Annals of Operations Research 64(1996)143–178

Second-order scenario approximation and refinement in optimization under uncertainty*

N.C.P. Edirisinghe and G-M. You
Management Science Program, University of Tennessee, Knoxville, TN 37996, USA

When solving scenario-based stochastic programming problems, it is imperative that the employed solution methodology be based on some form of problem decomposition: mathematical, stochastic, or scenario decomposition. In particular, the scenario decomposition resulting from scenario approximations has perhaps the least tendency to be computationally tedious due to increases in the number of scenarios. Scenario approximations discussed in this paper utilize the second-moment information of the given scenarios to iteratively construct a (relatively) small number of representative scenarios that are used to derive bounding approximations on the stochastic program. While the sizes of these approximations grow only linearly in the number of random parameters, their refinement is performed by exploiting the behavior of the value function in the most effective manner. The implementation SMART discussed here demonstrates the aptness of the scheme for solving two-stage stochastic programs described with a large number of scenarios.

Keywords: Bounds on stochastic programs, second-order scenario approximation, simplicial partitioning of joint domains.

1 Introduction

Incorporating uncertainty into mathematical optimization models for decision making is becoming increasingly popular due to at least two reasons: development of novel theoretical and algorithmic procedures for solving such models efficiently; an increasing number of real-world applications that demonstrate the superiority of stochastic solutions over their deterministic counterparts. A recent stochastic programming application for planning the finances of a major property and casualty insurer is discussed in Carinó et al. [4]. In such planning problems, typically, the future

*This paper was presented at the IFIP Workshop on *Stochastic Programming: Algorithms and Models*, Lillehammer, Norway, January 1994.

© J.C. Baltzer AG, Science Publishers

uncertainty is described using a set of outcome scenarios that the decision-maker deems appropriate. However, as the number of scenarios under consideration becomes very large, which is the case usually in practice, the resulting stochastic programming model may soon be pushed outside our current solution capability.

The numerical procedures in stochastic programming may be categorized in at least three ways: mathematical decomposition, sampling-based techniques, and bounds-based-approximation methods. The L-shaped decomposition (Van Slyke and Wets [32]) and its nested application for multistage problems (Birge [1], Gassmann [21]); and the scenario-aggregation techniques such as the progressive hedging algorithm due to Rockafellar and Wets [31] may be viewed as mathematical decomposition approaches. For parallel implementations, see Birge et al. [2] for instance. In sampling-based methods, one iteratively draws random samples from the underlying scenario-space for computing stochastic quasi-gradients (see Ermoliev [11], Ermoliev and Gaivoronsky [12]) or for stochastic L-shaped decomposition (Higle and Sen [23]), procedures which enjoy asymptotic convergence properties. Noting that the variance of the sample is the key to the convergence process, various variance reduction schemes, such as importance sampling within the L-shaped method (Dantzig and Glynn [5], Infanger [25]), have been employed for better algorithmic performance. In contrast to the above approaches, this paper is concerned with the third solution technique, namely, the successive approximation procedure based on computable bounds on the optimal objective value, see Ermoliev and Wets [13] for details.

In developing approximations, typically, one focuses on the value function ϕ arising in stochastic programming, being evaluated for the decisions x taken so far and the random variables ω observed so far, hence written as $\phi(x, \omega)$. In an equivalent two-stage decision problem, consequently, the stochastic program is formulated as (see Kall [26]):

$$Z^* := \min_{x \in X} \left\{ c'x + \sum_{s \in S} p_s \phi(x, \omega^s) \right\}, \tag{1}$$

where the scenarios $\omega^s \in \mathbb{R}^M$, with associated probabilities p_s, are indexed in the set S. We use Ω to denote a set in $\mathbb{R}^M$ that contains the set of given scenarios $\{\omega^s : s \in S\}$. The feasible set for choosing x is denoted by $X \subset \mathbb{R}^{n_1}$. The value function ϕ is determined by solving a second optimization problem for *recourse* decisions $y^s \in \mathbb{R}^{n_2}$ under each scenario ω^s, for a chosen first-stage decision x. Therefore, if there are many scenarios under consideration, i.e. $|S|$ is large, then (1) becomes simply too large to solve in a *straightforward* manner, i.e. by casting the problem in the so-called deterministic extensive-form. To alleviate this difficulty, one may resort to approximating either the scenario distribution (ω^s, p_s) along with its domain Ω or the value function ϕ, with the aim of obtaining accurate bounding functions:

$$\psi_L(x) \le \psi(x) := \sum_{s \in S} p_s \phi(x, \omega^s) \le \psi_U(x), \tag{2}$$

such that computing the lower and upper approximations $\psi_L(x)$ and $\psi_U(x)$, respectively, is relatively easy compared to computing $\psi(x)$. Furthermore, the bounds in (2) are required to be tight, for instance, in the sense of a certain generalized moment problem (e.g. see [8,9]), or at the very least, the approximations are required to coincide with $\psi(x)$, i.e. hence exact, if ϕ is linear (or bi-linear) in ω.

While the computational simplicity depends on size as well as exploitable structure available in the approximations $\psi_L(x)$ and $\psi_U(x)$, tightness of the bounds is dependent on such factors as the extent to which functional information of ϕ and probabilistic information of the underlying scenario space could be utilized in constructing the approximations in (2).

The two main functional properties that have been exploited for constructing approximations are the convexity of ϕ in ω or the convex–concave saddle property of ϕ in ω. These properties naturally arise in stochastic linear programming, when one considers the following second-stage problem for determining recourse decisions $y^s \in \mathbb{R}^{n_2}$ (under the scenario index $s \in S$):

$$\phi(x, \xi^s, \eta^s) := \inf_{y^s} \{q(\eta^s)' y^s : Wy^s = h(\xi^s) - T(\xi^s)x, y^s \geq 0\}, \tag{3}$$

where the random vector ω represents the right-hand side random components $\xi \in \Xi \subset \mathbb{R}^K$ and the objective random components $\eta \in \Theta \subset \mathbb{R}^L$, i.e. $\omega = (\xi, \eta)$. In particular, $\Omega \subseteq \Xi \times \Theta$ and all components of ω may be correlated. Moreover, T, h, and q are affinely generated according to:

$$T(\xi) = T_0 + \sum_{k=1}^{K} T_k \xi_k, \quad h(\xi) = H\xi + h_0, \quad \text{and} \quad q(\eta) = Q\eta + q_0, \tag{4}$$

where $H \in \mathbb{R}^{m_2 \times K}$, $Q \in \mathbb{R}^{n_2 \times L}$, and $T_0, \ldots, T_K \in \mathbb{R}^{m_2 \times n_1}$ are fixed matrices, while $h_0 \in \mathbb{R}^{m_2}$ and $q_0 \in \mathbb{R}^{n_2}$ are fixed vectors. Therefore, $\phi(x, \cdot, \cdot)$ is a convex–concave saddle (polyhedral) function for each $x \in X$. In the absence of uncertainty in the objective coefficients of (3), ϕ is a convex function in ξ. Also, by assuming a complete recourse matrix W (see, Kall [26]) and that the set $\{\pi : \pi W \leq q(\eta)\}$ is non-empty, we obtain ϕ as a *proper* function.

Under the convex case, a standard strategy is to use linear (or piecewise-linear) functions to approximate the convex ϕ from above and below. This yields the Jensen lower bound and the Edmundson–Madansky [29] upper bound, which use the first moment information of the given scenarios. While the latter upper bound in multi-dimensions is only applicable with rectangular supports under stochastic independence of random components, the extension to the dependent case under general polyhedral domains is given in Gassmann and Ziemba [22]. Incorporating joint moments, in addition to first moments, Frauendorfer [14[derived an upper bound under multi-dimensional rectangles.

If we define first-order information of the underlying scenario space as those moment conditions in which each random variable is raised at most to the power of one, then all of the above-mentioned bounds can be categorized as *first-order approximations* of scenarios. Let us similarly define *second-order approximations* of scenarios as those utilizing moment information in which each random variable is raised at most to the power of two, and at least one such moment condition is used in computing bounds. Dula's [6] upper bound is an example of second-order approximations in the convex case.

Considering the saddle case, which indeed is the more general case, the usual strategy is to compute lower and upper approximating functions that are bi-linear in ξ and η, see [8,9,15,16]. According to the foregoing definition, the bounds developed in the latter references may be classified as first-order scenario approximations. Recently, new second-order approximations were proposed by Edirisinghe [10]. This paper is concerned mostly with details of second-order scenario approximations and their implementation for solving problems with a large number of scenarios. It develops that the second-order approximations are not only tighter than the first-order scenario approximations, but they also lead to more effective procedures for further refinement of the approximations. Moreover, these second-order approximations are determined according to a unique representation of the underlying scenario set, thereby imparting a naturally decomposable structure to the approximating problems which may thus be solved efficiently under parallel computation.

The paper is organized as follows. Section 2 includes a review of the second-order approximations and a comparison with first-order approximations with regard to computational simplicity and tightness. Also included is a discussion on the iterative use of the approximations for solving stochastic programs and its convergence. Section 3 discusses the basic framework of the iterative approximation and its detail. In particular, various refining strategies are described. The implementation of the procedure is in section 4, while the computational experience with this implementation is in section 5. Section 6 concludes with some remarks. The required notation will be introduced as it becomes necessary.

2 Scenario approximations and clustering

Consider the stochastic program given in (1) along with the second-stage recourse function definition (3) in which uncertainty is described by the set of scenarios $\{\omega^s := (\xi^s, \eta^s): s \in S\}$ for the $(K+L)$-dimensional random vector $\omega = (\xi, \eta)$. Scenario approximation is usually accomplished under

(i) a certain geometric shape for either domain Ω, or separately for Ξ and Θ, covering the scenarios $\{\omega^s: s \in S\}$, and

(ii) a set of moment information computed for the scenarios $\{\omega^s: s \in S\}$.

Usually, the shape of Ω is intimately related to the amount of moment information that can be utilized to construct upper and lower approximations. Accordingly, the resulting bounds can be classified as either first-order or second-order scenario approximations, see section 1 for the definitions.

2.1 *First-order approximations*

The two main types of approximations in this category are those due to Edirisinghe and Ziemba [8,9] and Frauendorfer [15,16]. These approximations are based on utilizing marginal domains Ξ and Θ (of the scenarios) and then constructing scenario distributions – one for lower and one for upper bound – involving the extreme points of the marginal domains.

Consider the domains Ξ and Θ to be compact, being finitely generated by the sets of extreme points $\{u^i : i \in I\}$ and $\{v^j : j \in J\}$, respectively.

When Ω is a multi-dimensional (compact) rectangle, thus Ξ and Θ are rectangular, bounds in [15] use the first-order joint moment information:

$$\sum_{s\in S} p_s[\eta_l^s]^\theta \left[\prod_{r\in\Gamma_1} \xi_r^s\right] \quad \text{and} \quad \sum_{s\in S} p_s[\xi_k^s]^\theta \left[\prod_{r\in\Gamma_2} \eta_r^s\right] \tag{5}$$

for $\theta \in \{0, 1\}$, $l \in \{1,\ldots,L\}$, $k \in \{1,\ldots,K\}$, $\Gamma_1 \in B_K$, and $\Gamma_2 \in B_L$, where B_K is the set of all subsets of $\{1,\ldots,K\}$ and B_L is the set of all subsets of $\{1,\ldots,L\}$, to determine upper and lower approximating scenario distributions leading to inequalities of the form (with probabilities denoted by $\overline{\alpha}$ and $\underline{\alpha}$, respectively):

$$\psi(x) \le \sum_{i=1}^{I} \overline{\alpha}_i \phi(x, u^i, \hat{\eta}^i) \quad \text{and} \quad \psi(x) \ge \sum_{j=1}^{J} \underline{\alpha}_j \phi(x, , \hat{\xi}^j, v^j), \tag{6}$$

where $I = 2^K$ and $J = 2^L$. It should be observed in the above that the approximating distributions are independent of the decision process x and the recourse function ϕ, and thus, when used in the stochastic program (1), the inequalities in (6) yield approximating linear programs with the so-called L-shaped decomposition structure. However, when the dimension of uncertainty is large, i.e. K and L are large, the approximating distributions themselves could contain a large number of scenarios – exponential in K and L – to be considered, which can render these approximations computationally prohibitive.

