Importance Sampling in Stochastic Programming: A Markov Chain Monte Carlo Approach

Panos Parpas
Department of Computer Science
Imperial College London, South Kensington, SW7 2AZ
p.parpas@imperial.ac.uk

Berk Ustun
Computation for Design and Optimization
Massachusetts Institute of Technology, Cambridge, MA 02139
ustunb@mit.edu

Mort Webster
Engineering Systems Division
Massachusetts Institute of Technology, Cambridge, MA 02139
mort@mit.edu

Stochastic programming models are large-scale optimization problems that are used to facilitate decision-making under uncertainty. Optimization algorithms for such problems need to evaluate the expected future costs of current decisions, often referred to as the recourse function. In practice, this calculation is computationally difficult as it requires the evaluation of a multidimensional integral whose integrand is an optimization problem. In turn, the recourse function has to be estimated using scenario trees or Monte Carlo methods. Unfortunately, scenario trees do not scale well for problems with many dimensions of uncertainty and many stages, and Monte Carlo methods require very large numbers of samples. We introduce a novel importance sampling framework for multistage stochastic programming that can produce accurate estimates of the recourse function using a fixed number of samples. Previous approaches for importance sampling in stochastic programming were limited to problems where the uncertainty was modeled using discrete random variables, and the recourse function was additively separable in the uncertain dimensions. Our framework avoids these restrictions by using Markov Chain Monte Carlo and Kernel Density Estimation algorithms to create a non-parametric importance sampling distribution that can form lower variance estimates of the recourse function. We demonstrate the increased accuracy and efficiency of our approach in the context of multistage stochastic programming using variants of the Newsvendor problem. Our numerical results show that our framework produces more accurate estimates of the optimal value and solution of stochastic programming models, especially for problems with moderate to high variance, multimodal or rare-event distributions.

Key words: Benders' Decomposition, Importance Sampling, Markov Chain Monte Carlo, Stochastic Programming, Variance Reduction
History: Last Edited On: July 19, 2012

1. Introduction

Stochastic programming models are large-scale optimization problems that are used to facilitate decision-making under uncertainty. Optimization algorithms for such problems require the evaluation of the expected future costs of current decisions, often referred to as the recourse function. In practice, this calculation is computationally difficult as it requires the evaluation of a multidimen-

sional integral whose integrand is an optimization problem. Many algorithms approximate the value of recourse function using quadrature rules (see Pennanen and Koivu (2005)) or Monte Carlo (MC) methods (see Birge and Louveaux (2011) and Shapiro et al. (2009)). MC methods are particularly appealing in such cases because they are easy to implement and remain computationally tractable when the recourse function depends on a large number of random variables. Nevertheless, the sampling error in MC estimates can significantly alter the results of a stochastic programming model. Although we can reduce the sampling error in MC estimates of the recourse function by using more samples in the MC procedure, this approach is not computationally tractable in stochastic programming where an optimization problem has to be solved for each sample. As a result, MC methods need to be paired with a variance reduction technique that can produce estimates with lower sampling error for a fixed number of samples.

In this paper, we focus on a variance reduction technique known as importance sampling that aims to reduce the sampling error of MC estimates by generating samples from regions that contribute most to the value of the recourse function. Importance sampling requires an importance sampling distribution which is usually chosen to exploit prior knowledge about the randomness in the underlying model. Even though many distributions can be used in importance sampling, there exists an importance sampling distribution that is optimal in the sense that it can produce MC estimates with zero variance (see e.g. Asmussen and Glynn (2007)). The so-called zero-variance distribution cannot be used in practice, but it is often used to guide the design of effective importance sampling distributions. Importance sampling was first applied to stochastic programming in a series of papers by Dantzig and Glynn (1990) and Infanger (1992). The importance sampling distribution in these papers showed promising results as it was derived from the zero-variance distribution. Unfortunately, the distribution was developed under the assumptions that the uncertainty is modeled using discrete random variables, and that the cost surface is additively separable in the random dimensions. These assumptions can be restrictive in practice.

The primary contribution of this paper is an importance sampling framework that does not require such assumptions. Our framework, which we refer to as the Markov Chain Monte Carlo Importance Sampling (MCMC-IS) framework, exploits the fact that the zero-variance distribution is known up to a normalizing constant. It first uses a Markov Chain Monte Carlo (MCMC) algorithm to generate samples from the zero-variance distribution, and then uses a Kernel Density Estimation (KDE) algorithm to reconstruct an approximate zero-variance distribution from these samples. With this approximate zero-variance distribution at hand, we are able to generate a new, larger set of samples and construct a lower variance importance sampling estimate of the recourse function. MCMC-IS is flexible, in that it can accommodate a wide array of MCMC and KDE algorithms; non-parametric, in that it does not require users to specify a family of distributions; robust, in that it can generate good results for probability distributions that are difficult to work with using existing methods; and well-suited for stochastic programming, in that it produces lower variance estimates that improve the performance of statistical tests used to assess convergence in stopping procedures. Although both MCMC and KDE algorithms have received considerable attention in the literature, they have not, to our knowledge, been combined in this way before.

Importance sampling is just one of many variance reduction techniques that can be used in stochastic programming algorithms. Quasi-Monte Carlo (QMC) methods were studied in Koivu (2005) and in Drew and Homem-de Mello (2006). The non i.i.d. case of MC sampling has been studied in Homem-de Mello (2006). Control variates were proposed in Shapiro and Homem-de Mello (1998) and in Higle (1998). A sequential sampling algorithm was proposed in Bayraksan and Morton (2011). A computational assessment of conditional sampling, antithetic sampling, control variates and importance sampling appeared in Higle (1998). QMC and Latin Hypercube Sampling (LHS) were compared in Homem-de Mello et al. (2011). The effect of sampling to the solution quality of stochastic programming problems was discussed in Linderoth et al. (2006).

Using numerical experiments, we show that our framework performs well when compared to Crude Monte Carlo (CMC) methods, Quasi-Monte Carlo (QMC) methods and the importance sampling technique developed in Dantzig and Glynn (1990) and Infanger (1992) (DGI). In addition, we show that our framework significantly outperforms the existing sampling methods when the uncertainty is modeled using a higher variance, rare-event or multi-modal distribution.

MC methods need to be paired with optimization algorithms in order to solve stochastic programming problems. In turn, we illustrate the computational performance of the MCMC-IS framework by embedding it in the Stochastic Dual Dynamic Programming (SDDP) algorithm developed by Pereira and Pinto (1991). However, we stress that MCMC-IS can be paired with many other optimization algorithms, such as sample average approximation method (see Shapiro et al. (2009)), stochastic decomposition (see Higle and Sen (1991)), progressive hedging (Rockafellar and Wets (1991)), augmented Lagrangian methods (see Parpas and Rustem (2007)), other variants of Benders' decomposition (see Birge and Louveaux (2011)) or even used in Approximate Dynamic Programming algorithms Powell (2007). More generally, we expect MCMC-IS to yield similar benefits in other areas than expected value optimization such as sampling approaches for developing stopping rules (Morton (1998)), chance-constrained programming (see e.g. Watson et al. (2010)), and risk-averse stochastic programming (see Shapiro (2009)).

Our paper is structured as follows: in Section 2, we provide a brief overview of multistage stochastic programming models, and illustrate the mechanism through which decomposition algorithms can produce inaccurate estimates of the optimal value and solution of a multistage stochastic program when they are paired with a MC method. In Section 3, we introduce the MCMC-IS framework. Section 4 illustrates the properties of the MCMC-IS framework using numerical experiments based on a Newsvendor model. Section 5 demonstrates the benefits of the MCMC-IS framework when it is paired with the SDDP algorithm to solve stochastic programming problems using additional numerical experiments. We summarize our contributions and discuss future research in Section 6.

2. Motivation

We consider a multistage linear stochastic programming model defined as,

$$\begin{aligned} z^* = \min_{x_1} \quad & c_1^{\mathrm{T}} x_1 + \mathcal{Q}_1(x_1) \\ \text{s.t. } & A_1 x_1 = b_1, \\ & x_1 \geq 0, \end{aligned} \tag{2.1}$$

where $c_1 \in \mathbb{R}^{n_1}$, $A_1 \in \mathbb{R}^{n_1 \times m_1}$ and $b_1 \in \mathbb{R}^{m_1}$. In general, the function $\mathcal{Q}$ is called the recourse function, and is used to represent the expected future costs of current decisions,

$$\mathcal{Q}_t(x_t) = \mathbb{E}[Q_t(x_t, \xi_{t+1})], \quad t = 1, \ldots, T-1. \tag{2.2}$$

Given a fixed decision in the previous stage and a realization of the random parameters, the future costs of the model can be estimated by solving the linear program,

$$\begin{aligned} Q_{t-1}(\widehat{x}_{t-1}, \xi_t) = \min_{x_t} \quad & c_t^{\mathrm{T}}(\xi_t) x_t + \mathcal{Q}_t(x_t) \\ \text{s.t. } & A_t(\xi_t) x_t = b_t(\xi_t) - W_t(\xi_t) \widehat{x}_{t-1}, \\ & x_t \geq 0, \end{aligned} \tag{2.3}$$

where $Q_T(\widehat{x}_{T-1}, \xi_) \equiv 0$ without loss of generality. We will assume that $c_t \in \mathbb{R}^{n_t}$, $A_t \in \mathbb{R}^{n_t \times m_t}$, $W_t \in \mathbb{R}^{n_{t-1} \times m_t}$, $b_t \in \mathbb{R}^{m_t \times 1}$. The components of these parameters are deterministic for $t = 1$, but may

be random for $t = 2, \ldots, T$. We refer to the set of all random components of the parameters at stage t using a D_t-dimensional random vector ξ_t, and denote its joint probability density function, cumulative distribution function and support as f_t, F_t and Ξ_t respectively. We refer the interested reader to Birge and Louveaux (2011) for an overview of multistage stochastic programming.

Many algorithms have been developed to solve multistage stochastic programming problems. A key step in these algorithms is the discretization of the random parameters. An alternative to discretization is to generate samples of the random parameters, and use an MC method to estimate the recourse function. Such an approach is advantageous in that it can accommodate discrete or continuous random variables, remain computationally tractable for models with a large number of random variables, and produce estimates of the recourse function whose error does not depend on the number of random variables used in the model. Nevertheless, the error of these estimates can significantly alter the results of a stochastic programming model. In the next section, we explain how MC methods can be used in decomposition algorithms, and demonstrate how the sampling error of MC estimates can produce inaccurate estimates of the optimal value and solution of a multistage stochastic program.

