An Efficient Algorithm for Rare-event Probability Estimation, Combinatorial Optimization, and Counting

Zdravko I. Botev Dirk P. Kroese

February 25, 2008

Abstract

Although importance sampling is an established and effective sampling and estimation technique, it becomes unstable and unreliable for high-dimensional problems. The main reason is that the likelihood ratio in the importance sampling estimator degenerates when the dimension of the problem becomes large. Various remedies to this problem have been suggested, including heuristics such as resampling. Even so, the consensus is that for large-dimensional problems, likelihood ratios (and hence importance sampling) should be avoided. In this paper we introduce a new adaptive simulation approach that does away with likelihood ratios, while retaining the multi-level approach of the cross-entropy method. Like the latter, the method can be used for rare-event probability estimation, optimization, and counting. Moreover, the method allows one to sample exactly from the target distribution rather than asymptotically as in Markov chain Monte Carlo. Numerical examples demonstrate the effectiveness of the method for a variety of applications.

Keywords: likelihood ratio degeneracy, kernel density estimation, importance sampling, exact sampling, rare-event probability estimation, combinatorial counting

1 Introduction

Estimation and optimization problems in applied probability and statistics often concern sampling from specified target distributions. Standard sampling methods include the inverse-transform method, the acceptance-rejection method, and the alias method, as well as many other ad hoc methods; see, for example, [5]. Some of these methods are *exact*, that is, the generated random variables are distributed exactly according to the target distribution. In most cases, however, exact sampling is either infeasible or very costly. In such cases one often resorts to *approximate* sampling. A popular example is Markov chain Monte Carlo (MCMC), with its many variants; see, for example, [14].

Despite this abundance of sampling techniques, a significant amount of international research is currently being directed towards finding better random variable generation methods. There are several reasons for this, which are discussed next.

First, one of the main drawbacks of MCMC is that the underlying Markov chain needs to be run until it is close to stationarity — the so-called *burn-in* period. There are two problems here: (a) the Markov chain theoretically never reaches stationarity (unless some coupling argument is used), so that exact sampling is never achieved; and (b) the choice of the burn-in period is highly subjective, and can often be justified mathematically only in situations where a more direct exact sampling approach is also feasible. Although exact MCMC methods are currently being investigated (see, for example, [9]), these methods are still no real match for standard MCMC techniques in terms of applicability and speed.

Second, for problems where the target distribution involves *rare events*, standard generation techniques cannot be easily applied, as they require a large amount of simulation effort. There are a number of ways to deal with rare events. A well-known approach is to use *importance sampling*, where an "easy" sampling distribution is chosen that is close to the actual target. To get unbiased estimates, however, each sample needs to be weighted by the likelihood ratio of the sampling and target distribution. One difficulty with importance sampling is that a good choice for the sampling distribution may not be easily obtained. Fortunately, recent advances in *adaptive* importance sampling techniques have made this technique much more feasible; a typical example is the cross-entropy (CE) method [11, 16, 21], and its offshoots [17, 18, 20]. A more fundamental and serious difficulty with importance sampling is that the corresponding likelihood ratio *degenerates* when the dimension of the problem becomes large. Although various patches to this problem have been suggested (including resampling [7]), the general idea is that for large-dimensional problems, likelihood ratios (and hence importance sampling) should be avoided. A different approach in rare-event simulation is to view the rare event as a sequence of nested events. The advantage is that the estimation and generation can now be conducted in various stages, and that the product rule of probability can be invoked to calculate rare-event probabilities. One of the earliest examples is the *splitting* technique of Kahn and Harris [10], which was the precursor of the RESTART method [23] for rare-event simulation. Ross [15] and Holmes & Diaconis [6] use the nested level idea to solve discrete counting problems in statistical mechanics and computer science. Olafsson [13] describes a nested partitions approach for random search. We employ the nested level idea as well, but in a way that ensures exact sampling from the minimum variance importance sampling pdf corresponding to the rare-event problem at hand.

Third, MCMC methods often take advantage of the fact that the target only needs to be known *up to a normalization constant.* This is essential in Bayesian statistics, where the normalization constant is often difficult to compute or estimate. However, in many estimation problems it is *essential* that the normalization constant can be computed or estimated.

Finally, new random variable methods are needed to deal with the plenitude of new Monte Carlo applications in rare-event simulation, counting in #P complete problems, and continuous and discrete optimization; see, for example, [22].

The purpose of this paper is to introduce a new sampling method that addresses the shortcomings of traditional methods, while retaining various of their fundamental features. In particular, our method combines the adaptiveness and level-crossing idea of CE with an MCMC-like generation algorithm. However, the important differences with classical MCMC are that (1) the samples are generated *exactly* and (2) the normalization constant can be estimated (or is known) directly. In addition, the nesting of events and the use of the product rule — as in the splitting technique — will be crucial in the efficient estimation of rare-event probabilities. Moreover, our method avoids using likelihood ratios, making it suitable for large-dimensional problems. The method is based on recent insights into kernel density estimation [1, 3], the bootstrap method [8], and the multi-level approach of the CE method [21]. The method works well for both continuous and discrete estimation problems, and can also be used for perfect sampling from the Ising model [2].