A possible remedy for this situation – in reducing the number of approximating scenarios that one has to deal with – is perhaps the approximations in [8], wherein the domains Ξ and Θ may be considered to be polytopes (of a smaller number of vertices), hence Ω itself is a polytope. But now, the first-order moment information that one can utilize is weakened to the following first and cross moments:

$$\sum_{s \in S} p_s [\xi_k^s]^{\theta_1} [\eta_l^s]^{\theta_2} \quad \text{for} \quad \theta_1, \theta_2 \in \{0, 1\}. \tag{7}$$

Furthermore, the resulting approximations no longer possess, in general, the properties of being independent of the decisions x and the function ϕ, i.e. approximating scenarios as well as probabilities β depend on x and ϕ:

$$\psi(x) \leq \sum_{i=1}^{I} \overline{\beta}_i(x) \phi(x, u^i, \hat{\eta}^i(\overline{\beta})) \quad \text{and} \quad \psi(x) \geq \sum_{j=1}^{J} \underline{\beta}_j(x) \phi(x, , \hat{\xi}^j(\underline{\beta}), v^j). \tag{8}$$

Nevertheless, the above inequalities have been shown to generate lower and upper bounding functions which are piecewise linear convex, thus the resulting approximations on (1) would necessitate at most solution of linear programs. Moreover, this lower approximating linear program would grow only linearly, i.e. $O(L)$, in the constraints. Edirisinghe and Ziemba [10] showed how the latter approximations may be forced to have the desirable L-shaped structure for efficient solution whilst enjoying the flexibility in choosing any polyhedral shape for domains. The implementation RAWFC (Rectangular Approximation With First and Cross moments) discussed in the latter reference uses rectangular (polyhedral) domains to solve large scenario-based stochastic programs when the dimension of uncertainty is sufficiently small. If, on the other hand, the domains Ξ and Θ are chosen to be multi-dimensional simplices – thus, Ω is a $\times$-simplex – then the approximations in (8) simplify to the format in (6), but now with $I = K + 1$ and $J = L + 1$. These are the bounds developed in [16] and implemented in [17].

The extensions of the above results when the scenarios are contained in unbounded polyhedral domains appear in [9].

Towards sharpening the approximations (under the above bounding techniques), the usual strategy is to partition the domain (containing the scenarios) into non-overlapping cells (of the chosen geometric shape), and re-apply the bounding technique in each cell of the partition. Major issues that need to be resolved in this exercise are:

(i) how should a cell be chosen for further partitioning?, and

(ii) how should a chosen cell be further subdivided?

A cell (or cells) for further subdivision may be chosen based on a criterion which measures the probabilistically-weighted relative gap of lower and upper approximating objective values for that cell, see [10] for instance. Regarding the second issue, one major task is determining directions along which further partitioning of the chosen cell can be carried out. The standard practice is to use the notion of *degree of nonlinearity* of the recourse function along edges of the domain and to partition along directions which are perpendicular to such edges, see Frauendorfer and Kall [20]. Such practice is motivated by the fact that the aforementioned approximations are exact if the recourse function is linear in its domain. Nevertheless, since the first-

order approximations discussed so far are based on Ξ and Θ, a refinement process can only sub-divide the marginal domains, rather than the joint domain Ω. As a consequence, nonlinearity along joint-directions in Ω cannot be exploited in refining the approximations.

A possible fix for this shortcoming might be to use joint-partitioning directions directly in Ω, and then project the resulting partitioned cells on to Ξ and Θ domains in order to continue using first-order approximations. For instance, Ω may be considered to be a (multi-dimensional) simplex and then under joint-partitioning of this simplex, the projected cells on Ξ and Θ spaces become polytopes with at most $(K + L + 1)$ vertices, see figure 1 for an illustration. Consequently, on the resulting

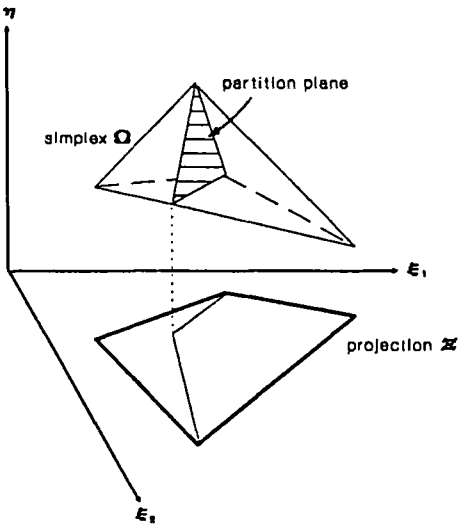

Figure 1. Projection of simplicial partitioning.

polytopes, first/cross moment bounds in [8] can be applied. This practice allows one to solve problems with larger dimensions of uncertainty since the sizes of these first-order approximations will now grow only linearly with partitioning. Such an implementation is considered in the computational section of this paper, referred to as SPWFC (Simplicial Projection With First and Cross moments).

However, the price one pays for this flexibility in partition-directions is the increase in the *volume* of the (simplicial) domain Ω containing the scenarios, which generally leads to weakening the tightness of the bounds on the objective value. Therefore, a natural strategy under such simplicial joint domains is to seek tighter approximations using higher-order moment information. The second-order approximations derived in Edirisinghe [7] are most appealing in that both second-order moment information as well as joint-partitioning in simplicial domains are admissible. We shall discuss them

in the following section, with particular attention to the refinement and convergence details. These approximations will be implemented later in this paper for solving large scenario-based stochastic programs.

2.2 *Second-order approximations*

Consider the joint domain Ω to be a $(K+L)$-dimensional simplex, having extreme points $w^i \equiv (u^i, v^i)$, $i = 1,\ldots,I := K+L+1$. Define the $I \times I$-dimensional matrix V by

$$V := \begin{bmatrix} w_1^1 & w_1^2 & \cdots & w_1^I \\ w_2^1 & w_2^2 & \cdots & w_2^I \\ \vdots & & & \vdots \\ w_{K+L}^1 & w_{K+L}^2 & \cdots & w_{K+L}^I \\ 1 & 1 & \cdots & 1 \end{bmatrix}. \tag{9}$$

Since Ω is a $(K+L)$-dimensional simplex, V is a nonsingular matrix, the inverse of which is denoted by V^{-1}. Define $\rho \in \mathbb{R}^I$ by

$$\rho := V^{-1} \begin{pmatrix} \sum_{s\in S} p_s \omega^s \\ 1 \end{pmatrix}, \tag{10}$$

$\hat{\rho}^l \in \mathbb{R}^I$ for each $l = 1,\ldots,L$ by

$$\hat{\rho}^l := V^{-1} \begin{pmatrix} \sum_{s\in S} p_s \eta_l^s \omega^s \\ \sum_{s\in S} p_s \eta_l^s \end{pmatrix}, \tag{11}$$

and $\tilde{\rho}^k \in \mathbb{R}^I$ for each $k = 1,\ldots,K$ by

$$\tilde{\rho}^k := V^{-1} \begin{pmatrix} \sum_{s\in S} p_s \xi_k^s \omega^s \\ \sum_{s\in S} p_s \xi_k^s \end{pmatrix}. \tag{12}$$

Then, the following bounding approximations hold:

Theorem 2.1 [7]

$$Z^* \le \min_{x\in X} \left\{ c'x + \sum_{i=1}^{K+L+1} \rho_i \phi(x, \hat{\omega}^i) \right\} \tag{13}$$

and

$$Z^* \ge \min_{x\in X} \left\{ c'x + \sum_{i=1}^{K+L+1} \rho_i \phi(x, \tilde{\omega}^i) \right\} \tag{14}$$

for $\hat{\omega}^i := (u^i, \hat{\eta}^i)$ and $\tilde{\omega}^i := (\tilde{\xi}^i, v^i)$, where $\hat{\eta}_l^i := \hat{\rho}_i^l / \rho_i$, $l = 1,\ldots,L$, and $\tilde{\xi}_k^i := \tilde{\rho}_i^k / \rho_i$, $k = 1,\ldots,K$.

Since $\hat{p}^l$ and $\tilde{p}^k$ in (11) and (12), respectively, use second moments of the given scenarios, the bounding results in theorem 2.1 are second-order approximations. It is easy to show that the lower approximating scenario distribution $\{(\tilde{\omega}^i, \rho_i)\colon i = 1,\ldots,K + L + 1\}$ and the upper approximating scenario distribution $\{(\hat{\omega}^i, \rho_i)\colon i = 1,\ldots,K + L + 1\}$ have the same first and cross moments as the original scenario distribution $\{(w^s, p_s)\colon s \in S\}$. However, simple numerical examples can be constructed to show that these approximating distributions may not have the same second moments as the original one. Consequently, in general, these second-order approximations may not represent solutions to the underlying *generalized moment problems*; see example 1 and the discussion in [7]. For an illustration of the bounds in theorem 2.1 for $K = L = 1$, see figure 2, where we have denoted $\zeta^i := (\tilde{\xi}^i, \hat{\eta}^i)$ for $i = 1,\ldots,K + L + 1$. Also notice that $\sum_{i=1}^{K+L+1} \rho_i \zeta^i = \bar{\omega} := \sum_{s \in S} p_s \omega^s$.

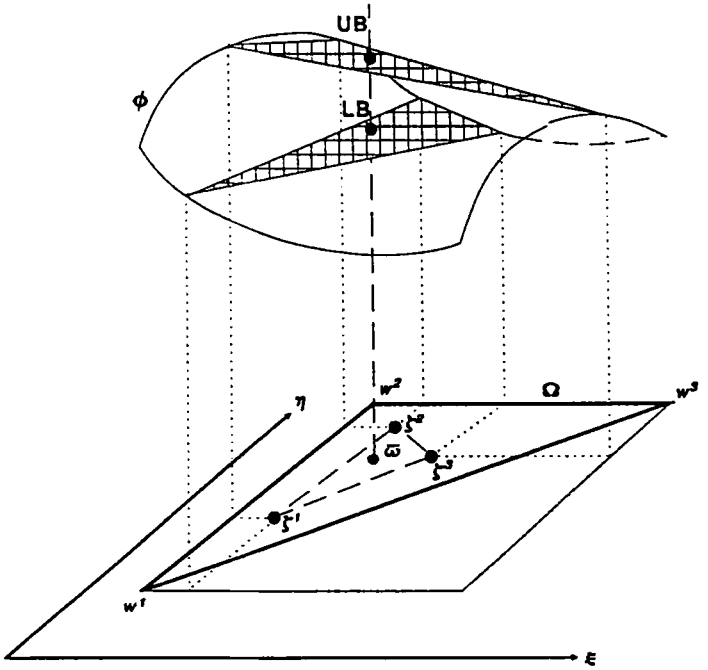

Figure 2. Second-moment bound for the expected recourse function.

The approximations in (13) and (14) possess the required dual block-angular structure with at most $(K + L + 1)$ blocks, in comparison to $|S|$ blocks in the exact formulation (1). $|S|$ denotes the cardinality of the scenario index-set S. Moreover, since $|S|$ is much larger than $K + L + 1$ – in practice, $|S|$ is at least 2^{K+L} – the approximations in theorem 2.1 are computationally much simpler than solving (1) directly.

2.3 *Partitioning and convergence*

For refining the second-order simplicial approximation, when the domain Ω is partitioned, the ensuing cells of the partition are required to be simplices. Details of performing such a simplicial partition is discussed in section 3.