2.1. The Perils of Sampling in Decomposition Algorithms

Decomposition algorithms are designed to solve multistage stochastic programming problems by constructing a piecewise linear approximation of the epigraph of the recourse function (see Birge and Louveaux (2011)). The approximation is composed of supporting hyperplanes to the recourse function at fixed values of x, which we denote as $\widehat{x}$ throughout this paper. The supporting hyperplanes are also known as cuts. Given a fixed value $\widehat{x}$, a cut takes the form of a linear inequality constraint,

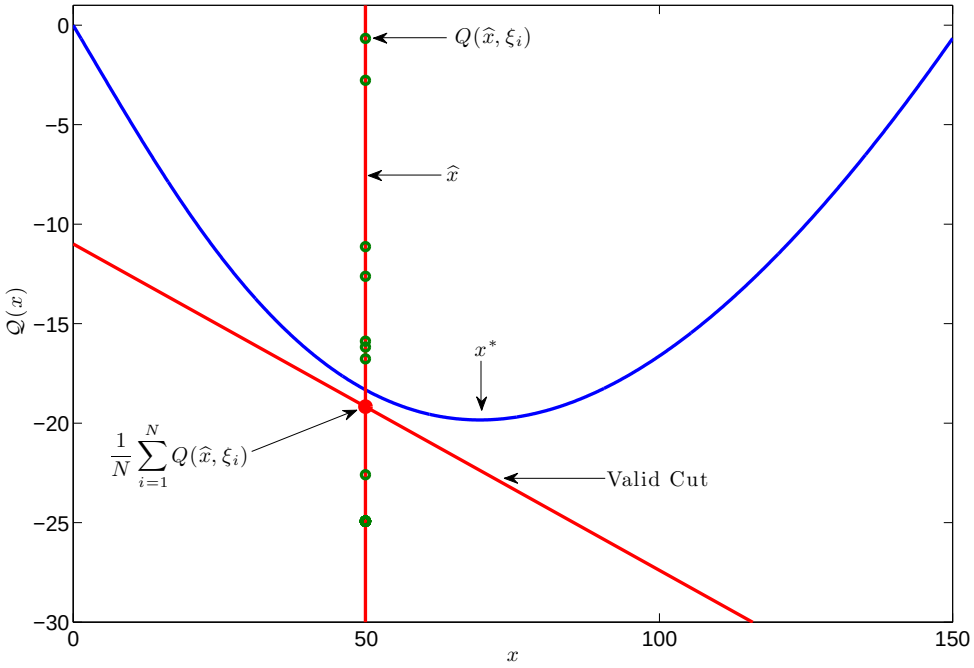
$$\mathcal{Q}_t(x_t) \geq \mathcal{Q}_t(\widehat{x}_t) + \partial\mathcal{Q}_t(\widehat{x}_t)(x_t - \widehat{x}), \tag{2.4}$$

where $\partial\mathcal{Q}_t$ represents the subgradient of the recourse function. We note that the parameters $\mathcal{Q}_t$ and $\partial\mathcal{Q}_t$ are the expected values of the optimal objective value and dual variables of the linear program in (2.3). The preceding inequality assumes that these parameters can be calculated exactly. In practice, this can only be achieved when the random variables in the model have a limited set of outcomes, or, as is the case in this paper, the expectations are replaced with their MC estimates,

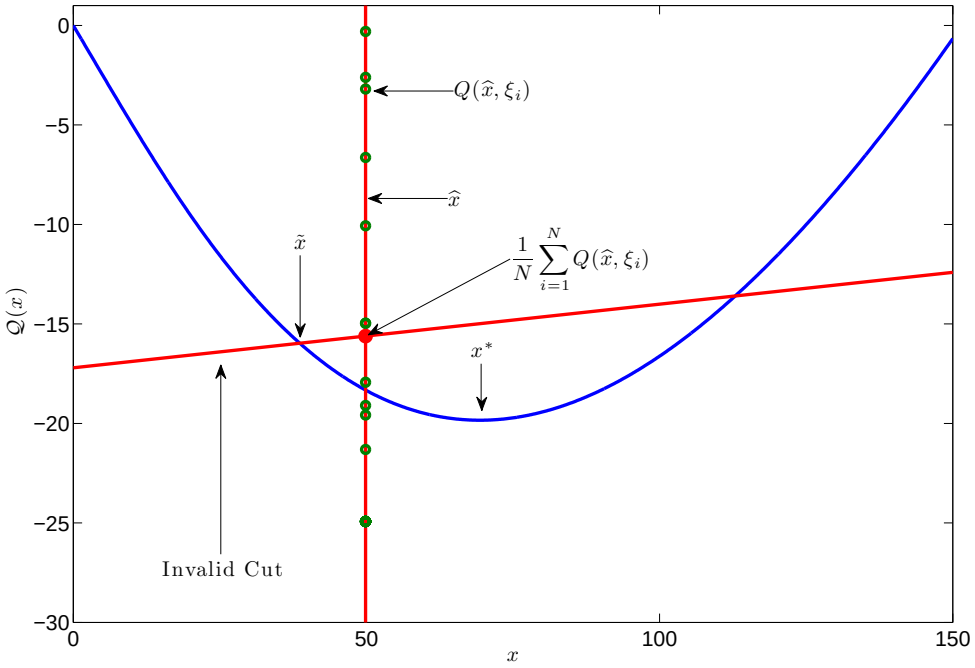
$$\begin{aligned} \widehat{\mathcal{Q}}_t^{MC}(\widehat{x}_t) &= \frac{1}{N}\sum_{i=1}^{N} Q_t(\widehat{x}_t, \xi_i), \\ \widehat{\partial\mathcal{Q}}_t^{MC}(\widehat{x}_t) &= \frac{1}{N}\sum_{i=1}^{N} \partial Q_t(\widehat{x}_t, \xi_i). \end{aligned} \tag{2.5}$$

Although MC methods can significantly reduce the computational burden in generating cuts relative to a scenario tree based approach, the cuts generated with MC methods are subject to sampling error. Even if the sampling error associated with each cut is negligible, the errors can compound across the iterations of a decomposition algorithm. As a result, decomposition algorithms that use a small number of samples may produce an invalid approximation of the recourse function that leads to inaccurate results for the original problem. We illustrate this phenomenon in Figure 1, where we plot sampled cuts that are produced when a CMC method paired with a decomposition algorithm in order to solve a simple two-stage Newsvendor model, whose parameters are specified in Section 4.1.

Both cuts in this example were constructed using $N = 50$ samples. For clarity, we plot a subset of the sample values $Q(\widehat{x}, \xi_i), i = 1, \ldots, N$ along the vertical line of $\widehat{x}$, as well as their sample average. In Figure 1(a), we are able to generate a valid sampled cut, which is valid because it underestimates the true recourse function $\mathcal{Q}(x)$ at all values of x. However, it is possible to generate a sampled cut



(a)



(b)

Figure 1 In 1(a) the sampled cut is valid; assuming that only valid cuts are generated in subsequent iterations, a decomposition algorithm will produce accurate estimates of x^* and z^*. In 1(b) the sampled cut is invalid; even if all the other cuts produced by the algorithm are valid, the true optimal solution at x^* will remain infeasible, and a decomposition algorithm will produce high-error estimates of x^* and z^*.

that in some regions overestimates, and in other regions underestimates the true recourse function $\mathcal{Q}(x)$. We illustrate this situation in Figure 1(b), where the sampled cut excludes the true optimal solution at $x^* \approx 69$ with $z^* \approx -20$. Assuming that the algorithm only generates valid cuts until the algorithm converges, the resulting estimates of x^* and z^* will be $\tilde{x} \approx 38$ and $\tilde{z} \approx -15$, corresponding to errors of 80% and 25% respectively. We note that the optimal solution x^* corresponds to the value of x that minimizes the sum of the first-stage costs and the recourse function, and not the value of x that minimizes the recourse function (although these values appear to be very close to each other in Figure 1).

It is true that we can entirely avoid generating invalid sampled cuts if we model the uncertainty in the problem using a scenario tree. While this approach allows us to calculate the exact values of the parameters in (2.4), it suffers from a different complication. Scenario trees are discrete in nature, and therefore require models that use discrete random variables or a suitable discretization procedure to represent continuous random variables using a finite outcomes and probabilities. In the latter case the scenarios are fixed and the parameters in (2.4) are easy to calculate. However, there are no guarantees that the solution obtained with the discretized scenario tree will be optimal for the original continuous problem unless a large number of scenarios is used. Even though scenario trees can yield accurate answers for stochastic programming problems with few random variables and time periods, they also present computational challenges for large-scale problems with multiple random variables and time periods. In such cases, scenario trees impose an unnecessary choice between high-resolution discrete approximations that yield accurate solutions but are difficult to store and solve, and low-resolution discrete approximations that may yield inaccurate solutions but are easier to store and solve.

3. The Markov Chain Monte Carlo Approach to Importance Sampling

It is well-known that we can reduce the sampling error in the cut parameters if we increase the number of samples that we use to construct their MC estimates. Even so, the $O(N^{-0.5})$ convergence rate of MC methods effectively implies that we have to solve four times as many linear programs in order to halve the sampling error of the cut parameters. Given the time that is required to solve a typical linear program within a large-scale multistage stochastic programming model, such an approach is simply not tractable. Fortunately, the sampling error of the cut parameters depends on $\frac{\sigma}{N}$ where σ denotes the variance of the estimate. As a result, an alternative way to reduce the sampling error in the cut parameters without increasing the number of samples is to reduce the underlying variance of the quantity that we are trying to estimate. Importance sampling is a variance reduction technique that can produce an estimate of $\mathcal{Q}$ that has lower variance, and consequently lower sampling error, than $\widehat{\mathcal{Q}}^{MC}$ in (2.5). The variance reduction is achieved by using a different probability distribution that can generate samples at regions that contribute the most to $\mathcal{Q}$. Although importance sampling estimates can have substantially lower variance than their MC counterparts, choosing a suitable importance sampling distribution is a challenging process that is difficult to generalize and has motivated many papers in the statistics and simulation literature. We refer the interested reader to Asmussen and Glynn (2007) for a review of importance sampling.

3.1. Importance Sampling and the Curse of Circularity

Importance sampling is a variance reduction technique that constructs lower variance estimates using an importance sampling distribution g, as opposed to the original sampling distribution f. When samples are generated from the importance sampling distribution g, the recourse function can be calculated as

$$\mathcal{Q}(\widehat{x}) = \mathbb{E}^f[Q(\widehat{x}, \xi)] = \mathbb{E}^g[Q(\widehat{x}, \xi)\Lambda(\xi)]. \tag{3.1}$$

where

$$\Lambda(\xi) = \frac{f(\xi)}{g(\xi)}. \tag{3.2}$$

The function $\Lambda : \Xi \to \mathbb{R}$ is called the likelihood function, and is used to correct the bias introduced by the fact that we generated the samples from g instead of f. In theory, the only requirement for the importance sampling distribution g is that the likelihood function Λ has to be well-defined over the support of f. In other words, $g(\xi) > 0$ at all values of ξ where $f(\xi) > 0$.