The rest of the paper is organized as follows. In Section 2 we first provide a brief background on rare-event simulation, and then gently introduce the ideas of the proposed method via three illustrative examples: a rare-event probability estimation example of large dimension, a permutation counting example, and a knapsack combinatorial optimization example. We provide explicit algorithms for each of the examples, making them easy to implement and test. In Section 3 we present a general version of the main method which broadens its applicability to many other rare-event simulation and optimization problems. Section 4 presents a number of applications of the method, including the traveling salesman problem, the quadratic assignment problem, and the estimation of light- and heavy-tailed rare-event probabilities. Finally we draw some conclusions based on the numerical results and give directions for future work.

2 Estimation and Optimization via Monte Carlo Simulation

The purpose of this section is to provide a number of concrete examples in estimation, optimization, and counting, to quickly explain the workings of the proposed method. We first briefly discuss the general sampling framework in the context of rare-event probability estimation.

Many estimation and optimization problems involve the estimation of probabilities of the form

$$\ell(\gamma) = \mathbb{P}\left(S(\mathbf{X}) \geqslant \gamma\right) , \tag{1}$$

where $\mathbf{X}$ is a random object (vector, path, etc.) that takes values in some set $\mathscr{X}$ and is distributed according to a pdf f, S is a real-valued function on $\mathscr{X}$, and $\gamma \in \mathbb{R}$ is a threshold level. We assume that sampling from f is easy. We are particularly interested in the case where ℓ is very small. Such rare-event probabilities are difficult to estimate via crude Monte Carlo (CMC), because in the corresponding CMC estimator,

$$\widehat{\ell}_{\text{CMC}} = \frac{1}{N}\sum_{i=1}^{N} I\{S(\mathbf{X}_i) \geqslant \gamma\}, \quad \mathbf{X}_1, \ldots, \mathbf{X}_N \sim_{\text{iid}} f(\mathbf{x}) , \tag{2}$$

most or all of the indicators $I\{S(\mathbf{X}_i) \geqslant \gamma\}$ are 0, unless a very large sample size N is used. Moreover, the *relative error* (RE) of the CMC estimator, defined as

$$\frac{\sqrt{\text{Var}(\widehat{\ell})}}{\mathbb{E}\widehat{\ell}} = \sqrt{\frac{1-\ell}{N\ell}}, \tag{3}$$

grows as $1/\sqrt{N\ell}$ as $\ell \to 0$. This shows that estimating small probabilities using CMC is computationally involved.

A standard solution to this problem is to use the *importance sampling* estimator:

$$\widehat{\ell}_{\text{IS}} = \frac{1}{N}\sum_{i=1}^{N} W(\mathbf{X}_i) I\{S(\mathbf{X}_i) \geqslant \gamma\}, \quad \mathbf{X}_1, \ldots, \mathbf{X}_N \sim_{\text{iid}} g(\mathbf{x}), \tag{4}$$

where $W(\mathbf{X}_i) = f(\mathbf{X}_i)/g(\mathbf{X}_i)$ is the *likelihood ratio*. The importance sampling pdf g is chosen so as to make the event $\{S(\mathbf{X}) \geqslant \gamma\}$ less rare, while maintaining a small RE for the estimator. It is well known that ℓ can be efficiently estimated via importance sampling, provided that the sampling distribution is close to the pdf

$$g^*(\mathbf{x} \,|\, \gamma) = \frac{f(\mathbf{x}) I\{S(\mathbf{x}) \geqslant \gamma\}}{\ell(\gamma)} \, . \tag{5}$$

This pdf is the minimum-variance importance sampling density for the estimation of ℓ; see, for example, [22]. An obvious difficulty is that g^* itself depends on the unknown constant ℓ, and therefore can not be used *directly* as an importance sampling pdf.

Often the importance sampling pdf g is chosen in the same parametric family $\{f(\mathbf{x};\mathbf{v}),\ \mathbf{v} \in \mathscr{V}\}$ as the nominal pdf $f(\mathbf{x}) = f(\mathbf{x};\mathbf{u})$, where $\mathscr{V}$ is an appropriate parameter space. Then, the problem of selecting a good g reduces to the problem of selecting a good parameter $\mathbf{v}$. Two well-known methods for choosing such an optimal reference parameter $\mathbf{v}$ are the CE method [21, 22] and the variance minimization (VM) method [12]. In the latter, the optimal $\mathbf{v}$ is determined or estimated from the solution to the variance minimization program

$$\min_{\mathbf{v}\in\mathscr{V}} \mathrm{Var}_{\mathbf{v}}(\widehat{\ell}_{\mathrm{IS}}) \;. \tag{6}$$

In the CE method the parameter $\mathbf{v}$ is determined or estimated from the solution to the program

$$\max_{\mathbf{v}\in\mathscr{V}} \mathbb{E}_{\mathbf{u}} I\{S(\mathbf{X}) \geqslant \gamma\} \ln f(\mathbf{X};\mathbf{v}) \;. \tag{7}$$

This program is motivated by information-theoretic arguments presented in [21].