Let $S^\nu := \{\Omega_r | \, r = 1,\ldots,R_\nu\}$ represent such a partition (at some partitioning iteration ν) of the joint simplicial domain Ω into subsimplices Ω_r such that

$$\bigcup_{r=1}^{R_\nu} \Omega_r = \Omega, \quad \Omega_r \bigcap \Omega_{r'} = \varnothing, \quad \forall r \neq r'; r, r' \in \{1,\ldots,R_\nu\}. \tag{15}$$

Let the corresponding index-sets of scenarios be $S_1,\ldots,S_{R_\nu}$, where $S_r := \{s | \, \omega^s \in \Omega_r, s \in S\}$. Observe that the dependence of S_r on the iteration count ν is suppressed for the convenience of exposition. The corresponding *cell probabilities* are evaluated as $P_r := \sum_{s \in S_r} p_s$, where $\sum_{r=1}^{R_\nu} P_r = 1$. Under the cell-conditional first and second moments computed by

$$\frac{1}{P_r} \sum_{s \in S_r} p_s \omega_k^s [\omega_l^s]^\theta, \quad \theta = 0,1; \; k,l = 1,\ldots,K+L, \tag{6}$$

one obtains the upper and lower approximating (second-order) scenario distributions for cell Ω_r, using a procedure similar to (9)–(14), $\{(\hat{\omega}^i(r), \rho_i(r)) | \, i = 1,\ldots,K+L+1\}$ and $\{(\tilde{\omega}^i(r), \rho_i(r)) | \, i = 1,\ldots,K+L+1\}$, respectively. The monotonicity of the approximations under this partitioning procedure is given by the result below, which is straightforward to prove.

Theorem 2.2

Under the above simplicial partitioning procedure of the joint domain Ω, the second-order bounds behave monotonically for all $x \in X$, i.e.

$$\psi(x) \leq \psi_U^\nu(x) := \sum_{r=1}^{R_\nu} P_r \phi_U^{\nu,r}(x) \leq \sum_{i=1}^{K+L+1} \rho_i \phi(x, \hat{\omega}^i) \tag{17}$$

and

$$\psi(x) \geq \psi_L^\nu(x) := \sum_{r=1}^{R_\nu} P_r \phi_L^{\nu,r}(x) \geq \sum_{i=1}^{K+L+1} \rho_i \phi(x, \tilde{\omega}^i), \tag{18}$$

where

$$\phi_L^{\nu,r}(x) := \sum_{i=1}^{K+L+1} \rho_i(r) \phi(x, \tilde{\omega}^i(r)) \quad \text{and} \quad \phi_U^{\nu,r}(x) := \sum_{i=1}^{K+L+1} \rho_i(r) \phi(x, \hat{\omega}^i(r)). \tag{19}$$

Corresponding to the above sequence of partitioning, one may obtain the sequence of upper and lower approximating objective value-sequences $\{Z_U^\nu\}$ and $\{Z_L^\nu\}$ by solving the linear programs

$$Z_U^\nu := \min_{x \in X}\{c'x + \psi_U^\nu(x)\} \quad \text{and} \quad Z_L^\nu := \min_{x \in X}\{c'x + \psi_L^\nu(x)\}, \tag{20}$$

along with their optimal first-stage solutions x_U^ν and x_L^ν, respectively. Due to theorem 2.2, the sequence $\{Z_U^\nu\}$ is monotonically decreasing while $\{Z_L^\nu\}$ is monotonically increasing. Furthermore, as long as $\max_{r=1,\ldots,R_\nu} P_r \to 0$ when $\nu \to \infty$, the approximating scenario distributions $\rho(r)|_{r=1}^{R_\nu}$ can be shown to converge *weakly* to the original distribution $p_s|_{s=1}^{S}$. This yields $\lim_{\nu\to\infty} Z_U^\nu = Z^*$ and $\lim_{\nu\to\infty} Z_L^\nu = Z^*$.

The question of convergence of the approximate x-solutions to true optimal solutions of (1) can be answered by appealing to the theory of *epiconvergence*. Loosely speaking, epiconvergence here refers to the convergence of the epigraphical sets of the approximating functions (convex in x, in our case) to the epigraph of the original recourse function $\psi(\cdot)$. With weak convergence of the approximating probability measures along with the joint domain Ω being a (compact) simplex, due to Birge and Wets [3, section 2.11], one has the required epiconvergence of the approximating functions. Under epiconvergence, theorem 9 of Wets [33] shows that the set of approximate solutions belongs to the set of optimal solutions, i.e.

$$\limsup[\arg\min g^\nu] \subset \arg\min g,$$

where $g^\nu(x) := c'x + \psi_U^\nu(x)$ (respectively, $g^\nu(x) := c'x + \psi_L^\nu(x)$) and $g(x) := c'x + \psi(x)$. Hence, $\{x_U^\nu\} \to x^*$ and $\{x_L^\nu\} \to x^{**}$ hold as desired, where x^* and x^{**} are optimal solutions of the stochastic program (1).

There are at least two related issues: stability and speed of convergence. While it follows from the preceding results that the approximations may be refined to a user-specified accuracy through a partitioning procedure, that the solution sets so generated behave in a continuous manner with respect to small perturbations in the approximating measures can be ensured by appealing to the stability results due to Kall [27] and Robinson and Wets [30]. The second important issue is how fast can the convergence (or a given accuracy for the upper and lower approximations) be achieved. Obviously, this depends on many factors such as the initial simplex used to contain the given set of scenarios, the manner in which simplicial partition of Ω is performed, as well as how efficiently one can solve the second-order approximations. These are the topics under discussion in section 3.

A final remark is that solving the upper and lower approximations in (20) for separate first-stage solution vectors x_U^ν and x_L^ν does not help much in identifying cells (i.e. clusters of scenarios) of the current partition for further refinement. Observing that the lower approximating solution x_L^ν is feasible in the upper approximating problem, the standard strategy is to compute a weaker upper bound by

$$\bar{Z}_U^\nu := c'x_L^\nu + \psi_U^\nu(x_L^\nu). \tag{21}$$

Consequently, the accuracy of the second-order approximation for estimating the objective value Z^* of (1), at some iteration ν, can be evaluated on the basis of the current lower bounding solution x_L^ν by (provided $Z_L^\nu > 0$)

$$\left| \frac{\bar{Z}_U^\nu - Z_L^\nu}{Z_L^\nu} \right| < \varepsilon, \tag{22}$$

i.e. if the *relative gap* of the upper and lower bounds for the current solution x_L^ν is within a certain tolerance ε, then no further refinement of the current approximation is needed.

3 Simplicial partitioning for iterative approximation

To apply the simplicial approximation technique presented in the preceding section, an initial ($K + L$ dimensional) simplex which can contain the given set of scenarios is needed. Usually, in practice, scenarios are constructed by following random variable outcome-trees, see for instance Cariño et al. [4], and thus when the scenarios are viewed as points in the $(K + L)$-dimensional space, their convex hull C may be far from being simplicial. Thus, an initial simplex Ω covering this convex hull is generally expected to consist of a larger volume of *zero-measure* which typically contributes to weakening the approximations. However, when refining the approximations through a simplicial partition of this initial simplex Ω, one is able to iteratively remove the areas of zero-measure, a procedure later described as *cell-redefining*.

3.1 *Constructing an initial simplex Ω*

Since the *empty spaces* in Ω adversely affect the tightness of the approximations, hence the speed of convergence, it is desirable to have an initial simplex as compact as possible. While many nonlinear programming formulations can be developed for this purpose, when the scenario set is large in number, such an approach becomes an onerous task in itself. Alternatively, one might determine a compact rectangle $\mathcal{R}$ containing C by defining:

$$\alpha_j := \min_{s \in S}\{\omega_j^s\} \quad \text{and} \quad \beta_j := \max_{s \in S}\{\omega_j^s\}, \quad \forall j = 1, \ldots, K + L, \tag{23}$$

and setting $\mathcal{R} := \times_{j=1}^{K+L}[\alpha_j, \beta_j]$. Subsequently, a simplicial partition of this rectangle may be attempted since $C \subseteq \mathcal{R}$. However, this leads to increasing the number of simplices in the problem exponentially, and thus, may not be desirable from a computational point of view. Instead, we take a simplistic approach by adopting from Horst and Tuy [24] and modifying as follows: Let

$$\beta_0(\delta) := \max_{s \in S}\left\{ \sum_{j=1}^{K+L} \delta_j \omega_j^s \right\} \tag{24}$$

for some fixed coefficient vector $\delta \in \mathbb{R}^{K+L}$ such that $\delta_j > 0$, $j = 1,\ldots,K+L$. Then, $\Omega(\delta)$ defined by

$$\Omega(\delta) := \left\{ \omega \in \mathbb{R}^{K+L} : \sum_{j=1}^{K+L} \delta_j \omega_j - \beta_0(\delta) \leq 0, \omega_j - \alpha_j \geq 0, \forall j = 1,\ldots,K+L \right\} \quad (25)$$

is a simplex containing C. Furthermore, this simplex is characterized by the set of extreme points $\mathcal{V} := \{w^1, w^2, \ldots, w^{K+L+1}\}$, where the jth component of w^1 is $w_j^1 = \alpha_j$ for $j = 1,\ldots,K+L$ and the remaining vertices w^i (for $i = 2,\ldots,K+L+1$) are given by

$$w_j^i(\delta) = \begin{cases} \alpha_j & \text{if } i \neq j+1, \\ \dfrac{1}{\delta_j}\left[\beta_0(\delta) - \sum_{k=1,k\neq j}^{K+1} \delta_k \alpha_k\right] & \text{if } i = j+1. \end{cases} \quad (26)$$

As a result, the *degree of uncertainty* – defined as the maximum variability of the values that the random variables can take for a given domain – has increased when considering $\Omega(\delta)$ instead of C. While this leads to the effect of diluting the quality of approximations, the cell-redefining strategy discussed in the next section may be used to reduce the degree of uncertainty under a simplicial partition of $\Omega(\delta)$.

Nevertheless, the coefficient vector δ may be chosen to control the effect of the initial simplex on the approximations. For instance, the uncertainty in the objective coefficients typically has a lesser effect on the tightness of the approximations (as will be demonstrated in the computational section) than that in the right-hand side coefficients of the recourse linear program. This motivates one to construct the simplex $\Omega(\delta)$ such that the increase in the degree of uncertainty in the ξ components is smaller than that in the η components. This may be accomplished using the following simple result:

Proposition 3.1

For e_j denoting the jth elementary vector and $\delta \in \mathbb{R}^{K+L}$ being a given fixed coefficient vector such that $\delta > 0$, $w_j^{j+1}(\delta + \varepsilon e_j) \leq w_j^{j+1}(\delta)$ holds for every $\varepsilon > 0$.

Proof

Rewriting, $\beta_0(\delta) = \max_{s\in S}\{C_s + \delta_j \omega_j^s\}$, where we have defined $C_s := \sum_{k\neq j,k=1}^{K+L} \delta_k \omega_k^s$. Also define $G := \sum_{k\neq j,k=1}^{K+L} \delta_k \alpha_k$. Clearly, $G \leq C_s$ for all $s \in S$. Hence,

$$\begin{aligned} w_j^{j+1}(\delta) &= \max_{s\in S}\{C_s/\delta_j + \omega_j^s\} - G/\delta_j \\ &= \max_{s\in S}\{(C_s - G)/\delta_j + \omega_j^s\} \\ &\geq \max_{s\in S}\{(C_s - G)/(\delta_j + \varepsilon) + \omega_j^s\} = w_j^{j+1}(\delta + \varepsilon e_j) \end{aligned}$$

holds since $(C_s - G) \geq 0$, $\forall s \in S$. □

Accordingly, one would be motivated to assign smaller δ_j for coordinates along which the underlying function is close to being linear. However, since the recourse function ϕ depends on first-stage decisions x as well, implementation of this idea is somewhat difficult. In section 5, we discuss this sensitivity of δ on the tightness of the approximations.

