Scale Stochastic Optimization.” *Operations Research* 43, 477–490.

Norkin, V.I., G.Ch. Pflug, and A. Ruszczyński. (1998). “A Branch and Bound Method for Stochastic Global Optimization.” *Mathematical Programming* 83, 425–450

Owen, A.B. (1997). “Monte Carlo Variance of Scrambled Equidistribution Quadrature.” *SIAM Journal on Numerical Analysis* 34, 1884–1910.

Plambeck, E.L., B.R. Fu, S.M. Robinson, and R. Suri. (1996). “Sample-Path Optimization of Convex Stochastic Performance Functions.” *Mathematical Programming, Series B.*

Rockafellar, R.T. (1970). *Convex Analysis.* Princeton University Press, Princeton, N.J.

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

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

Sen, S., R.D. Doverspike, and S. Cosares. (1994). "Network Planning with Random Demand." *Telecommunications Systems* 3, 11–30.

Shapiro, A. (1991). "Asymptotic Analysis of Stochastic Programs." *Annals of Operations Research* 30, 169–186.

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

Shapiro, A. and T. Homem-de-Mello. (2000). "On the Rate of Convergence of Optimal Solutions of Monte Carlo Approximations of Stochastic Programs." *SIAM Journal on Optimization* 11, 70–86.

Shapiro, A., T. Homem-de-Mello, and J. Kim. (2002). "Conditioning of Convex Piecewise Linear Stochastic Programs." *Mathematical Programming, Series A* 94(1), 1–19.

Stein, M. (1987). "Large Sample Properties of Simulations using Latin Hypercube Sampling." *Technometrics* 29, 143–151.

Zakeri, G. (2000). *Verifying a stochastic solution using metacomputing.* Presentation at INFORMS Winter Meeting. Salt Lake City, USA.