Although both the VM and the CE programs work well and perform similarly for a large variety of estimation problems, serious problems arise when the dimensionality of the problem becomes too large. In particular, when the dimension of $\mathbf{X}$ is large, the likelihood ratio terms in (4) suffer from the well-known likelihood ratio *degeneration* problem [22]. Various remedies have been suggested in the literature. For example, the screening method (see [19] and Section 8.2.2 of [22]) is a modification of the original CE and VM methods that reduces the number of dimensions of the likelihood ratio term W. Below we compare the proposed method with the CE and VM methods and the screening counterparts of the CE and VM methods. We should point out, however, that the CE method uses likelihood ratios only in estimation problems. For optimization problems, where no likelihood ratios are used, the CE method does not suffer from the degeneracy problem described above.

As mentioned before, the proposed method does not use importance sampling, but uses an adaptive level-set approach. As in the CE and VM methods, the minimum variance pdf (5) plays a crucial role in our method. Indeed, the method aims to generate exact samples from (5).

We illustrate the above framework and the workings of the method via three examples, dealing with rare-event probability estimation, counting, and optimization, respectively.

2.1 Rare-Event Probability Estimation Example

Consider the weighted graph in Figure 1. The system consists of $m \times n$ ordinary *bridge* graphs arranged in a grid. A source and terminal vertex are added, as well as zero-weight edges connecting the bridges, indicated by dashed lines. Denote the random lengths of the (solid-line) edges within the (i, j)-th bridge by $X_{ij1}, \ldots, X_{ij5}$, in the order indicated in the figure.

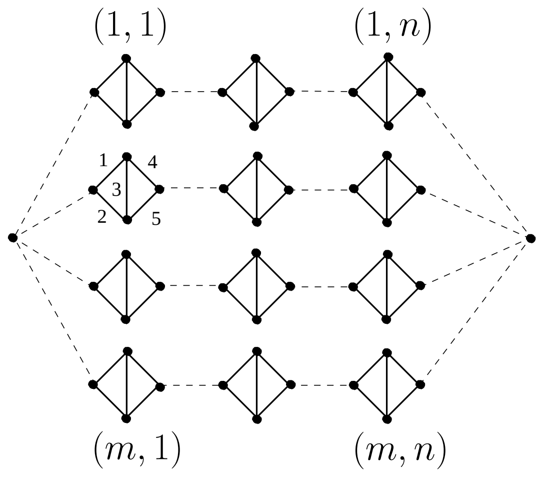

Figure 1: The $m \times n$ bridge system is an array of ordinary bridge systems.

The length of the shortest path through bridge (i, j) is

$$Y_{ij} = \min\{X_{ij1} + X_{ij4},\ X_{ij2} + X_{ij5}, X_{ij1} + X_{ij3} + X_{ij5},\ X_{ij2} + X_{ij3} + X_{ij4}\}\ , \quad (8)$$

and the shortest path from the source to the terminal is

$$S(\mathbf{X}) = \min\{Y_{11} + \cdots + Y_{1n}, \ldots, Y_{m1} + \cdots + Y_{mn}\}\ , \quad (9)$$

where $\mathbf{X}$ is the random vector of components $\{X_{ijk}\}$. Suppose that the $\{X_{ijk}\}$ are independent and that each component X_{ijk} has a $\mathsf{Weib}(\alpha, \lambda_{ijk})$ density; that is, with pdf

$$f_{ijk}(x; \alpha, \lambda_{ijk}) = \alpha\lambda_{ijk}(\lambda_{ijk}x)^{\alpha-1}\mathrm{e}^{-(\lambda_{ijk}x)^{\alpha}}, \quad x > 0\ .$$

The density of $\mathbf{X}$ is then $f(\mathbf{x}) = \prod_{ijk} f_{ijk}(x_{ijk}; \alpha, \lambda_{ijk})$.

Suppose we are interested in estimating the probability, ℓ, that the shortest path through the network exceeds some length γ. In other words, we wish to estimate $\ell = \mathbb{P}(S(\mathbf{X}) \geqslant \gamma)$. For large γ the probability ℓ becomes very small.