3.2 *Cell-Redefining (C-R) procedure*

A C-R procedure was introduced and implemented in [10] in order to further tighten first-order bounds when partitioning in rectangular domains. In this section, we pursue a similar idea with regard to simplicial domains.

Let us consider a simplicial partition of the initial simplex Ω, as indicated in (15). In particular, at some iteration v of partitioning, for some $r \in \{1,\ldots,R_v\}$, the cell Ω_r is a simplex itself in $\mathbb{R}^{K+L}$. The set of scenarios in this cell is described by $\{\omega^s: s \in S_r\}$. Let $C_r := conv\{\omega^s: s \in S_r\}$, the convex hull.

Definition 3.2
The cell Ω_r is said to have *removable* volume of zero measure if there exists a simplex Ω'_r such that $C_r \subseteq \Omega'_r \subset \Omega_r$.

The C-R procedure strives for such a simplex Ω'_r (with the least possible volume) for a given cell Ω_r. The motivation is that the upper and lower approximations on Ω'_r will be much tighter than those on Ω_r. See figure 3 for an illustration in $\mathbb{R}^2$.

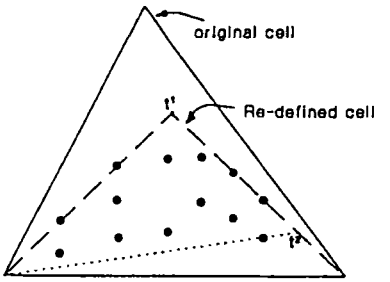

Figure 3. Cell-redefining procedure for a simplex.

For the convenience of exposition, let us drop the index r. The set of extreme points of Ω is $\{w^1,\ldots,w^{K+L+1}\}$ and the associated vertex matrix is denoted by V, see (9). The cell-redefining procedure selects a vertex w^j of Ω, checks if this vertex can be replaced (i.e. redefined) by a point $t \in \Omega$ such that the simplex

$$\Omega' := conv\{w^1,\ldots,w^{j-1},t,w^{j+1},\ldots,w^{K+L+1}\} \subset \Omega, \tag{27}$$

and if so, replaces Ω by Ω' provided that $\Omega \backslash \Omega'$ is of sufficiently large volume.

For each scenario ω^s ($s \in S$) in Ω, determine the *convex multipliers* $\lambda^s (\in \mathbb{R}^{K+L+1})$ by

$$\lambda^s := V^{-1} \begin{pmatrix} \omega^s \\ 1 \end{pmatrix} \tag{28}$$

and thus $0 \le \lambda_i^s \le 1$ and $\sum_{i=1}^{K+L+1} \lambda_i^s = 1$, $\forall s \in S$. The set of multipliers associated with vertex w^i is represented by $\Lambda_i := \{\lambda_i^s : s \in S\}$. Observe that $\mu_i := \max\{\Lambda_i\}$ is a *measure of closeness* of the scenarios (in the cell) to a given vertex w^i. If μ_i is close to one, then of course one would not expect a significant reduction in the volume of Ω when the vertex w^i is redefined. In fact, the *best* vertex for redefining, in this sense, is determined by the index

$$j := \arg\min\{\mu_i : i = 1, \ldots, K + L + 1\}. \tag{29}$$

Therefore, unless $\mu_j < \mu^*$, a user-specified threshold value (<1), the C-R procedure should not be invoked.

Suppose $\mu_j < \mu^*$, and thus, the vertex w^j is to be replaced by the $(K + L)$-dimensional point $t \in \Omega$ such that t is *farthest* as possible from w^j, in the sense of the associated multipliers, such that (27) holds. This problem may be formulated as:

$$\begin{aligned} \underset{t \in R^{K+L}}{\text{minimize}} \quad & \left[V^{-1} \begin{pmatrix} t \\ 1 \end{pmatrix} \right]_j \\ \text{subject to} \quad & V^{-1} \begin{pmatrix} t \\ 1 \end{pmatrix} \ge 0, \\ & \hat{V}^{-1} \begin{pmatrix} \omega^s \\ 1 \end{pmatrix} \ge 0, \quad \forall s \in S, \end{aligned} \tag{30}$$

where $\hat{V}$ is the matrix that results when the jth column of V is replaced by the column-vector $\binom{t}{1}$. The inverse of $\hat{V}$ is obtained by $\hat{V}^{-1} = UV^{-1}$, where U is the *eta-matrix* given by

$$U := \begin{bmatrix} 1 & 0 & \cdots & 0 & -\frac{\hat{t}_1}{\hat{t}_j} & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & -\frac{\hat{t}_2}{\hat{t}_j} & 0 & \cdots & 0 \\ \vdots & & & & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & \frac{1}{\hat{t}_j} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & -\frac{\hat{t}_{j+1}}{\hat{t}_j} & 1 & \cdots & 0 \\ \vdots & & & & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & -\frac{\hat{t}_{K+L+1}}{\hat{t}_j} & 0 & \cdots & 1 \end{bmatrix} \tag{31}$$

and

$$V^{-1}\begin{pmatrix} t \\ 1 \end{pmatrix} := \hat{t} \in \mathbb{R}^{K+L+1}. \tag{32}$$

Upon algebraic manipulation and simplification, (30) reduces to the following linear program:

$$\begin{aligned} &\underset{t \in \mathbb{R}^{K+L}}{\text{minimize}} \quad V^{-1(j)}\begin{pmatrix} t \\ 1 \end{pmatrix} \\ &\text{subject to} \quad V^{-1}\begin{pmatrix} t \\ 1 \end{pmatrix} \geq 0, \\ &\qquad (\lambda_i^s V^{-1(j)} - \lambda_j^s V^{-1(i)})\begin{pmatrix} t \\ 1 \end{pmatrix} \geq 0, \quad \forall i = j, i = 1, \ldots, K+L+1; s \in S, \end{aligned} \tag{33}$$

where $V^{-1(j)}$ refers to the jth row of V^{-1}. The LP in (33) has $(K+L)$ variables and $(|S|+1)(K+L)+1$ constraints. Thus, the dual problem, which is referred to as CRLP and has only $(K+L)$ constraints, is easier to solve generally provided $|S|$ is not too large. Initially, when the number of scenarios under consideration is large, therefore, cell-redefining should not be invoked; besides, in the initial iterations, given the way the initial simplex is constructed, cell-redefinition may not yield a significant reduction of the volume of zero measure. By setting μ^* small enough, one may avoid the latter situation. However, as the simplicial partitioning procedure progresses, resulting cells (Ω_r) are likely to contain a smaller number of scenarios, i.e. $|S_r|$ will be smaller, thus offering a better chance of being redefined to tighten the approximations. Several other remarks are in order:

Remarks

1. If $\mu_j = 0$, then the dimension of the current simplicial cell can be reduced by 1, i.e. the *dimension of uncertainty* is (effectively) one less in the current cell.
2. The multipliers λ^s required in CRLP are already available when allocating scenarios to cells of the simplicial partition. That is, when determining the set of scenarios that belong to the simplicial cell Ω_r, characterized by the *vertex matrix* V_r, the multipliers $\lambda^s = V_r^{-1}\begin{pmatrix} \omega^s \\ 1 \end{pmatrix}$ are computed for each $s \in S$ and consequently, the scenario ω^s is labeled as belonging to cell Ω_r if $\lambda^s \geq 0$.
3. The size of CRLP can be significantly reduced by considering only those scenarios which are extremal in S. One can do so by identifying the convex hull C using simple LP procedures, the details of which we shall omit here.

3.3 *Partitioning strategies*

At the current iteration ν, a cell for further partitioning is chosen by the maximum (probabilistically) weighted difference (WDIFF) criterion, i.e.

$$\max_{r=1,\ldots,R_\nu} P_r[\phi_U^{\nu,r}(x_L^\nu) - \phi_L^{\nu,r}(x_L^\nu)]. \tag{34}$$

When only a single cell is further partitioned at some iteration, it is referred to as *single partitioning*. However, when partitioning multiple cells, referred to as *multiple partitioning*, we pick all cells having WDIFF within 60% of the maximum in (34). This is typically done in order to save computations when the lower bounding solutions x_L^ν are stabilized, as indicated by the percent change in x solutions (for 2-norm):

$$\left| \frac{\|x_L^{\nu+1}\| - \|x_L^\nu\|}{\|x_L^\nu\|} \right| \times 100\%. \tag{35}$$

Given a cell chosen for further sub-division, the ways in which partitioning is actually carried out can have a major impact on solution efficiency. This was demonstrated for rectangular partitioning in [10]. In our case, with simplices, one may proceed in at least two ways: *radial partitioning* or *bi-partitioning*. In radial partitioning of a cell Ω_r, a point interior to Ω_r is chosen and it is joined to all vertices of Ω_r to generate $(K + L + 1)$ sub-simplices. While increasing the number of cells in this manner leads to rapid growth of the size of the lower approximation, if implemented in the initial partitioning iterations, it may also adversely affect the convergence behavior of the x-solutions due to the way in which partitioning is performed. Furthermore, it can over-partition a cell and lead to wasteful computational effort. Therefore, we resort to the bi-partitioning procedure in which a cell Ω_r is split into only two sub-simplices by choosing a single *partition plane* that passes through vertices, dividing an edge. In so doing, thus, one needs to address:

(1) the selection of the partitioned edge, and

(2) the point at which the selected edge is to be partitioned.

3.3.1 *Bisection partitioning*

The simplest strategy in partitioning is to choose the edge with the longest length and then partition it at the mid-point. That is, if the longest edge is determined by the two vertices (w^i, w^j), it will be partitioned at $\hat{w} := (w^i + w^j)/2$ under bisection partitioning, herein referred to as the Δ_0-strategy. This leads to two new simplicial cells after partitioning given by

$$conv\{w^1, \ldots, w^{i-1}, \hat{w}, w^{i+1}, \ldots, w^{K+L+1}\}$$

and

$$conv\{w^1, \ldots, w^{j-1}, \hat{w}, w^{j+1}, \ldots, w^{K+L+1}\}.$$

Note that the corresponding inverses of the two vertex-matrices can be obtained by single-pivot updates of the inverse V^{-1} of the cell just partitioned. The general motivation for bisection partitioning is that cells can be driven uniformly to very small sizes, thereby assuring theoretical convergence, see the discussion in section 2.3.

3.3.2 *Partitioning based on nonlinearity*

Instead of relying on (a passive strategy such as) arbitrarily small cell-sizes, by using more efficient partitioning directions based on the behavior of the recourse function in the chosen cell one may hope to accelerate the convergence process. Recalling that if the function is linear on a cell, then the approximations provide exact expected value, the motivation is to evaluate the *degree of nonlinearity* of the recourse function along edges of Ω_r, see [10] and [20], for instance. However, since both ξ and η coordinates may vary along certain edges, the standard procedure in the latter references for evaluating nonlinearity cannot be employed.

Let us drop the index r of the chosen cell Ω_r in the current partitioning iteration ν. Also drop the iteration index ν from the current lower bounding solution x_L^ν that is being used. Suppose the vertices $w^i \equiv (u^i, v^i)$ and $w^j \equiv (u^j, v^j)$ represent the edge to be divided and that $u^i \neq u^j$ and $v^i \neq v^j$. We employ one of two possible methods in such cases to measure the degree of nonlinearity associated with the edge (i, j).

