Sampling-based Decomposition Algorithms for Multistage Stochastic Programming

Harsha Gangammanavar*[1]

[1]Department of Operations Research and Engineering Management, Southern Methodist University, Dallas TX

Abstract

Sampling-based algorithms provide a practical approach to solving large-scale multistage stochastic programs. This chapter presents two alternative approaches to incorporating sampling within multistage stochastic linear programming algorithms. In the first approach, sampling is used to construct a sample average approximation (SAA) of the true multistage program. Subsequently, an optimization step is undertaken using deterministic decomposition methods that utilize randomization. A comparison of various algorithms that adopt this approach, including stochastic dual dynamic programming, is provided. In the second stochastic decomposition approach, sampling and optimization are carried out concurrently. In stochastic dynamic linear programming, an example of the second approach, the representation of uncertainty is refined in every iteration of the algorithm. The differences in how the approximations of the cost-to-go functions are constructed under these alternative approaches are illustrated.

1 Introduction

Multistage stochastic programming (MSP) provides a practical framework for modeling and solving problems on sequential decision-making under uncertainty. This framework allows us to avoid myopic plans by considering a decision's effect on long-term costs and future choices. It also allows us to account for uncertainty in future costs and resource constraints. These desirable properties have made MSP an effective decision-making tool in several application domains, including financial planning, supply chain management, power systems operations, and production planning. However, the MSP problems are among the computationally most challenging class of problems to solve. This chapter presents randomization-based solution methods for large-scale MSP problems.

A finite set of decision epochs, referred to as stages, characterizes the MSP problems. New uncertain information is revealed at each stage, upon which one can take a recourse decision in response to earlier decisions and the latest information. Note that this stage decision is made before the realization of future uncertainty. In this regard, the first-stage decision is based only on deterministic data and is, therefore, known as the here-and-now decision. At each stage,

*harsha@smu.edu

the decision is associated with a cost function, and the overall objective is to identify optimal decisions with respect to a deterministic cost of the first stage and an expected cost of future stages.

Consider an MSLP problem defined with a finite lookahead $T < \infty$. Denote by $\mathcal{T} := \{0, 1, \ldots, T\}$ the set of decision epochs or stages. The underlying exogenous uncertainty is modeled as a stochastic process $\{\tilde{\xi}\}_{t=1}^{T}$ defined on a filtered probability space $(\Omega, \mathcal{F}, P)$. A realization of this stochastic process over the entire horizon, i.e. $(\xi_1, \xi_2, \ldots, \xi_T)$, is referred to as the *sample path*. Let x_t denote the decision vector for stage t. This decision vector is confined to a feasible set, denoted by $\mathcal{X}_t(x_{t-1}, \tilde{\xi}_t) := \{x_t \mid D_t x_t = r_t - C_t x_{t-1},\ x_t \geq 0\}$, that depends on the decision of the previous stage and the uncertainty. Using these notations, the problem is stated in the nested form as

$$\min_{\substack{D_0 x_0 = r_0 \\ x_0 \geq 0}} \langle d_0, x_0 \rangle + \mathbb{E}\left[\min_{\substack{D_1 x_1 = r_1 - C_1 x_0 \\ x_1 \geq 0}} \langle d_1, x_1 \rangle + \mathbb{E}\left[\ldots + \mathbb{E}\left[\min_{\substack{D_T x_T = r_T - C_T x_{T-1} \\ x_T \geq 0}} \langle d_T, x_T \rangle \right] \right] \right]. \tag{1}$$

In general, any components of the stagewise cost coefficient c_t, the right-hand side r_t, the recourse matrix D_t, and transfer matrix C_t for $t > 0$ can be affected by the exogenous uncertainty. For $t = 0$, the parameters (c_1, r_1, D_1) are deterministic. The following assumption is made regarding the stochastic process:

(A1) The stochastic process satisfies stagewise independence and affects only the elements of right-hand side of stage constraints (i.e., r_t and C_t).

While some of the algorithms presented in this article accommodate more general stochastic processes, including those with dependence across time, the above assumption is made to illustrate the critical differences in algorithm design under a consistent setting.