Once we select a suitable important sampling distribution g, we can use it to generate a set of N i.i.d. samples $\xi_1 \dots \xi_N$ and construct an importance sampling estimate of the recourse function as

$$\widehat{\mathcal{Q}}^{IS}(\widehat{x}) = \frac{1}{N}\sum_{i=1}^{N} Q(\widehat{x}, \xi_i)\Lambda(\xi_i). \tag{3.3}$$

The benefit of generating samples from g depends on the amount of variance reduction that can be achieved. Importance sampling is most effective in the context of stochastic programming when g can generate samples from the regions that contribute the most to the value of the recourse function at a fixed point $\widehat{x}$. In fact, it is easy to show that the variance of an importance sampling estimate is minimized when we sample from,

$$g^*(\xi) = \frac{|Q(\widehat{x}, \xi)|}{\mathbb{E}^f |Q(\widehat{x}, \xi)|} f(\xi). \tag{3.4}$$

The importance sampling distribution g^* is optimal in the sense that no other distribution can produce an importance sampling estimate with lower variance (see e.g. Asmussen and Glynn (2007)). In fact, if $Q(x, \xi)$ is always positive then g^* produces estimates with zero variance, and is therefore usually referred to as the zero-variance distribution. The problem with using (3.4) in practice is that it requires knowledge of $\mathbb{E}^f |Q(x, \xi)|$, which is the quantity that we sought to compute in the first place. We are thus faced with a "curse of circularity" in that we can use (3.4) to construct zero-variance estimates if and only if we already have a zero-variance estimate of $\mathbb{E}^f |Q(\widehat{x}, \xi)|$.

The importance sampling framework that we introduce in this paper revolves around two key remarks. The first remark is that we can generate samples from (3.4) using an MCMC algorithm since we know the distribution up to a normalizing constant $\mathbb{E}^f |Q(x, \xi)|$. We note that we cannot use these samples to form a zero-variance importance sampling estimate because we need to evaluate the likelihood in of each sample as shown in (3.2). In this case, the likelihood of a given sample is given by

$$\Lambda^*(\xi) = \frac{\mathbb{E}^f |Q(x, \xi)|}{|Q(x, \xi)|}, \tag{3.5}$$

and it is also impossible to compute in practice as it depends on $\mathbb{E}^f |Q(x, \xi)|$. This leads us to the second remark: while we cannot use the samples to directly form an importance sampling estimate, we can use them to reconstruct an approximation of the zero-variance distribution using a KDE algorithm. With this approximate distribution in hand, we can generate a second, larger set of samples, evaluate the likelihood of each sample, and form a lower variance importance sampling estimate.

The careful reader may notice that in developing the zero-variance distribution, we have ignored the fact that the accuracy of decomposition algorithms not only revolves around accurate estimates of the recourse function $\mathcal{Q}(\widehat{x})$, but also depends on accurate estimates of the subgradient to the recourse function $\partial\mathcal{Q}(\widehat{x})$ as shown in 2.4. Fortunately, the convexity of the recourse function ensures that high values of $\mathcal{Q}(\widehat{x})$ also correspond to high values of $\partial\mathcal{Q}(\widehat{x})$ by duality. In turn, any importance sampling distribution that produces a lower variance estimate of $\mathcal{Q}(\widehat{x})$ will also generate a lower variance estimate of $\partial\mathcal{Q}(\widehat{x})$. This represents a significant computational advantage for our framework as it avoids the need to construct two separate importance sampling distributions.

3.2. Description of the MCMC-IS Framework

Our proposed framework consists of three steps: (1) generate samples from the zero-variance distribution using an MCMC algorithm, (2) reconstruct an approximate zero-variance distribution using a KDE algorithm, and (3) resample from the approximate zero-variance distribution to form a lower variance importance sampling estimate.

MCMC algorithms are an established set of MC methods that can generate samples from a density known up to a normalizing constant. In contrast to other MC methods, MCMC algorithms produce a sequence of serially correlated samples. This sequence forms a Markov Chain whose stationary distribution is the target density, given by (3.4) in our case. Although many different MCMC algorithms can be used within our framework, we restrict our focus to the Metropolis-Hastings algorithm because it is easy to implement, does not require the specification of many parameters, and does not depend on a restrictive set of assumptions. We refer the interested reader to Gelman et al. (2010) for more on the Metropolis-Hastings algorithm, and other MCMC algorithms that can be used within our framework.

The Metropolis-Hastings algorithm uses a simple accept-reject procedure in order to generate a Markov Chain that has (3.4) as its stationary distribution. In the k-th step, the algorithm generates a proposed state ζ_k using a proposal distribution whose density $q(\cdot \mid \xi_k)$ typically depends on the current state ξ_k. Together, the proposed state, the current state and the target density are used to evaluate an acceptance probability, $a(\xi_k, \zeta_k)$. The proposed state is accepted with probability $a(\xi_k, \zeta_k)$, in which case the Markov Chain transitions to the proposed state $\xi_{k+1} := \zeta_k$. Otherwise, the proposed state is rejected with probability $1 - a(\xi_k, \zeta_k)$, in which case the Markov Chain remains at its current state $\xi_{k+1} := \xi_k$.

In this paper, we use a special instance of the Metropolis-Hastings algorithm where new states are proposed using a random walk process. This implies that the proposed state ζ_k at each step of the Metropolis-Hastings algorithm is generated as

$$\zeta_k = \xi_k + v_k, \tag{3.6}$$

where v_k is a Gaussian random variable with mean 0 and covariance matrix Σ. In practice, the Metropolis-Hastings algorithm requires that Σ is specified beforehand. However, we can avoid specifying this parameter is we use the Adaptive Metropolis algorithm described in Haario et al. (2001). When states are proposed through a random walk process, the proposal distribution is symmetric and the acceptance probability can be expressed as,

$$a(\xi_k, \zeta_k) = \min\left\{ \frac{|Q(\widehat{x}, \zeta_k)| f(\zeta_k)}{|Q(\widehat{x}, \xi_k)| f(\xi_k)}, 1 \right\}. \tag{3.7}$$

It is well-known that a random walk Metropolis-Hastings algorithm is inefficient compared to other MCMC algorithms because it produces samples that are highly correlated. Nonetheless, the quality of numerical results we achieve with this simple and admittedly inefficient MCMC algorithm only reinforces the potential of our proposed framework.

Once a representative set of M samples have been generated from the zero-variance distribution specified in (3.4) using an MCMC algorithm, we can reconstruct an approximate zero-variance distribution from these samples using a KDE algorithm. KDE algorithms are established techniques that are used to reconstruct continuous probability distributions from a finite set of samples. We refer the interested reader to Devroye and Györfi (1985), Silverman (1986) and Scott (1992) for a detailed overview of these techniques. In this case, the kernel density estimator is a probability density function expressed as

$$\widehat{g}_M(\xi) = \frac{1}{M} \sum_{i=1}^{M} K_H(\xi, \xi_i), \tag{3.8}$$

where the function K_H is referred to as a kernel function, and $H \in \mathbb{R}^{D\times D}$ is its associated bandwidth matrix. In order to ensure that $\widehat{g}$ is a proper probability density function, we impose the following conditions on the kernel function,

$$\begin{aligned} K_H(\cdot,\cdot) &\geq 0, \\ \int_{\Xi} K_H(\xi,\cdot)d\xi &= 1. \end{aligned} \tag{3.9}$$

In addition, we assume that the kernel matrix is positive semidefinite, meaning that the matrix where $(i,j)^{th}$ entry given by $K_H(\xi_i,\xi_j),\ 1\leq i,j\leq M$ is positive semidefinite. These assumptions are required by most KDE algorithms, and are satisfied by the majority of kernels used in practice. A popular kernel, and the one that we use in this paper, is the Gaussian product kernel,

$$K_H(\xi,\xi_i) = \prod_{k=1}^{D} \frac{1}{\sqrt{2\pi}h_k} \exp\left(\frac{(\xi_k-\xi_{i,k})^2}{2h_k^2}\right), \tag{3.10}$$

The associated bandwidth matrix H for the Gaussian product kernel is a $D\times D$ diagonal matrix that contains the bandwidth parameters of each dimension $h_1,\ldots,h_D$ along its diagonal. In our implementation, we use a one-dimensional likelihood-based search to estimate the value of the bandwidth parameter h_k separately for each dimension k.

Using the approximate zero-variance distribution $\widehat{g}_M$, we can finally construct an importance sampling estimate of the recourse function by generating N additional samples from $\widehat{g}_M$. Although these samples will not originate from the true zero-variance distribution g^*, they can still be used to produce a lower variance importance sampling estimate provided that the KDE algorithm has constructed a $\widehat{g}_M$ is similar to g^*. Generating samples from $\widehat{g}_M$ is also beneficial in that the samples are independent and the kernel functions are easy to sample from. In practice, we construct $\widehat{g}_M$ using modest values of M and then construct $\widehat{\mathcal{Q}}^{IS}(\widehat{x})$ using large values of N. The computational burden of the MCMC step is a result of the accept-reject algorithm which typically requires more LP evaluations (proposed samples) than are used (accepted samples). The additional advantage of estimating and sampling the approximate importance sampling distribution is the relative efficiency of generating a larger number of samples. In addition, unlike the MCMC algorithm which is sequential, this latter step can be implemented on parallel architectures. We provide a formal description of this framework in Algorithm 1.

3.3. Ingredients of the Convergence Analysis

The MCMC-IS framework has two sources of error. The first source of error is due to the MCMC algorithm used to generate samples from the zero-variance distribution, and the second is due to the KDE algorithm used in the construction of the approximate zero-variance distribution. If the sampling algorithm is embedded within an optimization algorithm then there is also a third source of error, but in this section we focus on the sampling aspect. The main convergence condition for MCMC is that the Markov chain generated by the algorithm is irreducible and aperiodic. The irreducibility property essentially means that the chain can eventually reach any subset of the space from any state. The aperiodic condition means that the chain cannot return to a subset of the space in a predictable manner. Formal definitions of these properties can be found in Roberts and Rosenthal (2004). The first step in the convergence analysis is to show that these two conditions are satisfied. In the case of the SDDP algorithm, the MCMC algorithm will be used whenever a new sampled cut needs to be generated and therefore these two conditions will hold even if the problem does not have complete recourse. In Section 4.2 and Section 5 we present numerical results that suggest that a modest number of MCMC samples can achieve significant variance reduction. A possible explanation for this result is discussed in Section 4.2.

Algorithm 1 Markov Chain Monte Carlo Importance Sampling (MCMC–IS)

Require: $\widehat{x}$: previous stage decision
Require: M: number of samples generated using the MCMC algorithm
Require: N: number of samples generated using the approximate zero-variance distribution
Require: ξ_0: starting state of the MCMC algorithm

`Step 1: Generate Samples from the Zero-Variance Distribution using MCMC`

1: Set $k=0$
2: Given the current state ξ_k, generate $\zeta_k \sim q(\cdot \mid \xi_k)$.
3: Generate a uniform random variable $u \sim U \in (0,1)$.
4: Transition to the next state according to,

$$\xi_{k+1} = \begin{cases} \zeta_k & \text{if } \ u \leq a(\xi_k, \zeta_k) \\ \xi_k & \text{otherwise} \end{cases},$$

where,

$$a(\xi_k, \zeta_k) = \min\left\{ \frac{|Q(\widehat{x}, \zeta_k)| f(\zeta_k) q(\xi_k|\zeta_k)}{|Q(\widehat{x}, \xi_k)| f(\xi_k) q(\zeta_k|\xi_k)}, 1 \right\}$$

5: Let $k \leftarrow k+1$. If $k=M$ then proceed to Step 6. Otherwise return to Step 2.