In our proposed method, we directly sample from the sequence of minimum variance importance sampling pdfs: $\{g_t^*\} = \{g^*(\cdot \mid \gamma_t)\}$, where $-\infty = \gamma_{-1} < \gamma_0 < \gamma_1 < \ldots < \gamma_{T-1} < \gamma_T = \gamma$ is a sequence of levels. Note that $g^*(\cdot \mid \gamma_{-1}) \equiv f(\cdot)$. We will show that, as a consequence of sampling from the sequence $\{g_t^*\}$, the proposed method provides a natural estimate for ℓ. Note that we assume that the sequence of levels is such that the conditional probabilities $c_t = \mathbb{P}(S(\mathbf{X}) \geqslant \gamma_t \mid S(\mathbf{X}) \geqslant \gamma_{t-1}),\ t = 0, \ldots, T,\ \ \gamma_{-1} = -\infty$ are not small. In cases where it is not obvious how to construct such a sequence, we will later present an adaptive procedure that provides a suitable sequence of levels. Without going into further detail, the method as applied to rare-event probability estimation is outlined below.

Algorithm 2.1 (Estimation of Rare-Event Probabilities) *Given the sequence of levels $\gamma_0 < \gamma_1 < \cdots < \gamma_T = \gamma$ and the sample size N, execute the following steps.*

1. *Set a counter $t = 1$. Generate*

$$\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_N^{(0)} \sim f(\mathbf{x}).$$

Let $\widetilde{\mathscr{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)}, \ldots, \widetilde{\mathbf{X}}_{N_0}^{(0)}\}$ be the subset of the population $\{\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_N^{(0)}\}$ for which $S(\mathbf{X}_i^{(0)}) \geqslant \gamma_0$. Note that

$$\widetilde{\mathbf{X}}_1^{(0)}, \ldots, \widetilde{\mathbf{X}}_{N_0}^{(0)} \sim g^*(\mathbf{x} \mid \gamma_0)$$

and that an unbiased estimate for $c_0 = \ell(\gamma_0)$ is

$$\widehat{c}_0 = \widehat{\ell}(\gamma_0) = \frac{1}{N} \sum_{i=1}^{N} I\left\{S(\mathbf{X}_i^{(0)}) \geqslant \gamma_0\right\} = \frac{N_0}{N}.$$

2. *Sample uniformly with replacement N times from the population $\widetilde{\mathscr{X}}^{(t-1)}$ to obtain a new sample $\mathbf{Y}_1, \ldots, \mathbf{Y}_N$. Note that $\mathbf{Y}_1, \ldots, \mathbf{Y}_N \sim g^*(\mathbf{x} \mid \gamma_{t-1})$, since the resampling does not change the underlying distribution.*

3. *For each $\mathbf{Y} = (Y_1, \ldots, Y_r)$ (with $r = 5mn$) in $\{\mathbf{Y}_1, \ldots, \mathbf{Y}_N\}$, generate $\widetilde{\mathbf{Y}} = (\widetilde{Y}_1, \ldots, \widetilde{Y}_r)$ as follows:*

   (a) *Draw $\widetilde{Y}_1$ from the conditional pdf $g^*(y_1 \mid \gamma_{t-1}, Y_2, \ldots, Y_r)$.*

   (b) *Draw $\widetilde{Y}_i$ from $g^*(y_i \mid \gamma_{t-1}, \widetilde{Y}_1, \ldots, \widetilde{Y}_{i-1}, Y_{i+1}, \ldots, Y_r)$, $\quad i = 2, \ldots, r-1$.*

   (c) *Draw $\widetilde{Y}_r$ from $g^*(y_n \mid \gamma_{t-1}, \widetilde{Y}_1, \ldots, \widetilde{Y}_{r-1})$.*

   Denote the population of $\widetilde{\mathbf{Y}}$s thus obtained by $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)}$. Note that $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)} \sim g^*(\mathbf{x} \mid \gamma_{t-1})$, since conditional sampling does not change the distribution of $\mathbf{Y}_1, \ldots, \mathbf{Y}_N$.

4. *Let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)}, \ldots, \widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the subset of the population $\{\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)}\}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t$. We thus have that*

$$\widetilde{\mathbf{X}}_1^{(t)}, \ldots, \widetilde{\mathbf{X}}_{N_t}^{(t)} \sim g^*(\mathbf{x} \mid \gamma_t).$$

An unbiased estimate for the conditional probability

$$c_t = \mathbb{P}_f(S(\mathbf{X}) \geqslant \gamma_t \mid S(\mathbf{X}) \geqslant \gamma_{t-1}) = \mathbb{P}_{g^*_{t-1}}(S(\mathbf{X}) \geqslant \gamma_t)$$

is given by

$$\widehat{c}_t = \frac{1}{N} \sum_{i=1}^{N} I\left\{S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t\right\} = \frac{N_t}{N},$$

where $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)} \sim g^*(\mathbf{x} \mid \gamma_{t-1}) \equiv g^*_{t-1}(\mathbf{x})$.