The first measure is based on the directional derivative information. Let D_{ij} denote the directional derivative of the recourse function at w^i along the direction $w^j - w^i$. Similarly, D_{ji} is also defined. To determine D_{ij}, let (y^i, π^i) be the optimal primal and dual solutions of the recourse linear program determining $\phi(x_L, w^i)$. Then

$$D_{ij} = (y^i, \pi^i)\,\frac{d_{ij}}{|d_{ij}|}, \tag{37}$$

where

$$d_{ij} := \begin{pmatrix} q(v^j) - q(v^i) \\ h(u^j) - T(u^j)x_L - h(u^i) + T(u^i)x_L \end{pmatrix}. \tag{38}$$

Definition 3.3
The local nonlinearity measure Δ_1 for the edge (i, j) is defined by

$$\Delta_1(i, j) := |D_{ij} + D_{ji}|. \tag{39}$$

The motivation for the measure Δ_1 is that if the recourse function is linear on the edge (i, j), then $\Delta_i(i, j) = 0$. Consequently, $\Delta_1(i, j) \neq 0$ implies that the recourse function is nonlinear along the edge (i, j).

The second measure of nonlinearity, denoted $\Delta_2(i, j)$, considers the displacement from w^i to w^j in two possible two-step movements: $w^i \to \hat{w} := (u^j, v^i) \to w^j$ and $w^i \to \tilde{w} := (u^i, v^j) \to w^j$. Then, for each possible two-step displacement, the recourse function is approximated by two-piece linear functions, determined by primal and dual solutions of the recourse problem at w^i and w^j. Then the functional displacement due to this two-piece linear approximation is computed at both $\hat{w}$ and $\tilde{w}$, of which the minimum is taken as the nonlinearity measure; see figure 4 for an illustra-

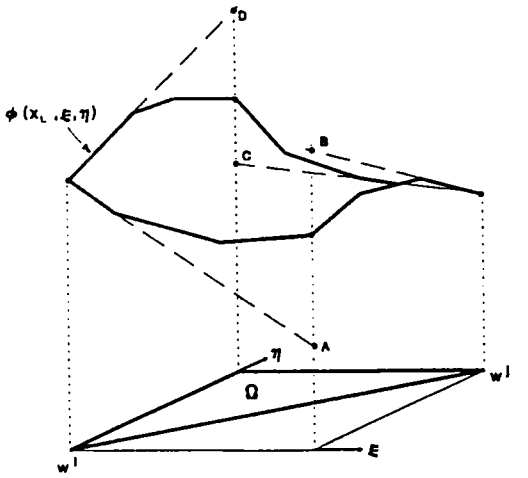

Figure 4. Nonlinearity measure $\Delta_2(i, j) = \min[AB, CD]$.

tion in $\mathbb{R}^2$. That is, denoting the optimal primal and dual solutions corresponding to $\phi(x_L, w^i)$ and $\phi(x_L, w^j)$ by (y^i, π^i) and (y^j, π^j), respectively, we have

$$\phi(x_L, \hat{w}) \geq \pi^i[h(u^j) - T(u^j)x_L] \quad \text{and} \quad \phi(x_L, \hat{w}) \leq q(v^i)'y^j. \tag{40}$$

Furthermore,

$$\phi(x_L, \tilde{w}) \geq \pi^j[h(u^i) - T(u^i)x_L] \quad \text{and} \quad \phi(x_L, \tilde{w}) \leq q(v^j)'y^i. \tag{41}$$

Consequently, the second measure of nonlinearity for the edge (i, j) is defined by

$$\Delta_2(i, j) := \min\{q(v^i)'y^j - \pi^i[h(u^j) - T(u^j)x_L], q(v^j)'y^i - \pi^j[h(u^i) - T(u^i)x_L]\}. \tag{42}$$

The intuition behind the measure Δ_2 is that if the recourse function is almost linear in the rectangle determined by the points w^i, $\hat{w}$, w^j and $\tilde{w}$, then Δ_2 is expected to be smaller. More importantly, if either $u^i = u^j$ or $v^i = v^j$ occurs, then the nonlinearity measure Δ_2 reduces to that proposed in [20]. As a final remark, observe that in the latter event, the first measure Δ_1 is a fairly poor measure of nonlinearity and thus should not be used, except possibly when $u^i \neq u^j$ and $v^i \neq v^j$.

3.3.3 Conditional-mean partitioning

Once an edge (i, j) is selected – corresponding to the maximum of the chosen nonlinearity measure – one can proceed with partitioning the current simplex either by bisecting the edge (i, j) or determining a partitioning plane based on probability mass. The former approach is referred to as mid-point partitioning and is identified by $\hat{\Delta}_1$ and $\hat{\Delta}_2$.

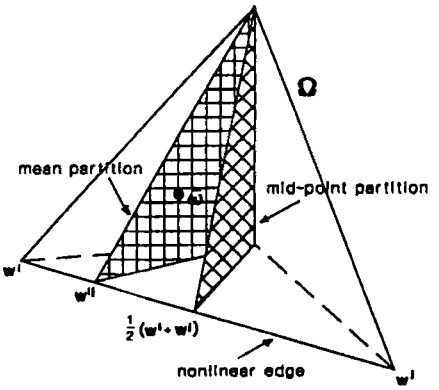

Figure 5. Conditional-mean and mid-point partitioning.

In the second approach, in order to balance the probability mass in the resulting two cells after partitioning, one is motivated to determine a partitioning plane that passes through the conditional mean (denoted by $\overline{\omega}$) for the current cell Ω. (See figure 5.) This practice is consistent conceptually with that in rectangular domains. The following result is useful in accomplishing this objective.

Proposition 3.4

For the $(K+L)$-dimensional simplex Ω with vertices w^k, $k = 1,\ldots,K+L+1$, let ρ be defined by (10) for the first moments $\overline{\omega}$. Then, the hyperplane in $\mathbb{R}^{K+L}$ determined by $\overline{\omega}$ and the $(K+L-1)$ vertices w^k $(k = 1,\ldots,K+L+1,\ k \neq i,\ k \neq j)$ intersects the edge (i,j) at

$$w^{ij} := \frac{\rho_i}{\rho_i + \rho_j} w^i + \frac{\rho_j}{\rho_i + \rho_j} w^j. \tag{43}$$

Proof

Let the partitioning plane in the proposition be represented by $a'\omega = b$ for coefficients $a \in \mathbb{R}^{K+L}$ and $b \in \mathbb{R}$. Thus, we have

$$a'\overline{\omega} = b \quad \text{and} \quad a'w^k = b, \quad \forall k = 1,\ldots,K+L+1,\ k \neq i,\ k \neq j. \tag{44}$$

Multiplying the last $(K+L-1)$ equalities by the corresponding ρ_k and summing up,

$$\sum_{k \neq i,j;k=1}^{K+L+1} \rho_k [a'w^k] = \sum_{k \neq i,j;k=1}^{K+L+1} \rho_k b. \tag{45}$$

Subtracting (45) from the first equality in (44) and rearranging, $a'w^{ij} = b$ follows, where w^{ij} is defined in (43). Thus, w^{ij} belongs to the hyperplane of interest. Moreover, w^{ij} belongs to the edge (i,j), which completes the proof. □

Notice that $\rho_i \neq 0$ and $\rho_j \neq 0$ must hold in the above proposition for w^{ij} not to be one of the vertices. Consequently, this ensures that a (valid) partition of Ω is available. The latter condition is trivially satisfied throughout the partitioning process since all cells are maintained to be of full-dimensionality, by following the first remark in section 3.2. Also see the implementation in section 4. The strategy that utilizes such conditional-mean partitioning planes is referred to as $\overline{\Delta}_1$ and $\overline{\Delta}_2$, corresponding to the nonlinearity measure.

4 Implementation

In the implementation of the Second Moment Approximation for lineaR sTochastic programs (with two decision-stages) – SMART –, it is recognized that the core problem data should be kept separately from the uncertainty data. The core problem data is the first-stage feasible set X, objective costs c, and the second-stage fixed matrices W, Q, $T_0,\ldots,T_K$, and H, and the vectors h_0, and q_0, while the stochastic data is the scenario set $\{(\omega^s, p_s): s \in S\}$. Thus, unless one desires to solve the Grand LP formulation, the core-input is not combined with the stochastic input. Instead, the SMART routine clusters the scenario set (initially, with just one cluster) according to a certain simplicial partition (of the underlying domain Ω), accomplished by the SIMDEC module, while the Second-Order Scenario Approximation (SOSA) module computes the upper and lower approximating scenario distributions corresponding to the output of SIMDEC. The BOUNDS module then generates the approximate LP files (in MPS format) by combining SOSA with the core input according to the cell (geometric) information in SIMDEC. After solving the approximate LPs, if the relative gap of the upper and lower bounds is not within the prescribed tolerance ε, then the lower bounding solution thus obtained becomes the input to SIMDEC for further refinement. This general scheme is depicted in figure 6. In what follows, we provide a brief description of each of these modules.

4.1 *SLP modules*

The CORE-INPUT module incorporates the Kall-Keller [28] GENSLP code for generating the fixed matrices and vectors where the first stage feasible set X is generated by the linear constraints $Ax = b$, $x \geq 0$ for matrix $A \in \mathbb{R}^{m_1 \times n_1}$. GENSLP is a random problem generator which allows *complete* recourse matrices W. The user inputs are the dimensions and density of the matrices A and W, as well as the dimensions of uncertainty K and L. The scenario generator module SCENGEN uses the inputs K and L, the degree of uncertainty for each random parameter specified by a lower and upper bound, as well as the number of scenarios N ($:= |S|$) which is required to be at least 2^{K+L}. It then assigns a scenario to each of the 2^{K+L} vertices of the rectangle formed by the upper and lower bounds on random parameters, and the remaining $N - 2^{K+L}$ scenarios are generated by taking (0–1 uniformly generated) random convex

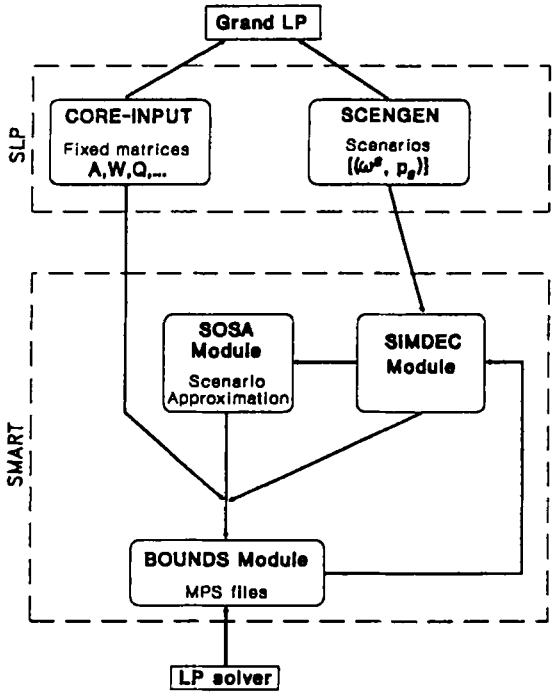

Figure 6. Schematic diagram of the SMART modules.

combinations of these vertices. Then each scenario is treated equally likely with probability $1/N$. This way scenarios may be generated to be statistically dependent.

4.2 *SMART modules*

The three main modules SIMDEC, SOSA, and BOUNDS constitute the SMART routine to solve the generated SLP problem through approximation. These three modules are successively invoked beginning with an initial simplex Ω enclosing the set of scenarios, as given in section 3.1. In particular, it allows sensitivity analysis with respect to the initial simplex parameter δ_0, by fixing δ in (25) as follows:

$$\delta_j = \begin{cases} \delta_0 & \text{if } j = 1, \ldots, K, \\ 1 & \text{if } j = K+1, \ldots, K+L. \end{cases} \tag{46}$$

4.2.1 SIMDEC module

This module performs a simplicial decomposition of the domain Ω, assigns scenarios to cells of the resulting partition, and maintains and updates the geometric information of cells from one partitioning iteration to another. When invoked with a current solution iterate x_L^ν, this module determines

(1) a cell – denoted herein by Ω_r at some partitioning iteration ν – for further partitioning, according to the rule (34),

(2) an edge (w^i, w^j) identified by a partitioning strategy such as Δ_0, Δ_1, or Δ_2, and

(3) a partitioning point w^* of the latter edge, i.e. either the mid-point $(w^i + w^j)/2$ or the point w^{ij} corresponding to the conditional-mean partitioning plane, see (43).