For the purposes here, it is worthwhile to view the above MSLP problem in its equivalent recursive form given by

$$h_t(x_{t-1}, \xi_t) = \min\ \{\langle d_t, x_t \rangle + H_{t+1}(x_t) \mid x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)\}, \tag{2}$$

where $H_t(x_t) := \mathbb{E}[h_{t+1}(x_t, \tilde{\xi}_{t+1})]$ for all $t \in \mathcal{T}$ and $H_{T+1}(x_T) = 0$. It is assumed that the MSLP problem satisfies the relatively complete recourse property resulting in finite-valued cost-to-go function $H_t(x_{t-1}) = 0$ for all x_{t-1}. Note that the feasible region depends on the decision vector x_{t-1} of the previous stage and a realization ξ_t of the random vector $\tilde{\xi}_t$.

The fundamental difficulty with solving the MSLP problem in (1) stems from the multidimensional integral (when Ω is a continuous set) for computing the expected cost-to-go function $H_t()$. In such cases, a sample-based approximation of the actual problem provides a viable path to tackle these problems. The sample paths used to build the approximations are usually generated using computer simulations. Alternatively, historical data can also be used for this purpose. This chapter presents two alternative sampling-based approaches that differ in how the sampling and optimization steps interact.

2 Randomized Deterministic Decomposition Methods

In the first approach, a sampling step is undertaken wherein the true distribution of $\tilde{\xi}_t$ is replaced with an empirical distribution based on a random sample of pre-determined fixed size in each

stage $t \in \mathcal{T}$. If N_t is the size of the sample employed in stage t, an approximation of the expected cost-to-go function, denoted by $\widehat{H}_t$, is given by

$$\widehat{H}_t(x_{t-1}) = \frac{1}{N_t} \sum_{i=1}^{N_t} \hat{h}(x_t, \xi_t^i) \tag{3}$$

where, $\hat{h}(x_t, \xi^i) = \min \{\langle d_t, x_t \rangle + \widehat{H}_{t+1}(x_t) \mid x_t \in \mathcal{X}_t(x_t, \xi_t^i)\}$. The resulting problem is a *sample average approximation* of the true problem (1). Notice that the total number of sample paths in the SAA problem is $N = \prod_{t=1}^{T} N_t$. While the cost-to-go function $H_t(\cdot)$ of the original problem is a convex function, its SAA $\widehat{H}(\cdot)$ is a piecewise linear convex function. The stagewise independence of the original stochastic process is preserved in the SAA problem. Both these properties are exploited in the design of the solution algorithms. Furthermore, SAA theory suggests that for a reasonable choice of N_t, the optimal solution to the SAA problem is a reasonable approximation of the solution to the true problem. The reader is referred to Chapter 5 in Shapiro et al. (2014) for a detailed exposition of these results.

For a given sample, the SAA problem is a deterministic approximation of the true stochastic program. As such, it can be tackled using deterministic decomposition approaches. Such methods for MSLP problems date back to the nested Benders decomposition (NBD) algorithm (Birge (1985)), a multistage extension of the L-shaped method (Van Slyke and Wets (1969)). As in the case of the L-shaped method, the NBD builds an outer approximation of the cost-to-go function using a collection of cuts (affine lower bounding functions). NBD applies to a any multistage program where the support of the underlying stochastic process is finite, even those with temporal dependence[1]. The stagewise independence assumption allows us to share the cuts across all the observations in a stage (Infanger and Morton (1996)). Therefore, it is sufficient to maintain a single approximation at each stage which takes the following form for stage t:

$$\begin{aligned} h_t^k(x_{t-1}, \xi_t) = \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)} \; & \langle d_t, x_t \rangle + \eta_t \\ \text{subject to } & \langle -\beta_{t,j}, x_t \rangle + \eta_t \geq \alpha_{t,j} \qquad \forall j \in \mathcal{J}_t^{k-1} \end{aligned} \tag{4}$$

where, $\mathcal{J}_t^k := (\alpha_{t,j}, \beta_{t,j})_{\forall j}$ denotes the collection of cuts added until iteration $k-1$. Subsequently, in the backward pass, the stage approximation is updated by computing new cuts with coefficients $(\alpha_t^{k,i}, \beta_t^{k,i})$ along all unique sample paths $i = 1, \ldots, \prod_{\tau=1}^{t} N_\tau$, for $t \in \mathcal{T} \setminus \{T\}$[2]. With this, an updated collection $\mathcal{J}_t^k$ is obtained by including the new cuts into $\mathcal{J}_t^{k-1}$.