`Step 2: Reconstruct the Zero-Variance Distribution using KDE`

6: For each state of the Markov chain generated from MCMC, reconstruct the approximate zero-variance distribution as,

$$\widehat{g}_M(\xi) = \frac{1}{M} \sum_{i=1}^{M} K_H(\xi, \xi_i).$$

`Step 3: Resample from the Approximate Zero-Variance Distribution to Form an Importance Sampling Estimate`

7: Generate N new samples from $\widehat{g}_M$ and form the importance sampling estimate,

$$\widehat{\mathcal{Q}}^{IS}(\widehat{x}) = \frac{1}{N} \sum_{i=1}^{N} Q(\widehat{x}, \xi_i) \frac{f(\xi_i)}{\widehat{g}_M(\xi_i)}$$

In order to control the error due to the KDE algorithm, we need to ensure that the number of samples are generated by the MCMC algorithm M is large enough, and that the bandwidth parameter h_k is small enough. In particular, if $(Mh^D)^{-1} \to \infty$, $h \to 0$ as $M \to \infty$, and the density function is approximated as,

$$\widehat{g}_M(\xi) = \frac{1}{M} \sum_{i=1}^{M} K_H(\xi, \xi_i) = (Mh^D)^{-1} \sum_{i=1}^{M} K\left(\frac{\xi - \xi_i}{h} \right),$$

then it has been shown that $\widehat{g}_M$ will probabilistically converge to g^* under the total variation norm (see Devroye and Györfi (1985)). Applying this result to the MCMC-IS framework is not straightforward. The complexity stems from the fact that the previous convergence proofs for

the KDE algorithm assume that samples are generated from g^*, whereas in our framework these samples are generated from a Markov chain whose stationary distribution is g^*.

4. Sampling Properties of the MCMC-IS Framework

In this section, we illustrate the properties of the MCMC-IS framework using a simple stochastic programming model, and compare them to estimates that are produced using a Crude Monte Carlo method (CMC), a Quasi Monte Carlo (QMC) method and the importance sampling method proposed in Dantzig and Glynn (1990) and Infanger (1992), which we refer to as the Dantzig-Glynn-Infanger (DGI) method. Our results in this section were produced in MATLAB 2012a. In particular, we used a Mersenne-Twister algorithm to generate random numbers used for CMC methods, and a Sobol sequence that was randomized using the Matousek-Affine-Owen scrambling algorithm for QMC methods. For the MCMC-IS method, we used our own implementation of the Metropolis-Hastings MCMC algorithm and the Adaptive Metropolis MCMC algorithm described in Haario et al. (2001), and we build approximate distributions using the MATLAB KDE Toolbox, which is available online at `http://www.ics.uci.edu/~ihler/code/kde.html`.

4.1. The Newsvendor Model

Our test problem is a two-stage Newsvendor model with uncertain demand and sales prices, where the first-stage decision-making problem is a linear program defined as,

$$\begin{aligned} z^* = \min_{x} \quad & x + \mathcal{Q}(x_1) \\ \text{s.t. } & x \geq 0, \end{aligned} \tag{4.1}$$

and the recourse function is linear program defined as,

$$\begin{aligned} Q(\widehat{x}, \xi) = \min_{y_1, y_2} \quad & -p(\xi) y_1 - r y_2 \\ & y_1 \leq d(\xi), \\ & y_1 + y_2 \leq \widehat{x}, \\ & y_1,\ y_2 \geq 0, \end{aligned} \tag{4.2}$$

where $\widehat{x}$ denotes the quantity of newspapers purchased in the first stage, $\xi = (\xi_1, \xi_2)$ represents the uncertainty in demand $d(\xi)$ and sales price $p(\xi)$ of newspapers in the second-stage and scalar r represents the price of recycling unsold newspapers. In our numerical experiments, we typically model the uncertainty in demand as $d(\xi) = 100 \exp(\xi_1)$ and the uncertainty in sales price as $p(\xi) = 1.5 \exp(\xi_2)$, where ξ_1 and ξ_2 are normal random variables with mean μ and and standard deviation σ. This implies that the uncertainty in $d(\xi)$ and $q(\xi)$ are modeled using a lognormal distribution. We set $\mu = 0.0$ and change the underlying variance of the model by altering the value of σ from $\sigma = 1$ to $\sigma = 2$.

The advantages of using this simple model are that the distributions can be easily visualized, and we can determine the value of true recourse function at various points using numerical integration procedures. In contrast to other test problems in the stochastic programming literature, this setup allows us to calculate the true values of optimal solution x^* and optimal value z^* of the underlying model. In turn, we can examine statistics such as the mean-squared error of the estimate of optimal solution $\tilde{x}$ (defined as $\text{MSE}(\tilde{x}) = \|x^* - \tilde{x}\|_2$), the mean-squared error of the estimate of the optimal value $\tilde{z}$ (defined as $\text{MSE}(\tilde{z}) = \|z^* - \tilde{z}\|_2$), the mean-squared error of the estimate of recourse function at a fixed point $\widehat{\mathcal{Q}}(\widehat{x})$ (defined as $\text{MSE}(\widehat{\mathcal{Q}}(\widehat{x})) = \|\mathcal{Q}(x) - \widehat{\mathcal{Q}}(\widehat{x})\|_2$) and the mean-squared error of the approximate zero-variance distribution (defined as $\text{MSE}(\widehat{g}) = \int (g(\xi) - \widehat{g}(\xi))^2 d\xi$). Indeed, such statistics are crucial when using importance sampling procedures as they typically yield estimates with low sample variance, but may be prone to high bias and high mean-squared error. The results presented below are the averages of thirty simulations. Note that all the results that are presented in Sections 4 and 5 have been normalized for clarity.

4.2. The Required Number of MCMC Samples and the Effects of the Bandwidth Parameter

Our numerical experiments with the Newsvendor model suggest that a modest number of MCMC samples (M) can produce an approximate zero-variance distribution ($\widehat{g}_M$) that yields substantial variance reduction in our estimates of the recourse function. As shown in 2(a), increasing M does reduce the error in our $\widehat{g}_M$. However, the computational cost of such an increase is not justified in terms of the marginal improvement in the accuracy of our recourse function estimates. This is a positive result as the MCMC algorithm represents the most computationally expensive part of our framework. A possible explanation for this empirical observation is that if our $\widehat{g}_M$ qualitatively agrees with g^*, then the sample statistics of the approximate distribution will qualitatively agree with the sample statistics of the zero-variance density. In order to illustrate this point, we plot the contours of the true zero-variance distribution g^* in Figure 2(b) and the contours of $\widehat{g}_M$ for different values of M in Figure 2(c)-2(e). These figures suggest that the approximate distributions produced in the MCMC-IS framework do qualitatively agree with the true zero-variance distribution even at low values of M. In Figure 2(f), we show the contours of our approximate zero-variance distribution after we reduce the bandwidth parameters of the kernel function by 20%. This decreases the MSE of $\widehat{g}_M$ by approximately 12% but increases its variance by approximately 15%, thereby demonstrating the bias-variance tradeoff of KDE algorithms.

4.3. Adaptive Sampling of the Important Regions

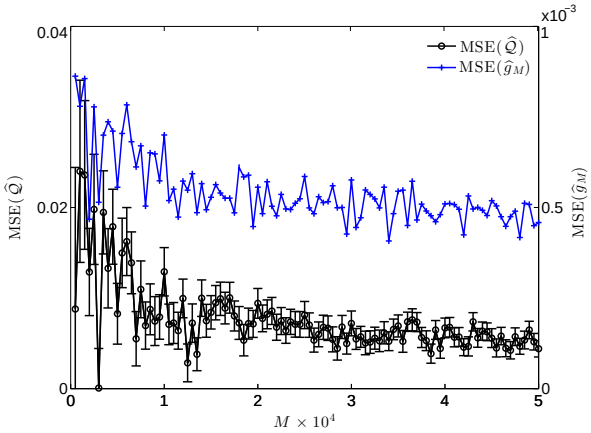
The major difference between our framework and a standard MC method is that we generate samples using an importance sampling distribution $\widehat{g}_M$ as opposed to the original distribution f. As a result, the samples that are generated using $\widehat{g}_M$ are typically located in regions that contribute the most to the value of the recourse function (i.e. in regions where $|Q(x,\xi)|f(\xi)$ is high) while the samples that are generated using f are typically located in regions where the original distribution attains high values. We demonstrate this difference in Figure 3 where we plot a set of samples generated using f and the CMC method (left), and another set of samples generated using $\widehat{g}_M$ and the MCMC-IS framework. The first set of contours in Figure 3 pertains to the original distribution f while the second set of contours pertains to the true zero-variance distribution g^*. Note that f and g^* are not only centered around different points but also have different shapes.

4.4. Dependence of the Sampling Distribution on the Previous Stage Decision

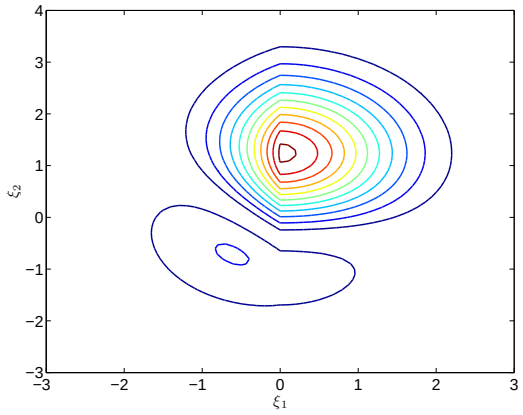
A desired characteristic of any importance sampling distribution used in stochastic programming algorithms is its dependence on the previous stage decision. We illustrate this dependence in Figure 4, where we plot the absolute difference between an approximate zero-variance distribution constructed around the point $\widehat{x}_r = 50$ and two other approximate zero-variance distributions constructed around the point $\widehat{x}_1 = 10$ (left) and $\widehat{x}_2 = 100$ (right). As shown, the approximate zero-variance distribution produced by our framework can vary substantially as we change the previous stage decision.