5. If $t = T$ go to Step 6; otherwise, set $t = t + 1$ and repeat from Step 2.

6. Deliver the estimator of the rare-event probability $\ell(\gamma)$:

$$\widehat{\ell}(\gamma) = \widehat{\ell}(\gamma_0) \prod_{t=1}^{T} \widehat{c}_t .$$

Remark 2.1 Step 2 of Algorithm 2.1 involves random resampling from the population $\mathcal{X}_{t-1}$ in order to obtain a new bootstrapped population of size N. We call this step the *bootstrap step* [8]. Step 3 involves sampling each component X_{ijk} from the minimum variance pdf g^* in (5) *conditional* on the other components — exactly as in the (systematic) Gibbs sampler [22]. We can therefore refer to the conditional sampling in Step 3 as a *Gibbs sampling step.* The conditional pdf can be written here explicitly as

$$g^*(x_{ijk} \,|\, \gamma_{t-1}, \mathbf{x}_{-ijk}) \propto f(x_{ijk}; \alpha, \lambda_{ijk})\, I\left\{ x_{ijk} > \gamma_{t-1} - \beta_{ijk} - \sum_{l \neq j} y_{il} \right\} ,$$

where $\mathbf{x}_{-ijk}$ denotes the vector $\mathbf{x}$ without the x_{ijk}-th component, y_{il} is given in (8), and

$$\begin{aligned}
\beta_{ij1} &= \min(x_{ij4}, x_{ij3} + x_{ij5}) \\
\beta_{ij2} &= \min(x_{ij5}, x_{ij3} + x_{ij4}) \\
\beta_{ij3} &= \min(x_{ij1} + x_{ij5}, x_{ij2} + x_{ij4}) \\
\beta_{ij4} &= \min(x_{ij1}, x_{ij2} + x_{ij3}) \\
\beta_{ij5} &= \min(x_{ij2}, x_{ij1} + x_{ij3}).
\end{aligned}$$

Note that all conditional pdfs are left-truncated Weibull pdfs, from which it is easy to generate using the inverse-transform method.

As a particular example, we compared the proposed method with the CE, VM, and screening variants listed in [21], for the case of a bridge network model with $m = 3$ and $n = 10$. The dimensionality of this problem is quite high, involving 150 variables. The model parameters are chosen as $\alpha = 1$, $u_{111} = u_{112} = u_{211} = u_{212} = u_{311} = u_{312} = 1$, and the rest of the $\{u_{ijk}\}$ are set to 4. We applied Algorithm 2.1 with $N = 40{,}000$, using the levels given in the third column of Table 1. How these levels were derived will be explained shortly; see Algorithm 2.2.

All other methods used a sample size $N = 500{,}000$ (see [19] for more details). Since the methods required different number of iterations or a preliminary run, to make a fairer comparison we allotted the same overall computational budget to each method. The total number of samples used by each method was approximately 2,800,000. The results are presented in Table 2, where the estimates of ℓ and the corresponding RE is estimated from 10 independent runs of each algorithm. It is thus important to emphasize that the RE is not estimated from a single simulation run, but rather from a batch of independent runs.

Table 1: The sequence of levels and the corresponding conditional probabilities for a typical run. The product of the conditional probabilities is here 5.88×10^{-8}.

t	γ_t	$\widehat{c}_t$
0	3.27	0.111
1	3.76	0.140
2	4.27	0.086
3	4.68	0.104
4	5.04	0.120
5	5.43	0.104
6	5.80	0.109
7	6	0.303

Table 2: Performance of the CE and VM methods and their screening counterparts together with the new approach over 10 independent trials on the 3×10 network model of dimension 150 with $\gamma = 6$. For all methods the total number of samples used is approximately 2,800,000. For all mathods the RE was estimated from the 10 independent trials.

Method	CE	VM	CE-SCR	VM-SCR	new method
mean $\widehat{\ell}$	2×10^{-8}	5.3×10^{-8}	5.3×10^{-8}	5.2×10^{-8}	5.92×10^{-8}
RE	1.06	0.28	0.33	0.15	0.021
mean iter.	6	6	7.4	8.9	7

It is clear from Table 2 that, for the given computational budget, the proposed method performs the best in terms of RE.

We now explain the method of computation of the sequence of levels given in Table 1. The sequence is determined adaptively in a way similar to the multilevel approach of the CE method.