Before partitioning a chosen cell Ω_r, its effective dimension is determined by $Edim(\Omega_r) := (K + L) - |I_0(r)|$, where $I_0(r) := \{i : \mu_i = 0,\ i = 1,\ldots,K + L + 1\}$, which is the set of vertices that can be removed (from the current vertex set $\mathcal{V}_r$) to yield a lower dimensional simplicial cell, denoted by Ω_r as well. Consequently, if $Edim(\Omega_r) + 1 \geq |S_r|$, it is preferable to solve the exact problem for cell Ω_r rather than approximations, and hence the cell is marked NOAPPX for the exact LP formulation later. Otherwise, Ω_r is partitioned into two new cells Ω_r^1, and Ω_r^2 and the information is updated according to the following procedure:

Procedure-SP:

begin

$S^\nu := \{S^{\nu-1} \setminus \Omega_r \cup \{\Omega_r^1, \Omega_r^2\}$;

$\mathcal{V}_r^1 := \{\mathcal{V}_r \setminus \{w^i\}\} \cup \{w^*\}$;

$\mathcal{V}_r^2 := \{\mathcal{V}_r \setminus \{w^j\}\} \cup \{w^*\}$;

$(V_r^1)^{-1} = U_i V_r^{-1}$;

$(V_r^2)^{-1} = U_j V_r^{-1}$;

$S_r^1 = \varnothing$;

$S_r^2 = \varnothing$;

for all $s \in S_r$ **do**

$\lambda\text{temp} = U_i \lambda^s$;

if $\lambda\text{temp} \geq 0$ **then begin**

$\lambda^s = \lambda\text{temp}$;

$S_r^1 = S_r^1 \cup \{s\}$;

else begin

$\lambda^s = U_j \lambda^s$;

$S_r^2 = S_r^2 \cup \{s\}$;

end; (*if*)

end; (*for*)

end;

For the cell Ω_r, U_i is the *eta matrix* with its ith column being the *eta column*

$$\left(\frac{-t_1}{t_i},\ldots,\frac{-t_{i-1}}{t_i},\frac{1}{t_i},\frac{-t_{i+1}}{t_i},\ldots,\frac{-t_{K+L+1}}{t_i}\right)', \quad \text{where} \quad t = V_r^{-1}(w^{*\prime},1)'. \quad (47)$$

The Procedure-SP can be invoked more than once before leaving the SIMDEC module, as determined by multiple cell partitioning, see [34] and the discussion that follows.

Cell-Redefining (C-R)

The C-R procedure is a part of the SIMDEC module for redefining the vertices of cells obtained through simplicial partitioning. As described in section 3.2, each new cell Ω_r created in the current partitioning step is *admissible* for redefining provided that the corresponding CRLP, i.e. the dual of (33), does not have too many variables, i.e. $|S_r|(K+L) \leq 1000$. For an admissible cell, the vertices that will be attempted for redefining are selected by the criterion $\mu_j \leq \mu^*$ (where the threshold μ^* is set to be 0.70, see section 3.2), for if not, the likelihood of obtaining a significant reduction in the cell volume is poor. As mentioned before, $\mu_j = 0$ implies that this vertex w^j may be eliminated completely to yield a lower dimensional simplex.

The Procedure-CR below, also (heuristically) tests the *replaceability* of the chosen vertex w^j by checking if there are scenarios on the facets (incident to w^j) describing the simplex, for if so, one would not expect to obtain a significant reduction in the size of the current simplex.

The input to Procedure-CR is the set of multipliers $\Lambda_i := \{\lambda_i^s : s \in S_r\}$, for $i = 1,\ldots, K+L+1$ and $\mu_i := \max \Lambda_i$. λ's are already available within SIMDEC. Denote the potential set of vertices for redefining by $J_0 := \{i : \mu_i \leq \mu^*, \mu_i > 0\}$. For a chosen vertex w^j, denote the optimal value of (33) by $CRLP(w^j) := V^{-1(j)}\binom{t^*}{1}$, where t^* is the point that replaces w^j.

Define J_1 as the index-set of the already re-defined vertices. Initialize $J_1 = \varnothing$.

Procedure-CR

Step 1 If $J_0 = \varnothing$, then STOP.

Step 2 Find $j := \arg\min\{\mu_i : i \in J_0\}$.

Step 3 Check *replaceability* of w^j as follows.

begin
  for $i = 1$ **to** $K+L+1$ **do if** $i \neq j$ **then begin**
    if $\min \Lambda_i > 0$ **then** go to step C-R 4;
  end (*for*);
  go to step C-R 5;
end;

Step 4 Solve dual of (33) to obtain $CRLP(w^j)$ and t^*.
If $CRLP(w^j) > 0$ then update the geometric information as follows.
begin
$w^j = t^*$;
$V_r^{-1} = U^j V_r^{-1}$;
for all $s \in S_r$ **do**
$\lambda^s = U^j \lambda^s$;
end;
Update $\Lambda_i := \{\lambda_i^s : s \in S_r\}$; and $\mu_i := \max \Lambda_i$, $\forall i = 1,\ldots,K+L+1$;
$J_0 = \{i : \mu_i \leq \mu^*, \mu_i > 0, i = 1,\ldots,K+L+1\}$
end; (U^j is the resulting eta-matrix when using (47) with t^* in place of w^*).

Step 5 Update: $J_1 = J_1 \cup \{j\}$
$J_0 = J_0 \setminus \{J_0 \cap J_1\}$ and go to step 1.

4.2.2 SOSA module

Given the geometric information of newly created cells in the simplicial partition, as well as the clustering of scenarios from SIMDEC, the SOSA module first computes the required second-moment information for each new cell Ω_r. Consequently, according to theorem 2.1, the lower and upper approximating scenario distributions are computed as outputs. However, if the cell has been marked NOAPPX in the SIMDEC module, see section 4.2.1, the exact recourse problem is developed for this cell Ω_r and is maintained throughout the remaining partitioning iterations ν. This is the basic level of operation of this module, as selected by SWITCH = 0 and referred to as SMART. It also provides several switches for computing other approximate distributions. These are described briefly next:

(a) SWITCH = 1: Each newly created simplex Ω_r is projected to ξ and η domains to obtain the vertices of the marginal domains Ξ and Θ. Then, only first and cross moment information for the scenarios in Ω_r is computed and used in developing lower and upper approximations according to [8], see the discussion in section 2.1. (In the computational section, we refer to this setting as Simplicial Projection With First and Cross moments: SPWFC.)

(b) SWITCH = 2: Just as in SWITCH = 1, compute the lower approximation using first and cross moments, but the upper approximation is computed using second moments following SWITCH = 0.

The implementation also allows maintaining different switches (either SWITCH = 0, 1, or 2) for different cells, as determined by choices in the previous partitioning iterations. That is, for instance, a cell under SWITCH = 0 in iteration ν may later be partitioned and labeled under SWITCH = 2 in iteration $\nu + 1$. When used, this setting is referred to as Hybrid-SMART (or HYBRID).

4.2.3 *BOUNDS module*

For the fixed data from CORE-INPUT, using the lower approximation from SOSA and the geometric information from SIMDEC, the BOUNDS module first generates the lower bounding linear program LBLP in MPS format, according to the SWITCH used in SOSA. For SWITCH = 0, see the formulation Z_L^ν in (20). Then an LP solver is called – in our case, IBM's OSL Subroutine Version 1.2 – to solve this LP and the lower bounding solution x_L^ν is obtained.

Then, for this first-stage solution x_L^ν, combining with CORE-INPUT data and information from SOSA and SIMDEC, the upper bounding LPs for each cell Ω_r are formed in MPS format and solved successively. The objective values $\phi_U^{\nu,r}(x_L^\nu)$ thus obtained, see (19), are accumulated for each cell and weighted by the associated probabilities P_r, $r = 1,\ldots,R_\nu$, to obtain the upper bound $\bar{Z}_U^\nu$ in (21).

Subsequently, the relative gap test is performed according to (22) to determine if termination should occur for the specified ε. If not, the current solution x_L^ν, as well as the weighted difference WDIFF of upper/lower bounds for all the current cells become the input to the SIMDEC module, and the iteration is continued.

The preceding description is the basic operation of the BOUNDS module, as selected by OPTION = 0. The other two options available are outlined below:

(a) OPTION = 1: If the stability of the generated x-solutions, as measured by (35), indicates that the current lower bounding solution has not changed significantly over the last few iterations, then the BOUNDS module can set the current x_L^ν fixed, and solve lower bounding LPs on each cell separately, thereby avoiding the solution of a larger LBLP. If this mode of operation does not yield a significant drop in the relative gap in the next few iterations, it can switch back to the mode determined by OPTION = 0. Observe that under OPTION = 1, the upper and lower bounding LPs are only solved for those cells newly created by SIMDEC.

(b) OPTION = 2: Although it is not of much use, this mode allows the solution of the full upper bounding LP to determine the tighter upper bound Z_U^ν in (21) along with its solution x_U^ν, see (20). The difference in x_L^ν and x_U^ν may serve as an indicator for convergence of lower bounding solutions, although it is not utilized in the implementation.

5 Computational results

The implementation in the preceding section was coded in FORTRAN 77 and developed on a SUN SparcStation 2 running under UNIX. OSL Version 1.2 was used as the subroutine for solving LPs. The code was compiled using the optimizing compiler flags -O4 -cg89.

The SMART code allows up to 16 (dependent) random variables (max: $K = 8$, $L = 8$) and up to 100,000 scenarios. The maximum number of simplicial partitions

allowed is 20. The core input has the limits: $m_1, n_1, m_2, n_2 \leq 100$ and the stochastic linear programs are generated by specifying a random seed for GENSLP and a second random seed for SCENGEN. The above limitations are set only due to memory restrictions of the machine.

All coefficients in the CORE-INPUT are generated uniformly from the interval $[-10, 10]$. For the purposes of the experiments reported here, all scenarios are generated such that the corresponding random variables have a maximum spread of $[-5, 5]$. Furthermore, only the complete-recourse matrix option is used in GENSLP along with matrices being specified to have 10% density.

In each category of experimentation, at least 5 problems have been solved and averaged when comparisons are made. Specifically, the following issues are investigated:

(1) the sensitivity of the initial simplex (specified by the value of δ_0) with respect to second moment approximations;

(2) the effect of joint partitioning compared to coordinate axis partitioning with respect to first-order approximations;

(3) the effect of using second-order information over first-order information when using joint-partitioning;

(4) the effect of partitioning strategy and multiple partitioning on second moment approximations.

The set of problem characteristics given in table 1 serves as the base for the above investigations.

Table 1

Randomly generated sample problem categories.

Problem category	K	L	(m_1, n_1, m_2, n_2)	# Scenarios
1	5	5	(20,40,20,40)	2048
2	5	2	(20,40,20,40)	2048
3	8	0	(10,20,10,20)	1024
4	1	1	(5,10,5,10)	[100,700][a]
5	5	5	(10,20,10,20)	1024
6	8	8	(30,50,30,50)	65536

[a]Number of scenarios is increased from 100 to 700 in steps of 100 scenarios.