4.5. Comparison to Other Sampling Algorithms

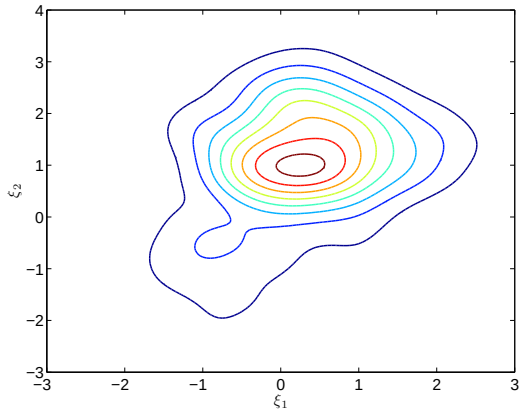
In our next experiment, we compare the estimates produced by MCMC-IS to the estimates produced by CMC, QMC and DGI methods. In Figure 5(b), we plot the sample standard deviation of the different methods. It is clear from this figure that the two importance sampling methods perform well in this respect. The MCMC-IS method does, however, perform better for smaller sample sizes. More importantly, when we plot the error in Figure 5(a), we find that the proposed MCMC-IS method and the QMC sampling method perform best. Our results suggest that the relative advantage of our framework over other variance reduction methods is greater when the uncertainty in the model is increased. This is to be expected since the error of MC based methods



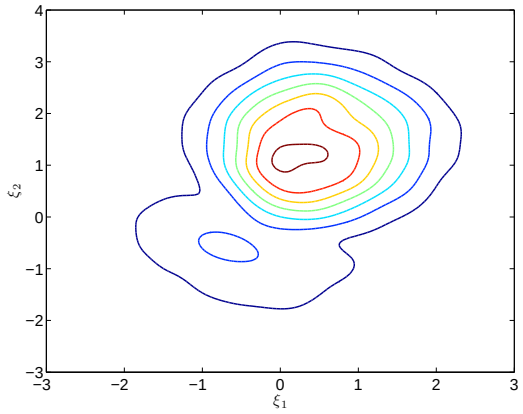
(a)



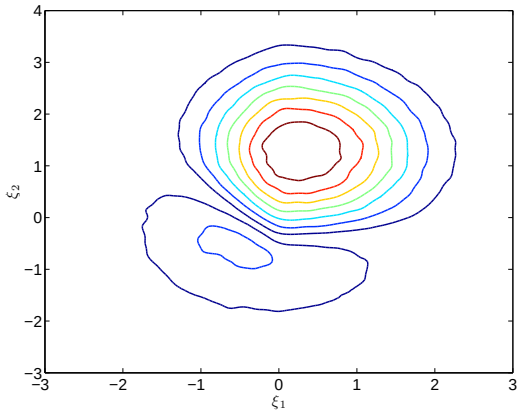
(b) g^*



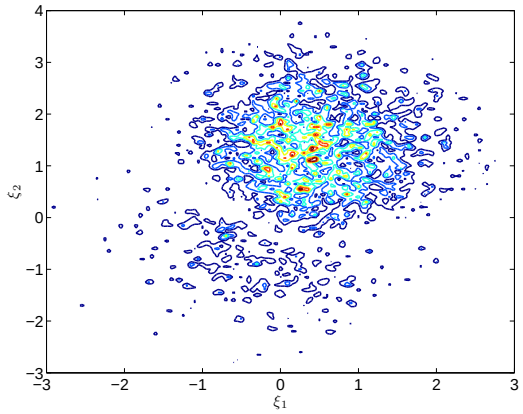
(c) $\widehat{g}_{500}$



(d) $\widehat{g}_{1000}$



(e) $\widehat{g}_{10000}$



(f) $\widehat{g}_{10000}$

Figure 2 (a) The majority of the gains in variance reduction and accuracy can be achieved for a small values of M. Note that the axis for MSE($\widehat{g}_M$) is the right, and the scale for MSE($\widehat{Q}$)is on the left. (b) Contours of g^*. (c)-(e) Contours of $\widehat{g}_M$ for different values of M; the bandwidth parameter for these distributions is estimated using a one-dimensional likelihood-based search. (f) $\widehat{g}_10000$ with a bandwidth that is 20% smaller for each dimension. The resulting mean square error is lower but the variance is higher for the density in (f)

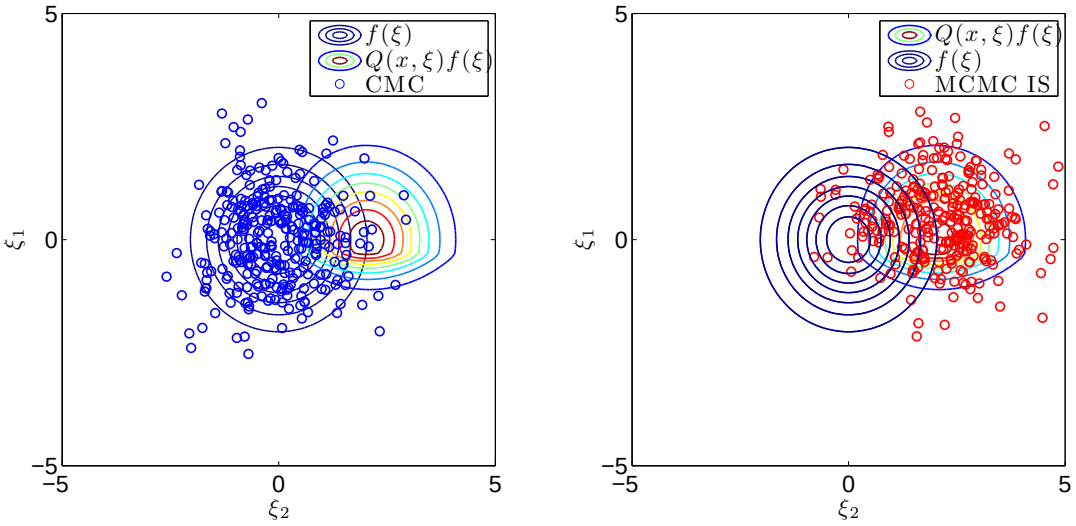


Figure 3 Comparison of points generated with the standard CMC approach, and the proposed MCMC-IS method. Left: using MC sampling; Right: using importance sampling.

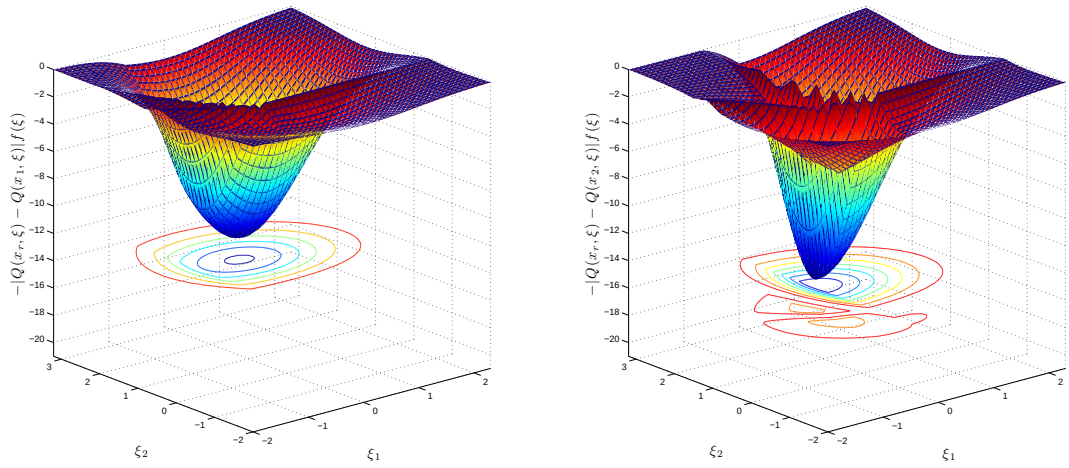


Figure 4 The absolute difference between an approximate zero-variance distribution constructed at $\widehat{x}_r = 50$ and two other approximate zero-variance distributions constructed at $\widehat{x}_1 = 10$ (left) and $\widehat{x}_2 = 100$ (right).

depends on the variance of the random parameters. To illustrate this point, we repeat the same calculations as above but increase the standard deviation of (ξ_1, ξ_2) from $\sigma = 1$ to $\sigma = 2$ as described in Section 4.1. Increasing the variance of the underlying model typically means that more samples are required for the algorithms to produce estimates that have comparable variance and error. In this regime, the MCMC-IS method outperforms all other methods (Figure 5(c) and 5(d)). We note that the error in the DGI estimates of the recourse function converges very slowly in this example because the DGI method uses an approximate zero-variance distribution that is specifically built for a recourse function that is additively separable in the random variables. In the our model,

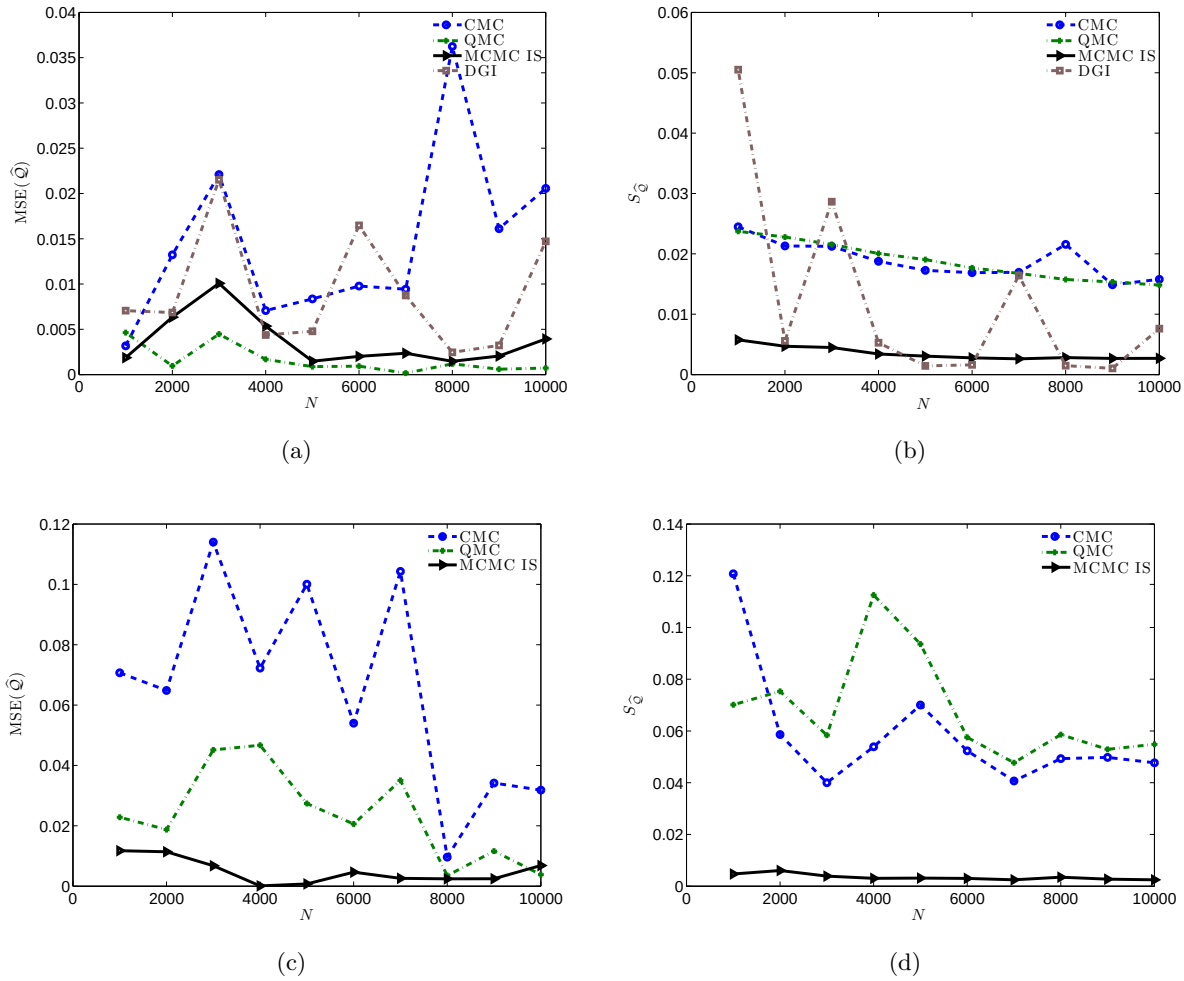


Figure 5 **Top:** Comparison of the accuracy and variance of estimates produced by different methods for a moderate-variance problem with $\sigma = 1$. **Bottom:** Comparison of the accuracy and variance of estimates produced by different methods for a higher variance problem with $\sigma = 2$. Note that we omit the results for the IDG method when $\sigma = 2$ for clarity. The normalized values of $S_{\widehat{Q}}$ and MSE($\widehat{Q}$) for DGI are around 20% and 40% respectively