The scenario decomposition method (Mulvey and Ruszczyński (1995)) and the progressive hedging algorithm (Rockafellar and Wets (1991)) provide attractive alternatives to NBD. In every iteration of all these algorithms, computations are carried out along every plausible sample path. The total number of nodes in the scenario tree and the total number of sample paths grow exponentially in the number of stages. Therefore, the deterministic decomposition methods become computationally unwieldy even for small stagewise sample sizes.

2.1 Stochastic Dual Dynamic Programming

To address the computational challenge of the multistage SAA problem, Pereira and Pinto developed an approach that incorporates randomization within the NBD framework (Pereira

[1]The SAA problem is a specific form of such multistage programs where its support is restricted to the observations in the random sample. The focus here is on the SAA problem with interstage independence to present the contrast in sampling approaches of different methods.

[2]Once the recently computed cuts are added to the collection, all the cuts are treated the same. Therefore, the iteration and sample path indices (k and i, respectively, in the superscript) are dropped and a subscript j is used to index elements of $\mathcal{J}_t^k$.

and Pinto (1991)). They called their solution approach *stochastic dual dynamic programming* (SDDP). The algorithm is first described as introduced originally in Pereira and Pinto (1991) and some modifications proposed in subsequent works are presented later.

As in NBD, an iteration of the SDDP algorithm begins by solving a root node. A key distinguishing feature of SDDP, relative to NBD, is how the forward pass computations are carried out. While in NBD, forward pass computations are carried out along all N sample paths, they are carried out only along $\widehat{N} \ll N$ randomly generated sample paths in SDDP. Denote by Ω^k the set of random sample paths. Let $(x_0^k, x_1^{k,i}, \ldots, x_T^{k,i})$ denote the decisions simulated by solving approximate problems of the form in (4) for $i = 1, \ldots, \widehat{N}$. The realized objective function value along each sample path is used to compute a statistical estimate of an upper bound on the optimal objective function value as

$$\hat{z}^k = \frac{1}{\widehat{N}} \sum_{i=1}^{\widehat{N}} \left(\langle d_0, x_0^k \rangle + \sum_{t=1}^{T} \langle d_t^i, x_t^{k,i} \rangle \right).$$

Further, to account for the randomness in the above estimate, the sample variance $\hat{\sigma}_{\hat{z}}^k$ is computed. Let $t_{n-1,a}$ denotes the $(1-a)$-level quantile of a student's t random variable with $n-1$ degrees of freedom. Using the sample variance and quantile, a confidence interval (CI) on $\hat{z}^k$ is constructed as

$$[\hat{z}^k - \epsilon^k, \hat{z}^k + \epsilon^k] \tag{5}$$

where, $\epsilon^k = t_{\widehat{N}-1,a} \frac{\hat{\sigma}_{\hat{z}}^k}{\sqrt{\widehat{N}}}$.

Since SDDP design is based on the principle of outer linearization, the optimal objective function value of the first-stage approximation ((4) with $t = 0$) serves as a lower bound for the optimal value of the SAA problem. Therefore, if the value $\langle d_0, x_0^k \rangle + \eta_0^k$ lies within the upper bound CI given in (5), then the SDDP algorithm can be terminated with the current root-stage solution x_0^k declared as the optimal solution. Otherwise, a backward pass starting from stage $T-1$ to stage 0 is carried out.