Algorithm 2.2 (Adaptive Selection of Levels)

1. *Set a counter* $t = 1$. *Given the user-specified parameters* N_p *(sample size for preliminary calculation of levels) and* $\varrho \in (0,1)$ *(called* rarity parameter*), initialize by generating*

$$\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_{N_p}^{(0)} \sim f(\mathbf{x}).$$

Let $\widehat{\gamma}_0$ be the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(0)}),\ldots,S(\mathbf{X}_{N_p}^{(0)})$ and let $\widetilde{\mathscr{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)},\ldots,\widetilde{\mathbf{X}}_{N_0}^{(0)}\}$ be the subset of the population $\{\mathbf{X}_1^{(0)},\ldots,\mathbf{X}_{N_p}^{(0)}\}$ for which $S(\mathbf{X}_i^{(0)}) \geqslant \widehat{\gamma}_0$. Observe that $\widehat{\gamma}_0$ is a random variable, which approximates the true $(1-\varrho)$ quantile of the distribution of $S(\mathbf{X})$, where $\mathbf{X} \sim f(\mathbf{x})$.

2. *Same as in Algorithm 2.1, but note that here the sample is only approximately distributed from the target.*

3. *Same as in Algorithm 2.1, with the caveat that we do not claim to preserve stationarity.*

4. *Set*
$$\widehat{\gamma}_t = \min\{\gamma, \widehat{a}\},$$
where $\widehat{a}$ is the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(t)}),\ldots,S(\mathbf{X}_{N_p}^{(t)})$. Let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)},\ldots,\widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the subset of the population $\{\mathbf{X}_1^{(t)},\ldots,\mathbf{X}_{N_p}^{(t)}\}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \widehat{\gamma}_t$.

5. *If $\widehat{\gamma}_t = \gamma$, set $T = t$ and go to Step 6; otherwise, set $t = t+1$ and repeat from Step 2.*

6. *Deliver the estimated sequence of levels $\widehat{\gamma}_0,\ldots,\widehat{\gamma}_{T-1},\gamma$.*

Remark 2.2 Since for the bridge network problem $S(\mathbf{X})$ is a continuous function, the $(1-\varrho)$ sample quantile is unique. Therefore, in Algorithm 2.2 we have

$$\frac{1}{N}\sum_{i=1}^{N} I\left\{S(\mathbf{X}_i^{(t)}) \geqslant \widehat{\gamma}_t\right\} = \varrho$$

for all $t = 0,\ldots,T-1$, whenever $N_p \times \varrho$ is an integer. Observe that this is not true for $t = T$, because $\widehat{\gamma}_T = \gamma$ is not necessarily the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(T)}),\ldots,S(\mathbf{X}_{N_p}^{(T)})$.

The idea of the procedure described above is to select the first level $\widehat{\gamma}_0$ such that the event $\{S(\mathbf{X}) \geqslant \widehat{\gamma}_0\}$, where $\mathbf{X} \sim f$, is no longer a rare event and we could easily obtain samples approximately distributed from $g^*(\cdot \mid \widehat{\gamma}_0)$. The next level $\widehat{\gamma}_1$ is chosen such that $\{S(\mathbf{X}) \geqslant \widehat{\gamma}_1\}$, where $\mathbf{X} \sim g^*(\cdot \mid \widehat{\gamma}_0)$, is no longer a rare event and we could use the samples from $g^*(\cdot \mid \widehat{\gamma}_0)$ to help generate samples approximately distributed from $g^*(\cdot \mid \widehat{\gamma}_1)$. The sample from $g^*(\cdot \mid \widehat{\gamma}_1)$ is in its turn used to help generate a sample from $g^*(\cdot \mid \widehat{\gamma}_2)$ and so on. This recursive procedure is continued until we have generated approximate samples from the target $g_T^*(\cdot) = g^*(\cdot \mid \gamma)$. The final step provides a sequence $\{\widehat{\gamma}_t\}_{t=0}^{T}$ such that the conditional probabilities $\{c_t\}_{t=0}^{T-1}$ in Algorithm 2.1 are approximately ϱ.

Using Algorithm 2.2 with $N_p = 400$ and $\varrho = 0.1$, we obtained the sequence of levels in Table 1. Note that the computational cost of selecting a suitable sequence of levels is negligible. Observe that all the methods in Table 2 require preliminary runs before an estimate is delivered. For example, the CE and VM methods require a preliminary step in which the optimal parameter $\mathbf{v}$ in (4) is estimated. For the VM algorithm this estimation step involves calls to a costly non-linear minimization subroutine. In addition, the VM and CE screening counterparts require an additional step in which the dimension reduction of the likelihood ratio W is determined.

An advantage of the proposed method is that it requires few user-specified parameters. These are the sample sizes N, N_p and the rarity parameter ϱ in the computation of a suitable sequence of levels. Our numerical experience shows that the parameter ϱ requires little or no tuning for different rare-event probability estimation problems.

In summary, we employ a two-stage procedure in cases where a suitable sequence of levels is not obvious, where the first (preliminary) stage gives a suitable sequence of levels $\{\gamma_t\}_{t=0}^{T-1}$ as in Algorithm 2.2, and the second (main) stage provides a dependent but exact sample from the pdf $g^*(\cdot \mid \gamma)$ and a consistent estimate of the rare-event probability as in Algorithm 2.1.