however, the recourse function is not additively separable. This leads to estimates of the recourse function that have high variance, and high MSE.

4.6. Multimodal Distributions and Rare-Event Simulation

Sampling in the presence of rare events and/or from multimodal distributions is a computationally difficult task. Multimodal distributions arise frequently in operations research models (see, e.g., Ravindran (2008)). Rare event simulation is also an important application area (see, e.g., Bucklew (2004)). When stochastic programming models contain these kinds of complex probability distributions, then many of the sampling techniques that have been used in stochastic programming are difficult to apply, let alone apply efficiently. Although numerous importance sampling techniques have been developed to generate samples from such probability distributions, using these techniques in the context of stochastic programming is not straightforward. First, as was illustrated in the previous section, an ideal importance sampling distribution depends on the incumbent solution and has to be created each time we wish to generate a new sampled cut. This implies that

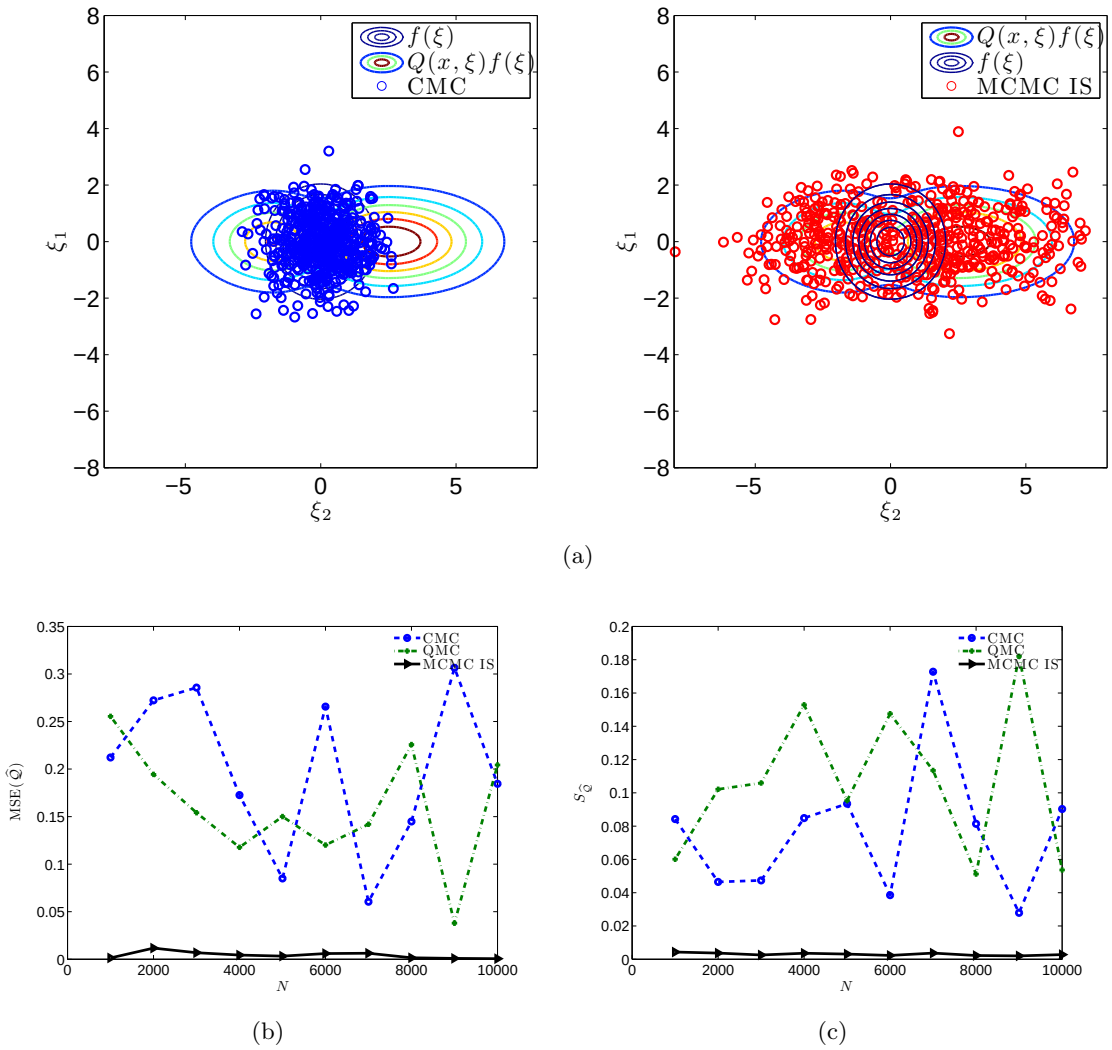


Figure 6 **Top**: Contours of a multimodal model. Samples generated using CMC are shown on the left and the samples from MCMC-IS are shown on the right. **Bottom:** Error and variance of estimates produced by different methods.

efficiency is an important consideration. Second, stochastic programming models not only require us to generate samples from these complex distributions, but to use them to compute an accurate estimate of the recourse function. In turn, an appropriate importance sampling technique has to also be able to accurately evaluate the likelihood of each sample that it generates as in (3.2) or risk generating biased results. These issues often preclude the application of stochastic programming when the distribution of the uncertain variables has a complex structure.

To demonstrate these issues and show that our proposed algorithm can efficiently sample such cases, we use an example where the important regions of the recourse function are described by a surface with two distinct modes, whose contours are shown in 6(a). In this example, the original integrand in the recourse function $Q(\widehat{x},\xi_1,\xi_2)f(\xi_1,\xi_2)$ was replaced with $Q(\widehat{x},w(\xi_1),w(\xi_2))f(\xi_1,\xi_2)$, where $w(\xi)=\exp(\frac{\xi^2}{2}-\frac{(\xi+3)^2}{8})+\exp(\frac{\xi^2}{2}-\frac{(\xi+1)^2}{8})$ and f is the standard multivariate normal density. This example illustrates rare-event sampling, in the sense that the majority of the samples from the important regions are outside of the 2σ interval under the original distribution f.

As in Section 4.4, we then generate a set of samples using the CMC method and the MCMC-IS framework. In this example, the samples that are generated using the CMC method are centered around the origin, where the original distribution f attains its highest values (Figure 6(a), left). In contrast, the samples that are generated using the MCMC-IS framework are centered around the two modes and in proportion to the depth of each mode. These areas constitute the regions that contribute the most to the value of the recourse function and correspond to the areas where the approximate zero-variance distribution $\widehat{g}_M$ takes on its largest values. As a result the MCMC-IS framework obtains an estimate of the recourse function that is both more accurate (Figure 6(b)) and has less variance (Figure 6(c)) than the other methods. In this example, we have omitted the results for the DGI method because the importance sampling weights turn out to be zero for all the samples, meaning that the estimates it produces do not converge. This is a well-known problem with the DGI method that has previously been discussed in Section 1.4 of Higle (1998).

5. Performance of the MCMC-IS Framework in Decomposition Algorithms

In this section, we illustrate the performance of the MCMC-IS framework when it is embedded in a decomposition algorithm and used to solve variants of the Newsvendor model from Section 4.1. We begin by showing that the MCMC-IS framework can lead to accurate lower variance estimates for stochastic programming models for different types of uncertainty. Next, we demonstrate how these lower variance MCMC-IS estimates can improve the performance of stopping tests that are used to assess the convergence in decomposition algorithms. Lastly, we demonstrate the computational benefits of using the MCMC-IS framework in multistage models and demonstrate that a sampling based approach can avoid the exponential growth in problem size that occurs when scenario trees or discretization methods are used to model the uncertainty across multiple time periods.

We solve all stochastic programming models in this section using a MATLAB implementation of the SDDP algorithm, where we use a MEX file and the IBM ILOG CPLEX 12.4 Callable Library to solve linear programs. The random numbers used in this section were produced in MATLAB 2012a. In particular, we used a Mersenne-Twister algorithm to generate random numbers used for CMC methods, and a Sobol sequence that was randomized using the Matousek-Affine-Owen scrambling algorithm for QMC methods. For the MCMC-IS method, we used our own implementation of the Metropolis-Hastings MCMC algorithm and the Adaptive Metropolis MCMC algorithm described in Haario et al. (2001), and we build approximate distributions using the MATLAB KDE Toolbox, which is available online at `http://www.ics.uci.edu/~ihler/code/kde.html`.

5.1. Accuracy and Variance of MCMC-IS Estimates from a Decomposition Algorithm

In our first experiment, we compare the estimates of the optimal value $\tilde{z}$, and the sample standard deviation $S_{\tilde{z}}$ from MCMC-IS, CMC and QMC methods. We consider an extension of the Newsvendor model from Section 4.1, where the Newsvendor buys and sells s different types of newspapers. We purposely do not include any constraints to couple the different types of newspapers so that we can extrapolate the true values of x^* and z^* for the extended problem using the true values from Section 4.1. In this case, we can assess the accuracy of our estimates for a $D = 2 \times s$ dimensional problem by noting that the optimal solution x^* has to be the same for each of the s different types of newspapers, and the optimal value z^* has to scale additively with the number of different newspapers s.