In stage t of the backward pass, using the current solution $x_{t-1}^{k,i}$, a subproblem is solved for all possible observations $\xi_t^{i'}$, $i' = 1, \ldots, N_t$. Let $\pi_t^{k,i'}$ and $(\theta_{t,j}^{k,i'})_{j \in \mathcal{J}^k}$ denote the optimal dual solution of $h_t^k(x_{t-1}^{k,i}, \xi_t^{i'})$ for the stage constraints and the cuts that appear in (4). From these dual solution vectors, new cuts are constructed as

$$\begin{aligned} \widehat{H}_t(x_{t-1}) \geq & \frac{1}{N_t} \sum_{i'=1}^{N_t} \left(\langle \pi_t^{k,i'}, r_t^{i'} \rangle + \sum_{j \in \mathcal{J}^k} \theta_{t,j}^{k,i'} \alpha_{t+1,j} + \langle \langle \pi_t^{k,i'}, -C_t^{i'} \rangle, x_{t-1} \rangle \right) \\ = & \alpha_t^{k,i} + \langle \beta_t^{k,i}, x_{t-1} \rangle \qquad \forall i = 1, \ldots, \widehat{N}_{t-1}, \end{aligned} \tag{6}$$

where

$$\alpha_t^{k,i} = \frac{1}{N_t} \sum_{i'=1}^{N_t} \left(\langle \pi_t^{k,i'}, r_t^{i'} \rangle + \sum_{j \in \mathcal{J}^k} \theta_{t,j}^{k,i'} \alpha_{t+1,j} \right); \quad \beta_t^{k,i} = \frac{1}{N_t} \sum_{i'=1}^{N_t} \langle \pi_t^{k,i'}, -C_t^{i'} \rangle.$$

These backward pass calculations are similar to NBD, albeit for a smaller set of sample paths $\widehat{N}$, as opposed to N. Once the set of cuts is updated at the root stage, an iteration of SDDP is complete.

2.2 Cutting-plane and Partial-sampling

The cutting-plane and partial-sampling algorithm was proposed in Chen and Powell (1999). This algorithm is an NBD-type algorithm, in the sense that an outer linearizations of the expected cost-to-go function is built. The algorithm also shares a few critical algorithm designs with the SD method.

An iteration of CUPPS begins by solving an approximate problem of the form in (4) at the root stage. Following that a single sample path is randomly generated, denoted by $(\xi_1^k, \xi_2^k, \ldots, \xi_T^k)$. Approximate problems (4) with x_{t-1}^k (optimal solution of the previous stage approximate problem) and ξ_t^k as input are solved for all $t = 1, \ldots, T$. These steps of the algorithm can be viewed as forward pass of the SDDP algorithm carried out with $\widehat{N} = 1$.

The algorithm does not carry out an explicit backward pass, but the optimal dual solutions obtained during the forward pass are used in cut generation. Let $\pi_t^{k,k}$ denote the optimal dual solution to the subproblem corresponding to the observation along the current sample path, i.e., ω_t^k. This dual vector is added to a collection of dual vectors encountered by the algorithm thus far, this collection is denoted by $\mathcal{V}_t^k$. Notice that the dual of the stage approximate problem, given by

$$\left\{ \left(\pi_t, (\theta_{t,j})_{j \in \mathcal{J}_t^{k-1}}\right) \;\middle|\; D_t \pi_t - \sum_{j \in \mathcal{J}_t^{k-1}} \theta_{t,j} \beta_{t,j} = d_t,\ \sum_{j \in \mathcal{J}_t^{k-1}} \theta_{t,j} = 1,\ \theta_{t,j} \geq 0,\ \forall j \in \mathcal{J}_t^{k-1} \right\},$$

does not depend on $\tilde{\xi}_t$. Therefore, all elements of the set $\mathcal{V}_t^k$ are feasible for every observation of $\tilde{\xi}_t$. Using this fact, for observations $\Omega_t \setminus \{\omega_t^k\}$, a dual vector is identified using the following "argmax" procedure:

$$(\pi_t^{k,i}, \theta_t^{k,i}) \in \arg\max \ \{\langle \pi_t, r_t^i - C_t^i x_t^k \rangle + \sum_{j \in \mathcal{J}_t^{k-1}} \theta_{t,j} \alpha_{t,j}\} \tag{7}$$

The above dual vector provides the best possible lower bounding approximation of the cost-to-go-function. The dual solutions identified above are used to compute a cut as shown on the right-hand side of (6). While $\widehat{N}_t$ cuts are generated in SDDP, only one cut is generated in every iteration of the CUPPS algorithm. The feature of carrying out calculation only along a single sample path and using the argmax operation to inexactly compute the cuts is common with the SD method.