The sensitivity of the approximations to the initial simplex is investigated by varying δ_0, see (46), using the problem categories #1 and #2. In both cases, single-partitioning is used with the $\bar{\Delta}_2$ strategy. As shown in figures 7 and 8, increasing δ_0 has the tendency to strengthen the second moment approximations in the initial

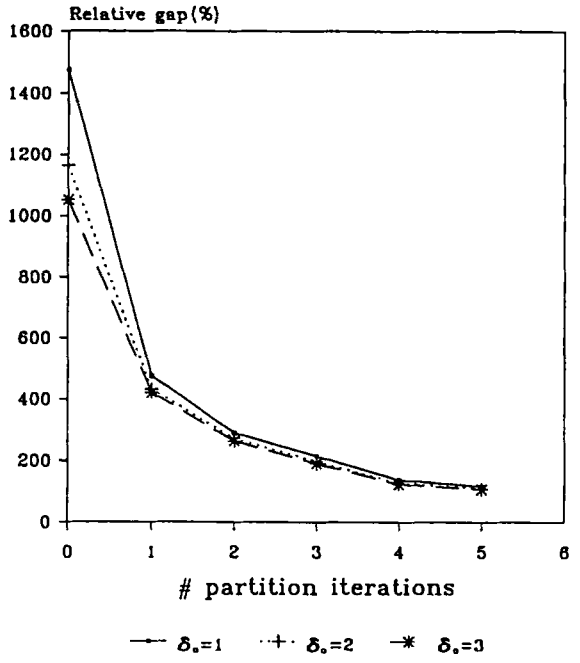

Figure 7. Sensitivity on the initial simplex (2048 scen., 10 r.v.).

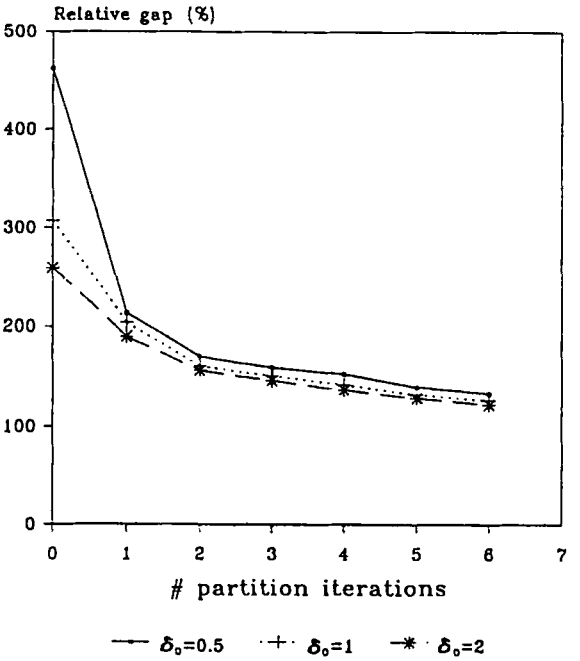

Figure 8. Sensitivity on the initial simplex (2048 scen., 7 r.v.).

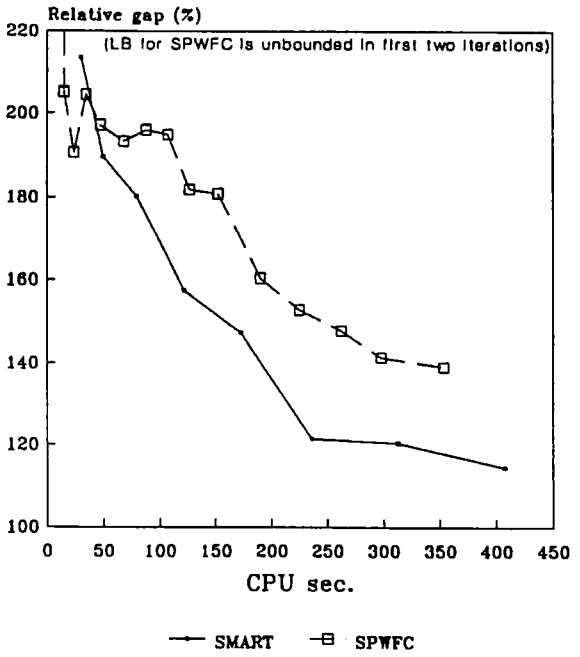

Figure 9. Effect of second moments
(problem category #3).

iterations, but eventually this effect is diminished. The noted behavior is for the case with no cell-redefining. Accordingly, we have set $\delta_0 = 1$ for the remaining experiments.

Figure 9 uses the problem category #3 to illustrate the effect of using second-moment information over first moments under the partitioning strategy $\overline{\Delta}_2$ involving single partitioning. Given that this problem category does not have η random variables, Δ_1 strategies are not appropriate, see the discussion in section 3.3.2. Also notice that since $L = 0$, in the simplicial projection bounds, SPWFC, the resulting lower bound is Jensen's bound. It is evident that SMART (with SWITCH = 0) produces much faster convergence of objective values compared to using only first moments. Another significant departure from first-order approximations is that much of the adjustments in (lower bounding) x solutions occur in the initial iterations in SMART, while SPWFC continues to change x solutions until much later. Thus, convergence of x may be attained in SMART much faster. The effect of using the conditional-mean partitioning plane versus the mid-point partitioning plane, under single partitioning, is depicted in figure 10 for the same problem category. Our computational experience suggests that mid-point partitions may be dominated by conditional-mean partitions.

It may be noted that while the second-order lower bound is tighter than Jensen's bound (for $L = 0$), the former involves a much larger lower bounding LP problem.

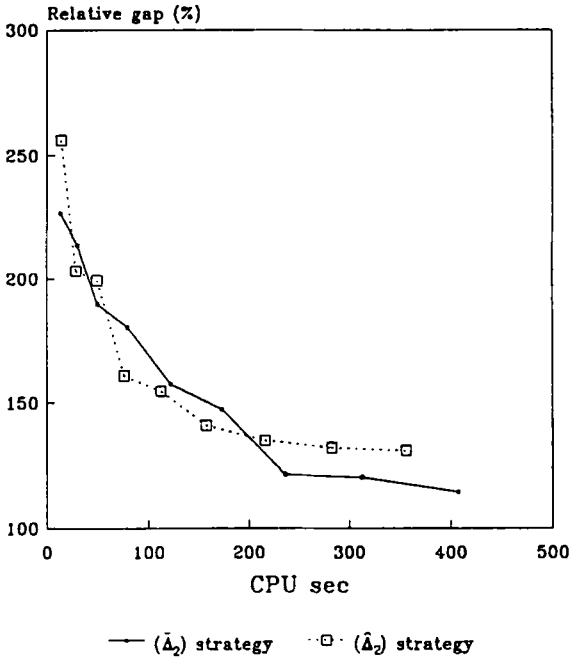

Figure 10. Effect of partitioning point (problem category #3).

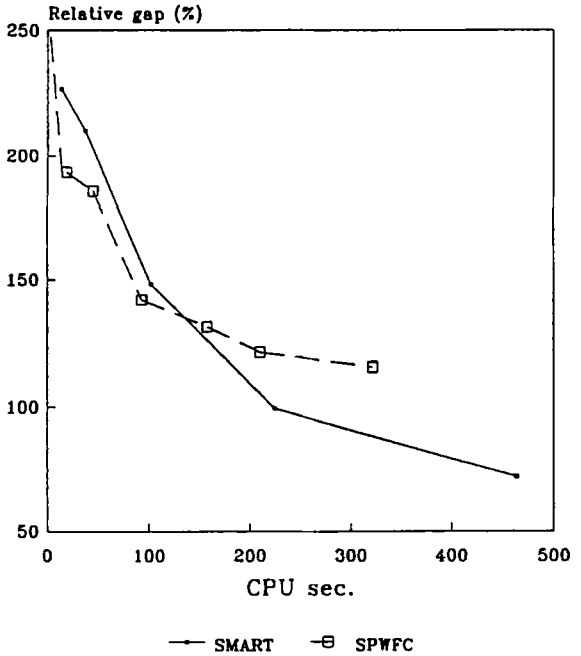

Figure 11. Effect of multiple partitioning (problem category #3).

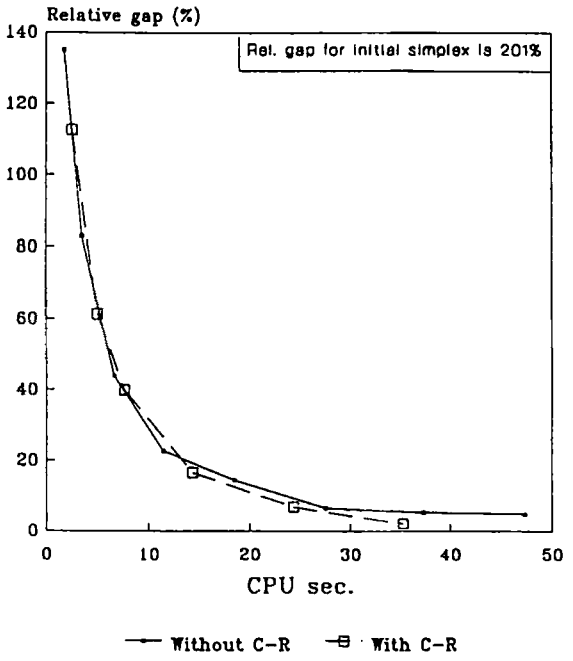

Figure 12. Effect of cell-redefining (problem category #4).

However, since second moment bounds tend to force x solutions towards convergence much faster, an appropriate strategy might be to use multiple partitioning after the initial few iterations. As figure 11 shows, the latter practice indeed is the most pragmatic computational procedure. Consequently, in the remaining experiments we have utilized multiple partitioning according to the 60% rule mentioned in section 3.3.

The effect of cell-redefining strategy is illustrated using the problem category #4, see figure 12, where the partitioning strategy is $\bar{\Delta}_2$ and the number of scenarios is 200. Observe that in the initial iterations, there is very little improvement brought about by the C-R strategy. But, after a significant refinement of the simplicial partition, the C-R strategy is able to effectively remove the areas of zero measure. Consequently, the approximations become tighter and convergence is achieved faster. Using the same problem category, we illustrate the sensitivity of SMART solution time to the number of scenarios in the problem, see figure 13. In the same figure, the CPU times for solving the grand LP are plotted. A word of caution in interpreting this comparison is in order. While the grand LP solution time is adversely affected by the number of scenarios, rather than the dimension of uncertainty (i.e. K and L), the SMART CPU times are more sensitive to the sizes of K and L, rather than the number of scenarios. In this figure, we have used $K = L = 1$ to illustrate the former sensitivity. However, if K and L are increased, the SMART CPU times are affected adversely. Nevertheless, increasing the dimensions K and L, while holding the number of

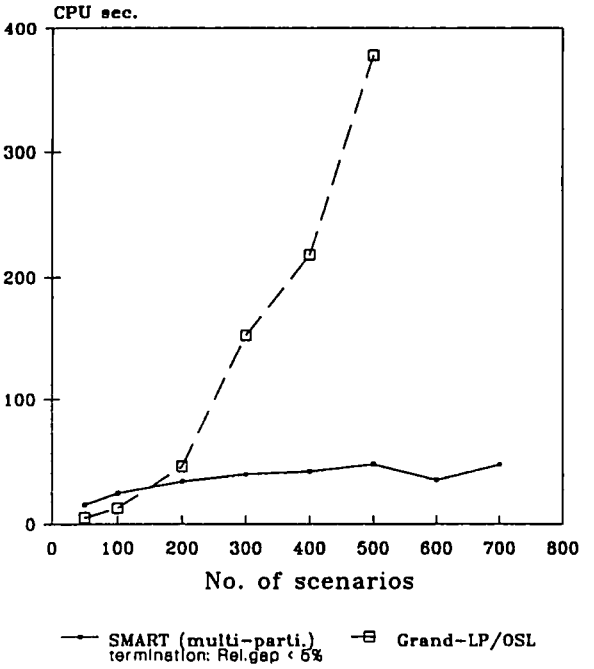

Figure 13. Sensitivity on number of scenarios (problem category #4).