In contrast to the experiments in Section 4, the accuracy of $\tilde{z}$ depends on the number of sampled cuts that are added to the first-stage problem through a decomposition algorithm, as well as the sampling method that is used to generate these estimates. Note that in our implementation of SDDP we count the number of iterations by the number of cuts added to the first stage problem. In practice, the number of iterations needed for the algorithm to converge is determined by a stopping test that is designed to assess whether the decomposition algorithm has converged. In this

experiment, however, we compare estimates that are produced after a fixed number of iterations (we use more conventional stopping rules for the remainder of this section). Fixing the number of iterations ensures that each sampling method produces estimates using the same number of samples, and isolates the performance of the sampling method from the performance of the stopping test, which we examine in Section 5.2. During our numerical experiments we fixed the number of iterations to $8 \times s$. We found that this simple rule was sufficient to show the numerical properties of the different sampling algorithms.

Figure 7 shows the convergence of the estimates that we obtain when we solve a two-stage Newsvendor problem with $D = 2 \times 3 = 6$ random variables after $8 \times 3 = 24$ cuts have been added to the first-stage problem. In Figures 7(a) - 7(d), we model the uncertainty in the demand and sales price of each newspaper using the lognormal distributions from Section 4.1, and we build the approximate zero-variance distribution for each sampled cut using $M = 3000$ samples that are generated from a standard Metropolis Hastings MCMC algorithm. In Figures 7(e) - 7(f), we model the uncertainty in the demand and sales price of each newspaper using the multimodal rare-event distribution from Section 4.6, and we build the approximate zero-variance distribution for each sampled cut using $M = 3000$ samples that are generated from the Adaptive Metropolis algorithm described in Haario et al. (2001).

Our results suggest that the relative advantage of using MCMC-IS estimates depends on the inherent variance of the underlying stochastic programming model. In models where the uncertainty is modeled using a lower variance distribution, MCMC-IS produces estimates that are just as accurate as the estimates produced by a QMC method, but that are still more accurate than the estimates produced by a CMC method. In models where the uncertainty is modeled using a higher variance or rare-event distribution, MCMC-IS produces estimates that are much more accurate than those produced by QMC and CMC methods. Our numerical results also suggest that MCMC-IS produces estimates with sample standard deviations that are far lower than the estimates produced by CMC and QMC methods.

5.2. Performance of MCMC-IS Estimates for Stopping Tests in Decomposition Algorithms

In our second experiment, we illustrate how the low sample standard deviations that we obtain through the MCMC-IS can improve the performance of stopping tests that are used to assess convergence in decomposition algorithms. Once again, we use the Newsvendor model from Section 4.1 in order to gauge the accuracy of our estimates. We restrict our focus to estimates of the optimal decision, optimal value, and the sample standard deviation of the recourse function, which we denote as $\tilde{x}$, $\tilde{z}$ and $S_{\tilde{z}}$ respectively. We illustrate the effect of the underlying variance using models where the demand and sales price of newspapers is distributed according to a lower variance lognormal distribution with $\sigma = 1$, a higher variance lognormal distribution with $\sigma = 2$, and a multimodal rare-event distribution. The exact parameters of these distributions are specified in Sections 4.1 and 4.6 respectively.

Given that both the lower bound LB and the upper bound UB are random variables in this setting, our stopping test is designed to assess whether the expected values of the upper bound $\mathbb{E}[UB]$ and lower bound $\mathbb{E}[LB]$ of the model are equal to one another. This requires a one-sided two-sample t-test for the equality of means,

$$H_0 : \mathbb{E}(UB) = \mathbb{E}(LB) \text{ vs } H_A : \mathbb{E}(UB) > \mathbb{E}(LB). \tag{5.1}$$

We assume that the samples are unpaired, that the sample sizes are unequal and that the standard deviation of these variables are unknown but identical. We estimate the expected values of these parameters using the sample averages $\overline{LB}$ and $\overline{UB}$. Similarly, we estimate the standard deviation of these parameters using the sample standard deviations S_{LB} and S_{UB}. The sample average and the

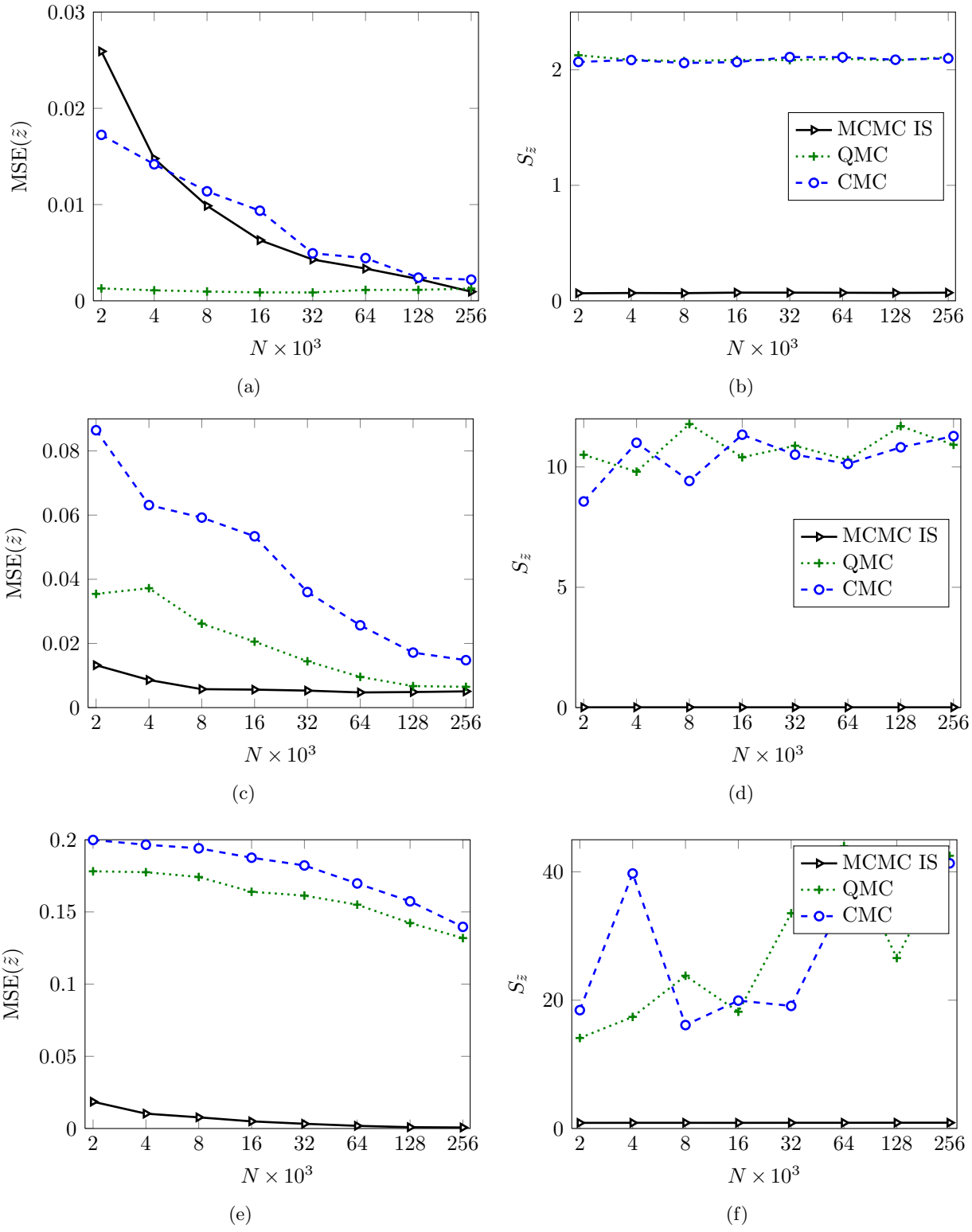


Figure 7 Error and variance of estimates for a Newsvendor model where the uncertainty in demand and sales price is modeled using a lower variance lognormal distribution with $\sigma = 1$ (7(a) - 7(b)), a higher variance lognormal distribution with $\sigma = 2$ (7(c) - 7(d)), and multimodal rare-event distribution (7(e) - 7(f))

sample standard deviation for the lower bound is constructed using $M_{LB} = 3000$ and $N_{LB} = 16000$ samples, while the sample average and sample standard deviation for the upper bound is constructed using $N_{UB} = 16000$ samples. We note that we do not need to rebuild an approximate zero-variance distribution to construct an upper bound estimate, as the lower bound and upper bound estimates are constructed around the same first-stage solution $\widehat{x}$. Unlike traditional hypothesis tests, we are not seeking to reject the null hypothesis but to accept it. As such, our test stops when we are unable to reject the null hypothesis with a significance level of $\alpha = 0.99$. The well-known duality between hypothesis tests and confidence intervals implies that this procedure is similar to the stopping tests that involve confidence intervals that are suggested in the literature. We refer the interested reader to Pereira and Pinto (1991) and Birge and Louveaux (2011) for a more detailed explanation of how to construct the upper and lower bound estimates within SDDP and decomposition algorithms in general.

Our results in Tables 1-3 suggest that MCMC-IS has a positive impact on stopping tests because it reduces the sample standard deviation of upper and lower bound estimates and thereby improves the power of the underlying stopping test. In the context of stopping tests, a test with low power means that the null hypothesis H_0 is frequently rejected when it is false. In practice, a stopping test with low power terminates a decomposition algorithm before has converged and ultimately results in high errors in the values of $\tilde{x}$ and $\tilde{z}$. As in previous sections, these effects become more significant when the variance of the underlying model is increased. Given that these results are based on a simple two-stage model with only two random variables, we expect such effects to be more prominent for models that contain more random variables and/or require more cuts to achieve convergence as in 5.1. However, we note these results cannot be immediately extended to multistage stochastic programming models: although the MCMC-IS framework can consistently be used to compute lower variance estimates of the lower bound for all stochastic programming models, it can only be used to compute lower variance estimates of the upper bound for two-stage models. This is because the MCMC-IS framework build an approximate zero-variance distribution around a single set of incumbent solutions, while the upper bound estimate in SDDP is constructed around multiple sets of incumbent solutions. In light of the large impact that the MCMC-IS framework can have on stopping procedures, we plan to explore more ways to use the framework to generate upper bound estimates in our future work.

Method	S_{LB}	S_{UB}	Cuts Added	MSE($\tilde{x}$)	MSE($\tilde{z}$)
MCMC-IS	40	48	7.1	4.4%	0.7%
CMC	326	329	5.9	9.2%	2.0%
QMC	312	316	5.6	16.5%	3.0%

Table 1 Output from a Newsvendor model using a lower variance lognormal distribution ($\sigma = 1$)

Method	S_{LB}	S_{UB}	Cuts Added	MSE($\tilde{x}$)	MSE($\tilde{z}$)
MCMC-IS	788	839	7.5	5.4%	0.6%
CMC	44655	43686	4.7	39.0%	33.4%
QMC	33376	40552	5.0	36.3%	23.8%

Table 2 Output from a Newsvendor model using a higher variance lognormal distribution ($\sigma = 2$)

Method	S_{LB}	S_{UB}	Cuts Added	MSE($\tilde{x}$)	MSE($\tilde{z}$)
MCMC-IS	276	233	6.4	5.9%	0.5%
CMC	40589	20902	3.7	37.9%	10.7%
QMC	19901	16358	3.8	58.8%	12.7%

Table 3 Output from a Newsvendor Model using a multimodal rare-event distribution

5.3. Computational Performance MCMC-IS for Multistage Stochastic Programming Models

In our final experiment, we demonstrate the computational benefits of using a sampling approach within a decomposition algorithm. In particular, we pair the MCMC-IS framework within the SDDP algorithm and use this setup to solve a multistage extension of the Newsvendor model in Section 4.1.