2.3 Abridged Nested Benders Decomposition

The abridged nested decomposition (AND) algorithm was proposed by Donohue and Birge (2006). In AND, a first-stage approximate problem is solved at the beginning of each iteration, a feature common to all the NBD-type algorithms. Like SDDP, AND involves multiple ($\widehat{N}$) randomly generated sample paths in the forward pass. However, the forward pass does not proceed along all the generated sample paths; herein lies the distinguishing feature. This algorithm is motivated by the fact that multiple sample paths may yield similar solutions. Similar solutions result in cuts that are not significantly different and do not contribute much to improving the approximation. The cut-generation procedure of AND is described next.

After solving the first-stage approximate problem, a random subset of size $\widehat{N}_1$ of the second-stage observations is generated. With the first-stage solution x_0^k and ξ_1^i as input, approximate second-stage problems is solved to build a pool of second-stage solutions $\{x_1^k\}$. Using this

solution pool, a further subset, say of size $\widehat{M}_1 \leq \widehat{N}_1$, is identified to proceed to the next stage. The solutions selected to proceed forward are known as the branching values. A branching value can either be a solution from the collection $\{x_1^k\}$ or some combination of these solutions. This approach is justified by the fact that the subsequent stage problem remains feasible, under relatively complete recourse assumption, for any combination of the current solutions. That is, for any selection of parameter vector $(w_t^i)_{\forall i}$ that satisfy $\sum_{i=1}^{N_t} w_t^i = 1$ and $0 \leq w_t^i \leq 1$ $\forall i = 1, \ldots, N_t$, the solution $\bar{x}_t = \sum_{i=1}^{N_t} w_t^i x_t^i \in \mathcal{X}_t$ results in non-empty feasible region for all ω_{t+1}. Therefore, relatively complete recourse ensures that $H_{t+1}(\bar{x}_t) < \infty$. The same procedure is adopted at all non-terminal stages of the problem to complete the forward pass.

Similar to SDDP, the cut coefficient calculations at stage t involve solving subproblems corresponding to all possible realizations in stage $t+1$ (as in (6)). However, the calculations on the backward pass are carried out only along the branching values. Since cuts are calculated using only the branching values, $\widehat{M}_t$ cuts are added in stage t in every iteration. Therefore, the computational effort in both the forward and backward passes of AND is lower than the SDDP algorithm.

Motivated by the above algorithms, there have been significant research efforts on randomization-based algorithms for MSLP. The use of randomization in the backward pass, a feature from the SD and CUPPS algorithms, was incorporated within SDDP by Linowsky and Philpott (2005). Later, Philpott and Guan (2008) provides the almost sure convergence of randomization-based NBD-type algorithms. They establish that SDDP can identify the optimal solution to the MSLP with finite support (including solutions to an SAA of MSLP problem) in a finite number of iterations. Further, when randomization is performed in the backward pass, it must be performed independently of the forward pass randomization. It must be done such that all observations in a stage are sampled infinitely often, with probability one. Shapiro (2011) discuss the statistical properties and analyze the SDDP solution of the SAA problem and its relation to the optimal solution of the true MSLP problem.

Despite the popularity of these randomization-based methods, notably the SDDP algorithm, they are known to exhibit slow convergence. The reader is referred to Lan (2020) for a study on the complexity analysis of the SDDP algorithm. Furthermore, the approximate stage problem size increases linearly with the number of iterations (due to the inclusion of cuts). Quadratic regularization has proved to be an effective technique to address this issue. Motivated by their success in the two-stage setting, Asamov and Powell (2018) presents a regularized variant of the SDDP algorithm. The capabilities of the SDDP algorithm were further extended by the inclusion of the risk measures in the objective (Philpott and de Matos (2012)) and the ability to solve MSLP problems with binary state variables (Zou et al. (2019)). The NBD algorithm's randomized variants remain a fruitful area of active research with many open questions. Nevertheless, it is important to recognize that the SDDP, CUPPS, and AND algorithms use randomization to solve a deterministic optimization problem.