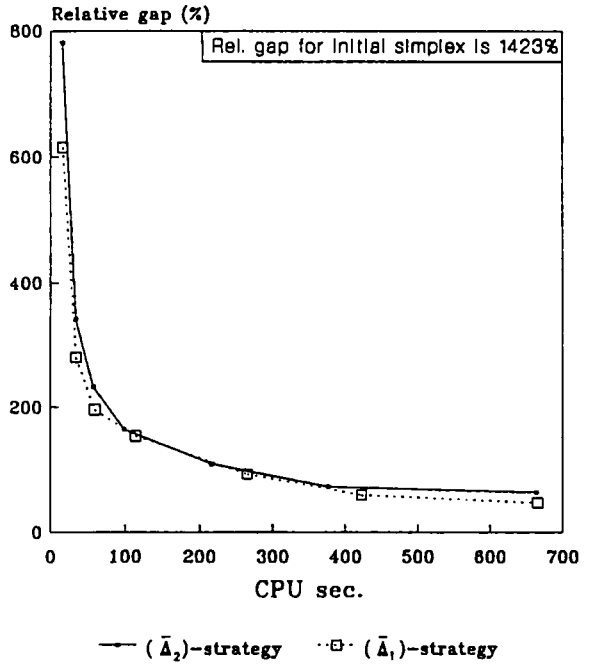

Figure 14. Effect of partitioning strategy (problem category #5).

scenarios constant, leads to larger volumes of zero measure in the cells of the simplicial partition. Consequently, when the C-R strategy is utilized, one has better chances of removing such areas of zero measure and even perhaps reducing the effective dimension of the resulting cells.

When both K and L are non-zero, one may use either of the strategies Δ_1 or Δ_2. This effect is presented in figure 14 using conditional-mean partitioning for problem category #5. Our limited computational experience does not favor one over the other, although in many instances where K and L are roughly about the same size, $\overline{\Delta}_1$ does seem to have a slight advantage over $\overline{\Delta}_2$. Figure 15 shows that simplicial bounds with

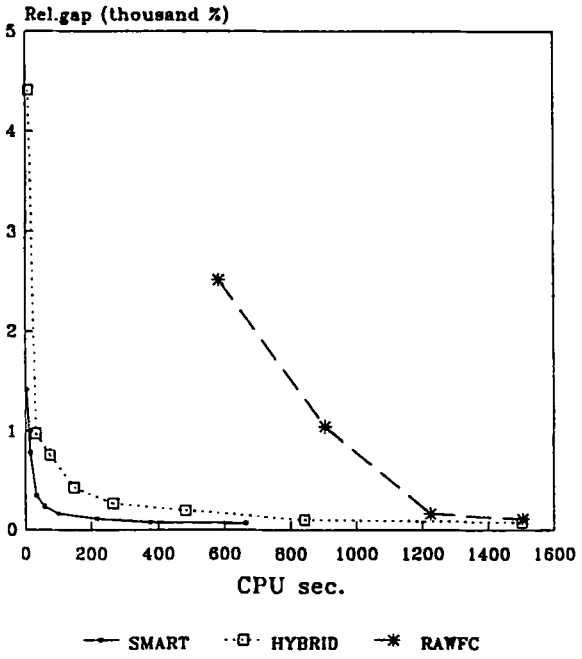

Figure 15. Comparison of SMART with Hybrid version (problem category #5).

second moments are much superior to using either rectangular first order bounds or simplicial first order bounds. This superiority becomes more evident when both K and L are significantly large. As an example of larger K and L, we use the problem category #6 and graph the performance of SMART in figure 16. It is the general observation that the second-order lower bounds tend to converge much faster than the upper bounds. This observation may suggest that one need not solve the complete lower bounding formulation for the entire domain towards the final iterations, but rather solve (smaller) lower bounding problems, by fixing x, only for those cells which are newly partitioned. Although the SMART code allows this practice, we have not performed extensive testing so far.

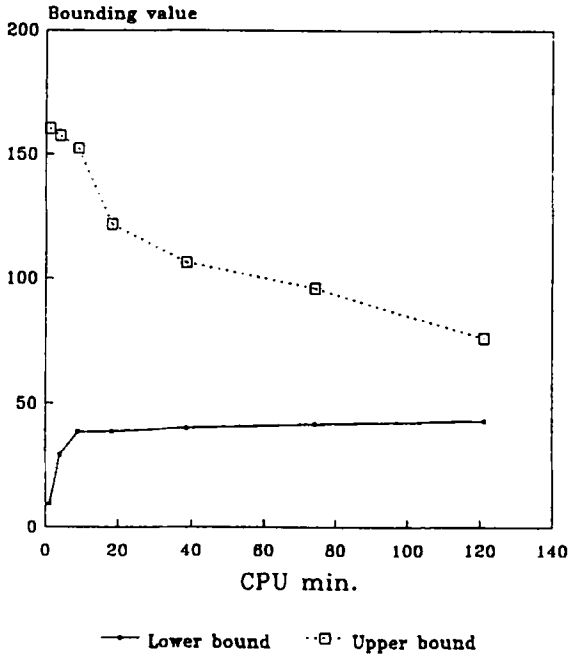

Figure 16. Behaviour of SMART bounds
(problem category #6)

6 Concluding remarks

In this paper, we have demonstrated the effectiveness of incorporating second-order approximations of the expected recourse function when solving two-stage stochastic linear programming problems. Using second-order moments as well as partitioning the domain along joint directions yield the desired computational efficiency when solving larger problems. The limited computational experience gained here, using the implementation SMART, provide sufficient motivation for further testing of such solution procedures. In particular, when the number of scenarios is large, but the dimension of uncertainty is sufficiently small – up to 16 in our computational experiments –, the second-order scenario approximations may prove a powerful alternative for solving stochastic programs.

While the discussion in the paper is on two-stage versions of the model, extensions of this general approach for multistage models seem fairly straightforward, at least conceptually. That is, viewing the recourse function at each decision stage as a saddle function, approximating scenario distributions may be constructed stage-wise. Such an approach is taken by Frauendorfer [18, 19] using first-order approximations. However, the apparent shortcoming in a direct implementation is that when the number of decision stages is large, the resulting number of approximating scenarios can become too excessive to warrant the use of approximations as a solution approach.

An alternative approach is to view the complete set of scenarios, spanning the full number of decision stages, under one domain and employ approximations on it. Then, of course, one would have to assure, among other things, that solutions obtained through such representative scenarios do converge to true solutions. These are potentially useful avenues for future research.

References

[1] J.R. Birge, An L-shaped method computer code for multistage linear programs, in *Numerical Techniques for Stochastic Optimization*, Y. Ermoliev and R.J-B Wets, eds., Springer, 1988.

[2] J.R. Birge, C.J. Donohue, D.F. Holmes and O.G. Svintsitsli, A parallel implementation of the nested decomposition algorithm for multistage stochastic linear programs, Technical Report 94-1, Department of Industrial and Operations Engineering, University of Michigan, 1994.

[3] J.R. Birge and R.J-B Wets, Designing approximation schemes for stochastic optimization problems, in particular for stochastic programs with recourse, Mathematical Programming Study 27, 1986, 54–102.

[4] D.R. Cariño, T. Kent, D.H. Myers, C. Stacy, M. Sylvanus, A.L. Turner, K. Watanabe and W.T. Ziemba, The Russell–Yasuda Kasai model: An asset-liability model for a Japanese insurance company using multistage stochastic programming, Interfaces 24, 1994, 29–49.

[5] G.B. Dantzig and P.W. Glynn, Parallel processors for planning under uncertainty, Annals of Operations Research 22, 1990, 1–21.

[6] J.H. Dula, An upper bound on the expectation of simplicial functions of multivariate random variables, Mathematical Programming 55, 1992, 69–80.

[7] N.C.P. Edirisinghe, New second-order bounds approximations in stochastic linear programming, to appear in Operations Research.

[8] N.C.P. Edirisinghe and W.T. Ziemba, Bounds for two-stage stochastic programs with fixed recourse, Mathematics of Operations Research 19, 1994, 292–313.

[9] N.C.P. Edirisinghe and W.T. Ziemba, Bounding the expectation of a saddle function, with application to stochastic programming, Mathematics of Operations Research 19, 1994, 314–340.

[10] N.C.P. Edirisinghe and W.T. Ziemba, Implementing bounds-based approximations in convex-concave two-stage stochastic programming, to appear in Mathematical Programming.

[11] Y. Ermoliev, Stochastic quasigradient methods and their application to systems optimization, Stochastics 9, 1983, 1–36.

[12] Y. Ermoliev and A. Gaivoronsky, Stochastic quasigradient methods and their implementation, in *Numerical Techniques for Stochastic Optimization*, Y. Ermoliev and R.J-B Wets, eds., Springer, 1988.

[13] Y. Ermoliev and R.J-B Wets, *Numerical Techniques for Stochastic Optimization*, Springer, 1988.

[14] K. Frauendorfer, SLP recourse problems with arbitrary multivariate distributions – the dependent case, Mathematics of Operations Research 13, 1988, 377–394.

[15] K. Frauendorfer, SLP problems: objective and right-hand side being stochastically dependent – Part II, Manuscript, IOR, University of Zurich, 1988.

[16] K. Frauendorfer, A solution method for nonlinear convex stochastic optimization problems, Operations Research 38, 1990, 761–769.

[17] K. Frauendorfer, *Stochastic Two-Stage Programming*, Springer, Berlin, 1992.

[18] K. Frauendorfer, Barycentric scenario trees in convex multistage stochastic programming, Manuscript, Institute for Operations Research, University of St. Gallen, 1993.

[19] K. Frauendorfer, Multistage stochastic programming: Error analysis for the convex case, to appear in Zeitschrift für Operations Research 39/1, 1994.

[20] K. Frauendorfer and P. Kall, A solution method for SLP problems with arbitrary multivariate distributions – The independent case, Problems of Control and Information Theory 17, 1988, 177–205.

[21] H.I. Gassmann, MSLiP: A computer code for the multistage stochastic linear programming problem, Mathematical Programming 47, 1990, 407–423.
[22] H. Gassmann and W.T. Ziemba, A tight upper bound for the expectation of a convex function of a multivariate random variable, Mathematical Programming Study 27, 1986, 39–53.
[23] J.L. Higle and S. Sen, Stochastic decomposition: An algorithm for two-stage stochastic linear programs with recourse, Mathematics of Operations Research 16, 1991, 650–669.
[24] R. Horst and H. Tuy, *Global Optimization (Deterministic Approach)*, Springer, 1990.
[25] G. Infanger, Monte Carlo (Importance) Sampling within a Benders decomposition algorithm for stochastic linear programs extended version: including results of large-scale programs, Technical Report SOL 91-6, Stanford University, 1991; to appear in: Annals of Operations Research.
[26] P. Kall, *Stochastic Linear Programming*, Springer, Berlin, 1976.
[27] P. Kall, On approximations and stability in stochastic programming, in *Parametric Optimization and Related Topics*, J. Guddat, H.Th. Jongen, B. Kummer and F. Nozicka, eds., Akademie-Verlag, Berlin, 1987, pp. 387–407.
[28] P. Kall and E. Keller, GENSLP: A program for generating input for stochastic linear programs with complete fixed recourse, Manuscript, Institut für Operations Research der Universität Zürich, Zürich CH-8006, Switzerland, 1985.
[29] A. Madansky, Bounds on the expectation of a convex function of a multivariate random variable, Annals of Mathematical Statistics 30, 1959, 743–746.
[30] S.M. Robinson and R.J-B Wets, Stability in two-stage stochastic programming, SIAM Journal of Control and Optimization 25, 1987.
[31] R.T. Rockafellar and R.J-B Wets, Scenarios and policy aggregation in optimization under uncertainty, Mathematics of Operations Research 16, 1991, 119–147.
[32] R. Van Slyke and R.J-B Wets, L-shaped linear programs with application to optimal control and stochastic optimization, SIAM Journal on Applied Mathematics 17, 1969, 638–663.
[33] R.J-B Wets, Convergence of convex functions, variational inequalities and convex optimization problems, in *Variational Inequalities and Complimentary Problems*, R.W. Cottle, F. Giannesi and J-L. Lions, eds., Wiley, New York, 1980, pp. 375–403.