In our multistage extension of the Newsvendor model, the Newsvendor buys and sells a single type of newspapers over T consecutive days. We assume that the sales price and demand of the newspaper are distributed according to a lower variance lognormal distribution with $\sigma = 1$. We also assume that any newspapers that are to be sold on day $t+1$ have to be bought on day t, and that any unsold newspapers at the end of day $t+1$ have to be recycled at a price of r. Together, these assumptions allow us to extrapolate the true values of the optimal solution x^* and optimal value z^* for the multistage model from their corresponding values in the two-stage model in Section 4.1. We reason that the optimal value z^* scales additively with the number of time periods, and the optimal solution x^* remains the same. As such, we can verify that the SDDP algorithm has converged to its true value without having to rely on estimates of the upper and lower bound.

Figure 8(a) shows that the computational complexity of our setup increases quadratically with the time horizon of the underlying problem. Moreover, as is clear from Figure 8(b) the solution estimated with the MCMC-IS framework is within 1% of the true value. This represents a significant computational advantage in comparison to a scenario-tree based approach, where the number of linear programs that have to be solved to achieve convergence increases exponentially. The exact number of linear programs that have to be evaluated in this case is is determined by the number of samples that we use to construct the sampled cuts at each iteration of the SDDP algorithm, as well as the number of iterations of the SDDP algorithm that we have to run until a stopping test indicates convergence. In this case, we construct sampled cuts using the MCMC-IS algorithm with $M_{LB} = 3000$ and $N_{LB} = 16000$ samples, and we use the stopping test we describe in Section 5.2 to assess convergence. We note that the stopping test from 5.2 requires an upper bound estimate, which we construct at each iteration of the using $N_{UB} = 16000$ samples.

6. Conclusions

Multistage stochastic programming problems are considered computationally challenging mainly because the evaluation of the recourse function involves the solution of a multidimensional integral. Numerical methods such as Sample Average Approximation (SAA) and Stochastic Dual Dynamic Programming (SDDP) rely on sampling algorithms to approximately estimate the recourse function. The sampling algorithm used in conjunction with the optimization algorithm has a major bearing on the efficiency of the overall algorithm and on the accuracy of the solution. As a result the development of efficient sampling methods is an active area of research in stochastic programming.

The main contribution of this paper is the development of an importance sampling framework that is based on Markov Chain Monte Carlo (MCMC) to generate biased samples, and a kernel density estimation method to compute the likelihood function. Importance sampling has been proposed before in the literature of stochastic programming. The proposed method makes fewer restrictive assumptions than the importance sampling algorithm proposed in Dantzig and Glynn (1990) and Infanger (1992), and in particular can perform well even when the objective function

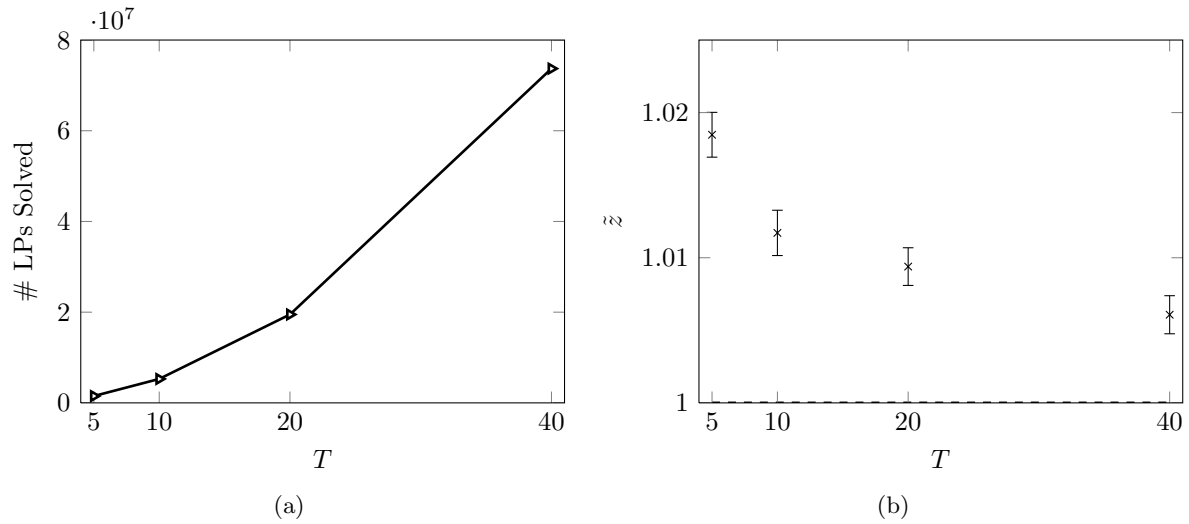


Figure 8 (a) Complexity of SDDP with the proposed sampling algorithm grows quadratically with the number of dimensions. (b) Estimated solution is within 1% even for problems with a large number of time periods.

is not additively separable. Our numerical experiments show that the method outperforms Crude Monte Carlo and Quasi Monte Carlo algorithms when the problem has moderate or high variance, and when the probability density function is difficult to sample from.

The importance sampling framework proposed in this paper can be extended in many ways. We have shown how importance sampling can be used in the context of a decomposition algorithm and expected value optimization. However, it is possible to use our approach with different algorithms (e.g. SAA) and with different types of stochastic programming models (e.g. risk averse stochastic programming). In addition, we have shown that the proposed method performs well when compared to existing methods. However we only used simplistic MCMC and kernel density estimation algorithms. We expect that the proposed algorithm will be even more efficient if more advanced MCMC or KDE algorithms are used.

References

Asmussen, Søren, Peter W. Glynn. 2007. *Stochastic simulation: algorithms and analysis*, *Stochastic Modelling and Applied Probability*, vol. 57. Springer, New York.

Bayraksan, G., D.P. Morton. 2011. A sequential sampling procedure for stochastic programming. *Operations Research* **59**(4) 898–913.

Birge, J.R., F. Louveaux. 2011. *Introduction to stochastic programming*. Springer Verlag.

Bucklew, James Antonio. 2004. *Introduction to rare event simulation*. Springer Series in Statistics, Springer-Verlag, New York.

Dantzig, G.B., P.W. Glynn. 1990. Parallel processors for planning under uncertainty. *Annals of Operations Research* **22**(1) 1–21.

Devroye, Luc, László Györfi. 1985. *Nonparametric density estimation*. Wiley Series in Probability and Mathematical Statistics: Tracts on Probability and Statistics, John Wiley & Sons Inc., New York. The L_1 view.

Drew, S.S., T. Homem-de Mello. 2006. Quasi-monte carlo strategies for stochastic optimization. *Proceedings of the 38th conference on Winter simulation*. Winter Simulation Conference, 774–782.

Gelman, A., S. Brooks, G. Jones, X.L. Meng. 2010. *Handbook of Markov Chain Monte Carlo: Methods and Applications*. Chapman & Hall/CRC.

Haario, H., E. Saksman, J. Tamminen. 2001. An adaptive metropolis algorithm. *Bernoulli* 223–242.

Higle, J.L. 1998. Variance reduction and objective function evaluation in stochastic linear programs. *INFORMS Journal on Computing* **10**(2) 236–247.

Higle, J.L., S. Sen. 1991. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of operations research* 650–669.

Homem-de Mello, T. 2006. On rates of convergence for stochastic optimization problems under non-iid sampling. *Manuscript, available on Optimization Online* .

Homem-de Mello, Tito, Vitor de Matos, Erlon Finardi. 2011. Sampling strategies and stopping criteria for stochastic dual dynamic programming: a case study in long-term hydrothermal scheduling. *Energy Systems* **2** 1–31. URL `http://dx.doi.org/10.1007/s12667-011-0024-y`. 10.1007/s12667-011-0024-y.

Infanger, G. 1992. Monte carlo (importance) sampling within a benders decomposition algorithm for stochastic linear programs. *Annals of Operations Research* **39**(1) 69–95.

Koivu, M. 2005. Variance reduction in sample approximations of stochastic programs. *Mathematical programming* **103**(3) 463–485.

Linderoth, J., A. Shapiro, S. Wright. 2006. The empirical behavior of sampling methods for stochastic programming. *Annals of Operations Research* **142**(1) 215–241.

Morton, D.P. 1998. Stopping rules for a class of sampling-based stochastic programming algorithms. *Operations research* 710–718.

Parpas, Panos, Berç Rustem. 2007. Computational assessment of nested Benders and augmented Lagrangian decomposition for mean-variance multistage stochastic problems. *INFORMS J. Comput.* **19**(2) 239–247.

Pennanen, T., M. Koivu. 2005. Epi-convergent discretizations of stochastic programs via integration quadratures. *Numerische mathematik* **100**(1) 141–163.

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

Powell, W.B. 2007. *Approximate Dynamic Programming: Solving the curses of dimensionality*, vol. 703. Wiley-Blackwell.

Ravindran, A. Ravi, ed. 2008. *Operations research and management science handbook*. Operations Research Series, CRC Press, Boca Raton, FL.

Roberts, Gareth O., Jeffrey S. Rosenthal. 2004. General state space Markov chains and MCMC algorithms. *Probab. Surv.* **1** 20–71. doi:10.1214/154957804100000024. URL `http://dx.doi.org/10.1214/154957804100000024`.

Rockafellar, R.T., R.J.B. Wets. 1991. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of operations research* 119–147.

Scott, David. 1992. *Multivariate Density Estimation: Theory, Practice, and Visualization (Wiley Series in Probability and Statistics)*. 1st ed. Wiley.

Shapiro, A. 2009. On a time consistency concept in risk averse multistage stochastic programming. *Operations Research Letters* **37**(3) 143–147.

Shapiro, A., D. Dentcheva, A.P. Ruszczyński. 2009. *Lectures on stochastic programming: modeling and theory*, vol. 9. Society for Industrial Mathematics.

Shapiro, A., T. Homem-de Mello. 1998. A simulation-based approach to two-stage stochastic programming with recourse. *Mathematical Programming* **81**(3) 301–325.

Silverman, Bernard. 1986. *Density Estimation for Statistics and Data Analysis (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. 1st ed. Chapman and Hall/CRC.

Watson, J.P., R.J.B. Wets, D.L. Woodruff. 2010. Scalable heuristics for a class of chance-constrained stochastic programs. *INFORMS Journal on Computing* **22**(4) 543.