3 Stochastic Decomposition Methods

Unlike the deterministic decomposition methods described above that work with a finite representation of the uncertainty, stochastic decomposition (SD) methods rely upon sequential or internal sampling. These methods dynamically update uncertainty representation by incorporating new observations of the underlying stochastic process. Therefore, the successive approximations generated in SD methods are based on SAA functions that use an ever-increasing set of sample paths.

The SD method was first introduced for two-stage SP problems in Higle and Sen (1991). It was later extended to include quadratic regularization in Higle and Sen (1994), which enabled its application in several large-scale problems. The reader is referred to related chapters in the encyclopedia for details regarding the two-stage SD algorithm. The SD methods offer several salient features that are identified below.

- The SD methods do not require prior discretization and can support problems with continuous support.
- These methods do not require knowledge of the probability distribution associated with the underlying stochastic process but only need a mechanism to simulate sample paths. In this regard, one can use external simulators as a source of sample paths.
- Furthermore, incorporating new sample paths on the fly can support online optimization with streaming data.

The successes in two-stage settings motivated the development of the multistage SD algorithm (Sen and Zhou (2014)). This method accommodates general stochastic processes that exhibit correlation across stages using a nodal formulation of MSLP. This chapter presents stochastic dynamic linear programming, a counterpart of multistage SD designed for MSLP problems with stagewise independence.

3.1 Stochastic Dynamic Linear Programming

The stochastic dynamic linear programming (SDLP) algorithm was proposed in Gangammanavar and Sen (2021). As in many multistage algorithms, an iteration of the SDLP algorithm includes forward and backward recursion steps. However, unlike the other MSLP methods, the SDLP forward and backward recursion computations for a given iteration are carried out along a single new sample path. This sample path is generated independently from the previously encountered sample paths. This sample path is denoted by $\{\xi_t^k\}_{t=1}^T$

The forward recursion involves stage decision simulation at each stage along the observed sample path. The forward recursion is carried out in two passes. In a prediction pass, a solution mapping is used to identify an incumbent solution (prox-center) for each nonterminal stage. This incumbent solution trajectory is denoted by $\{\hat{x}_t^k\}_{t=0}^{T-1}$. Using a incumbent solution, a regularized stage approximate problem is solved for all $t \in \mathcal{T} \setminus \{T\}$ given by

$$\begin{aligned} \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t^k)} \quad & \langle d_t, x_t \rangle + \eta_t + \frac{\sigma}{2}\|x_t - \hat{x}_t^k\|^2 \\ \text{subject to } & \langle -\beta_{t,j}, x_t \rangle + \eta_t \geq \alpha_{t,j} \qquad \forall j \in \mathcal{J}_t^{k-1}, \end{aligned} \tag{8}$$

where $\sigma \geq 1$ is a given proximal parameter. This step results in a trajectory of candidate solutions $\{x_t^k\}_0^{T-1}$. The solution mapping used to identify the incumbent solution is not discussed here and the focus is only on the approximation generating procedure of SDLP.

While the role of the backward recursion remains the same, namely to update the piecewise affine convex outer linearization-based approximations, these approximations are built for a sample mean function using only the sample paths encountered until the current iteration of the algorithm. Furthermore, the ability to share the approximation across all observations in a given stage remains valid even when sequential sampling in employed. However, additional

steps need to be incorporated to handle the dynamically evolving nature of the sample mean functions. In iteration k, the stage t sample mean takes the following form:

$$\widehat{H}_t^k(x_{t-1}) = \frac{1}{N_t^k} \sum_{i=1}^{N_t^k} \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t^i)} \{\langle d_t, x_t \rangle + \eta_t \mid \langle -\beta_{t,j}, x_t \rangle + \eta_t \geq \alpha_{t,j},\ \forall j \in \mathcal{J}_t^k\} \tag{9}$$

where, N_t^k denotes the current number of observations in stage t. Contrast the above sample mean to the SAA function in (3).