2.2 Counting Example

Let $\mathscr{X}$ be the set of all permutations $\mathbf{x} = (x_1, \ldots, x_n)$ of the integers $1, \ldots, n$. Consider the problem of estimating the number of permutations in $\mathscr{X}$ for which $\sum_{j=1}^n j x_j \geqslant \gamma$. In other words, we are interested in estimating the size $|\mathscr{X}^*(\gamma)|$ of the set

$$\mathscr{X}^*(\gamma) = \left\{ \mathbf{x} \in \mathscr{X} \ : \ \sum_{j=1}^n j x_j \geqslant \gamma \right\}. \tag{10}$$

For example, we have that $|\mathscr{X}^*(\sum_{j=1}^n j^2)| = 1$. Observe also that $\mathscr{X}^*(0) \equiv \mathscr{X}$. To solve the above estimation problem, consider instead estimating the probability

$$\ell(\gamma) = \mathbb{P}\left(\sum_{j=1}^n j X_j \geqslant \gamma \right),$$

where $\mathbf{X} = (X_1, \ldots, X_n)$ is uniformly distributed over $\mathscr{X}$. This is exactly the setting of (1) with $S(\mathbf{x}) = \sum_{j=1}^n j x_j$. We now have

$$\ell(\gamma) = \frac{|\mathscr{X}^*(\gamma)|}{|\mathscr{X}|} = \frac{|\mathscr{X}^*(\gamma)|}{n!},$$

which is a rare-event probability for n large and γ close to $\sum_{j=1}^n j^2$. For example, $\ell\left(\sum_{j=1}^n j^2\right) = \frac{1}{n!}$. Hence, to estimate the size of the set $\mathscr{X}^*(\gamma)$, we need to estimate only the rare-event probability $\ell(\gamma)$. We employ the proposed method to estimate $\ell(\gamma)$, by sampling exactly from the zero-variance pdf

$$g^*(\mathbf{x} \,|\, \gamma) \propto I\{S(\mathbf{x}) \geqslant \gamma\}\,, \quad \mathbf{x} \in \mathscr{X}\,, \tag{11}$$

via a sequence of importance sampling pdfs $\{g_t^*(\mathbf{x})\}$, with $g_t^*(\mathbf{x}) = g_t^*(\mathbf{x} \,|\, \gamma_t) \propto I\{S(\mathbf{x}) \geqslant \gamma_t\}$. Note that the normalization constants for the $\{g_t^*(\mathbf{x})\}$ and for $g^*(\mathbf{x} \,|\, \gamma)$ are *unknown*. Our algorithm for this problem can be stated as follows.

Algorithm 2.3 (Algorithm for Counting) *Given the sequence of levels $\gamma_0 < \gamma_1 < \cdots < \gamma_{T-1} < \gamma$ and the user-specified sample size N, execute the following steps.*

1. *Set a counter $t = 1$. Generate N uniformly distributed permutations over $\mathscr{X}$. Denote the population of permutations by $\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_N^{(0)}$. Let $\widetilde{\mathscr{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)}, \ldots, \widetilde{\mathbf{X}}_{N_0}^{(0)}\}$, $N_0 > 0$ be the subset of the population $\{\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_N^{(0)}\}$ for which $S(\mathbf{X}_i^{(0)}) \geqslant \gamma_0$. Note that*

$$\widetilde{\mathbf{X}}_1^{(0)}, \ldots, \widetilde{\mathbf{X}}_{N_0}^{(0)} \sim g^*(\mathbf{x} \,|\, \gamma_0) \propto I\left\{\sum_{j=1}^{n} j x_j \geqslant \gamma_0\right\},\ \mathbf{x} \in \mathscr{X}$$

   and that an unbiased estimate for $c_0 = \ell(\gamma_0)$ is

$$\widehat{c}_0 = \widehat{\ell}(\gamma_0) = \frac{1}{N}\sum_{i=1}^{N} I\left\{S(\mathbf{X}_i^{(0)}) \geqslant \gamma_0\right\} = \frac{N_0}{N}.$$

2. *Sample uniformly with replacement N times from the population $\widetilde{\mathscr{X}}^{(t-1)}$ to obtain a new sample $\mathbf{Y}_1, \ldots, \mathbf{Y}_N$. Note that $\mathbf{Y}_1, \ldots, \mathbf{Y}_N \sim g^*(\mathbf{x} \,|\, \gamma_{t-1})$, since random resampling does not change the underlying distribution.*

3. *For each $\mathbf{Y} = (Y_1, \ldots, Y_n)$ in $\{\mathbf{Y}_1, \ldots, \mathbf{Y}_N\}$ repeat the following three steps n times:*

   (a) Draw a pair of indices (I, J) such that $I \neq J$ and both I and J are uniformly distributed over the integers $1, \ldots, n$.