To generate the lower bounding affine function for the above sample mean, the minimization problem corresponding to ξ_t^k on the right-hand side of (9) with x_t^k as input is solved. The optimal dual solutions are denoted by π_t^k and $(\theta_{t,j}^k)_{j \in \mathcal{J}^k}$, the latter correspond to the current collection of affine lower bounding functions. For all other ξ_i (where $i \neq k$), a slight modification (detailed later in this section) of the argmax operation in (7) is used to identify the dual vertex that provides the best lower bound. These dual vertices are denoted by π_t^i and $(\theta_t^i t, j)_{j \in \mathcal{J}^k}$. Using the collection of the dual vertices, the coefficients are computed as

$$\alpha_t^k = \frac{1}{N_t^k} \sum_{i=1}^{N_t^k} \langle \pi_t^i, r_t^i \rangle; \qquad \beta_t^k = \frac{1}{N_t^k} \sum_{i=1}^{N_t^k} \left(\langle \pi_t^i, -C_t^i \rangle + \sum_{j \in \mathcal{J}_{t+1}^k} \theta_{t,j}^i \alpha_{t+1,j} \right).$$

The resulting affine function lower bounds the sample mean in (9), that is, it satisfies $\ell_t^k = \alpha_t^k + \langle \beta_t^k, x_{t-1} \rangle \leq \widehat{H}_t^k(x_{t-1})$ for all x_{t-1}. It is worthwhile to note that the above coefficients are stochastic in nature. Therefore, the affine functions are viewed as minorants of the sample mean rather than cuts (a term more appropriate for deterministic decomposition). Similar calculations using the incumbent solution $\hat{x}_t^k$ results in an incumbent minorant $\hat{\ell}_t^k(x_t) = \hat{\alpha}_t^k + \langle \hat{\beta}_t^k, x_{t-1} \rangle$.

Since new sample paths are introduced in every iteration of SDLP, additional steps need to be taken to ensure that the minorants computed in any iteration remain valid in later iterations. The minorants are termed as valid if they continue to provide a lower bound on the sample mean available in that iteration. To ensure validity in stage t, the minorant generated in an earlier iteration (say $j < k$) are scaled down by a factor $(j/k)^{T-t}$. Notice that the scaling operation can be perfomed recursively. The resulting stage approximation takes the following form:

$$h_t^k(x_{t-1}) = \max \left\{ \left\{ \left(\frac{k-1}{k} \right)^{T-t} \ell_t^j(x_{t-1}) \right\}_{j \in \mathcal{J}_{t-1}^{k-1}},\ \ell_t^k(x_{t-1}),\ \hat{\ell}_t^k(x_{t-1}) \right\}.$$

In other words, the updated collection of minorants $\mathcal{J}_{t-1}^k$ used to approximate the value function h_t include the scaled minorants in $\mathcal{J}_{t-1}^{k-1}$, a candidate minorant computed using x_t^k, and an incumbent minorant computed using $\hat{x}_t^k$. Theorems 2 and 3 in Gangammanavar and Sen (2021) capture the behavior of these approximations across stages and iterations, respectively. These results establish the validity of approximations to the sample mean with increasing sample size and show uniform asymptotic convergence of these approximations. While any update scheme that consistently maintains the validity of the approximation h_k^t to the current sample mean is admissible, the particular choice presented above is recommended with an eye on its implementational ease.

4 Discussion

This chapter illustrated two classes of multistage SP algorithms that rely upon randomization. The first class of algorithms is applicable to solve deterministic optimization problems, including

the SAA of MSLP problems. The randomization-based deterministic algorithms explicitly use the probability distribution (or empirical distribution) information in constructing their approximate cost-to-go functions (see (6)). In contrast, the second class of algorithms is inherently random because they work with a dynamic representation of the underlying uncertainty. They do not require explicit knowledge of probability distribution but need the ability to generate new sample paths on the fly. In this regard, stochastic decomposition is an online optimization approach that can work with streaming data.

The convergence results and the analysis techniques employed to achieve them differ in these two classes of algorithms. These differences are due to the discrepancies in the role of sampling. As in the case of NBD, the randomization-based decomposition methods can ensure convergence to the optimal solution of the deterministic problem in a finite number of iterations. This result is achieved under mild conditions on the sampling techniques used in the forward and backward passes. The reader is referred to Philpott and Guan (2008) for a detailed discussion on these sampling conditions.

On the other hand, the SDLP algorithm provides a sequence of decisions and corresponding value function estimates that asymptotically converge with probability one. The SDLP solutions and optimal value are for the original expectation-valued MSLP problem. Quadratic regularization involving proximal centers identified using the basic feasible policy plays a critical role in convergence analysis. The reader is referred to Gangammanavar and Sen (2021) for a thorough exposition of the basic feasible policy and the convergence results of SDLP.

Whether randomization-based deterministic or stochastic decomposition is adopted to solve, it is essential to recognize that one deals with random approximations of the actual MSLP problem. These algorithms' solutions and values are random estimates of their true counterparts. Therefore, a solution from a single run of these algorithms may mislead the decision-maker. To address this, variance reduction techniques such as using multiple replications Mak et al. (1999) and compromise policies Sen and Liu (2016) are viable approaches. Along with these, developing statistical validation of optimality and stopping rules for multistage algorithms remains a fruitful research direction.

References

Asamov, T. and Powell, W. (2018). Regularized decomposition of high-dimensional multistage stochastic programs with markov uncertainty. *SIAM Journal on Optimization*, 28(1):575–595.

Birge, J. R. (1985). Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, 33(5):989–1007.

Chen, Z. and Powell, W. (1999). Convergent cutting-plane and partial-sampling algorithm for multistage stochastic linear programs with recourse. *Journal of Optimization Theory and Applications*, 102(3):497–524.

Donohue, C. and Birge, J. (2006). The abridged nested decomposition method for multistage stochastic linear programs with relatively complete recourse. *Algorithmic Operations Research*, 1(1).

Gangammanavar, H. and Sen, S. (2021). Stochastic dynamic linear programming: A sequential sampling algorithm for multistage stochastic linear programming. *SIAM Journal on Optimization*, 31(3):2111–2140.

Higle, J. L. and Sen, S. (1991). Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of Operations Research*, 16(3):650–669.

Higle, J. L. and Sen, S. (1994). Finite master programs in regularized stochastic decomposition. *Mathematical Programming*, 67(1-3):143–168.

Infanger, G. and Morton, D. P. (1996). Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75(2):241–256.

Lan, G. (2020). Complexity of stochastic dual dynamic programming. *Mathematical Programming*, pages 1–38.

Linowsky, K. and Philpott, A. (2005). On the convergence of sampling-based decomposition algorithms for multistage stochastic programs. *Journal of Optimization Theory and Applications*, 125(2):349–366.

Mak, W., Morton, D. P., and Wood, K. (1999). Monte carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1):47 – 56.

Mulvey, J. M. and Ruszczyński, A. (1995). A new scenario decomposition method for large-scale stochastic optimization. *Operations Research*, 43(3):477–490.

Pereira, M. and Pinto, L. (1991). Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52(1-3):359–375.

Philpott, A. B. and de Matos, V. L. (2012). Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218(2):470 – 483.

Philpott, A. B. and Guan, Z. (2008). On the convergence of stochastic dual dynamic programming and related methods. *Operations Research Letters*, 36(4):450 – 455.

Rockafellar, R. T. and Wets, R. J. B. (1991). Scenarios and policy aggregation in optimization under uncertainty. *Math. Oper. Res.*, 16(1):119–147.

Sen, S. and Liu, Y. (2016). Mitigating uncertainty via compromise decisions in two-stage stochastic linear programming: Variance reduction. *Operations Research*, 64(6):1422–1437.

Sen, S. and Zhou, Z. (2014). Multistage Stochastic Decomposition: A bridge between Stochastic Programming and Approximate Dynamic Programming. *SIAM Journal on Optimization*, 24(1):127–153.

Shapiro, A. (2011). Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, 209(1):63 – 72.

Shapiro, A., Dentcheva, D., and Ruszczynski, A. (2014). *Lectures on Stochastic Programming: Modeling and Theory, Second Edition*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Van Slyke, R. M. and Wets, R. J. B. (1969). L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663.

Zou, J., Ahmed, S., and Sun, X. A. (2019). Stochastic dual dynamic integer programming. *Mathematical Programming*, 175(1):461–502.