   (b) Given $(I, J) = (i, j)$, generate the pair $(\widetilde{Y}_i, \widetilde{Y}_j)$ from the conditional bivariate pdf

$$g^*(\widetilde{y}_i, \widetilde{y}_j \,|\, \gamma_{t-1}, \{Y_k\}_{k \neq i,j}), \quad (\widetilde{y}_i, \widetilde{y}_j) \in \{(y_i, y_j), (y_j, y_i)\}.$$

   (c) Set $Y_i = \widetilde{Y}_i$ and $Y_j = \widetilde{Y}_j$, that is,

$$\mathbf{Y} = (Y_1, \ldots, \widetilde{Y}_i, \ldots, \widetilde{Y}_j, \ldots, Y_n).$$

Denote the resulting population of $\mathbf{Y}$*s by* $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)}$. *Again note that* $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)} \sim g^*(\mathbf{x} \,|\, \gamma_{t-1})$.

4. *Same as in Algorithm 2.1.*

5. *Same as in Algorithm 2.1.*

6. *Same as in Algorithm 2.1.*

Remark 2.3 In Step 3 we repeat the conditional sampling n times. Note that the algorithm will theoretically yield exact samples from the minimum variance pdf g_T^* even if we only update a single pair $(\widetilde{Y}_i, \widetilde{Y}_j)$. The performance of such a sampler might be poor due to undesirable correlation within the population. We can formally introduce an additional tuning parameter, say b, that counts the number of conditional sampling steps. We will refer to this parameter as the *burn-in* parameter. Note, however, that b does not play the same role as the *burn-in* period in standard MCMC sampling methods. The important difference is that, unlike MCMC algorithms, which have to discard any burn-in samples, the proposed algorithm can use the burn-in samples in the estimate of the conditional probabilities $\{c_t\}$. The reason, of course, is that our approach maintains stationarity with respect to the target density and thus yields exact samples for any value of b. The only rationale for using a large b is to reduce the correlation within the population $\widetilde{\mathscr{X}}^{(t-1)}$.

Remark 2.4 In Step 3 of Algorithm 2.3 we sample from the bivariate pdfs:

$$g^*(\widetilde{y}_i, \widetilde{y}_j \,|\, \gamma_{t-1}, \{y_k\}_{k \neq i,j}) \propto I\left\{ i\widetilde{y}_i + j\widetilde{y}_j \geqslant \gamma_{t-1} - \sum_{k \neq i,j} k y_k \right\}, \; i \neq j \;,$$

where $(\widetilde{y}_i, \widetilde{y}_j) \in \{(y_i, y_j), (y_j, y_i)\}$ and $i, j \in \{1, \ldots, n\}$. These are the conditional densities of the the minimum variance importance sampling pdf $g^*(\mathbf{x} \,|\, \gamma_{t-1})$. There are $\binom{n}{2}$ such bivariate conditional pdfs. Observe that the conditional sampling is similar to a single scan in the random Gibbs sampler [22]. A systematic Gibbs scan will sample from all the $\binom{n}{2}$ bivariate conditionals in a fixed order. In contrast, a random Gibbs scan will sample from a fixed number of conditional (not necessarily all of them) pdfs in a random order. Sampling a pair $(\widetilde{Y}_i, \widetilde{Y}_j)$ from $g^*(\widetilde{y}_i, \widetilde{y}_j \,|\, \gamma_{t-1}, \{y_k\}_{k \neq i,j})$ with $j > i$ is accomplished as follows. Draw a Bernoulli variable with success probability 1/2, say $B \sim \mathsf{Ber}(1/2)$. If $B = 1$ and $S([y_1, \ldots, \widetilde{y}_j, \ldots, \widetilde{y}_i, \ldots, y_n]) \geqslant \gamma_{t-1}$, set $\widetilde{Y}_i = \widetilde{y}_j$ and $\widetilde{Y}_j = \widetilde{y}_i$, otherwise set $\widetilde{Y}_i = \widetilde{y}_i$ and $\widetilde{Y}_j = \widetilde{y}_j$.

Remark 2.5 The additive structure of the objective function $S(\mathbf{x})$ allows for its efficient evaluation. More specifically, suppose we are given a permutation $\mathbf{x}$ and know $S(\mathbf{x})$. Suppose further that a single run through (a), (b) and (c) in

Step 3 of the algorithm exchanges the i-th and j-th element of the permutation $\mathbf{x}$ to obtain $\tilde{\mathbf{x}}$. Then, to compute the score at the new value $\tilde{\mathbf{x}}$, set $S(\tilde{\mathbf{x}}) = S(\mathbf{x}) - (i-j)x_i - (j-i)x_j$.

As an example, consider the case where $n = 32$ and $\gamma = \sum_{j=1}^{n} j^2$, with $N = 10^4$ and the sequence of levels given in Table 3. We obtained $\widehat{\ell} = 3.64 \times 10^{-36}$ with an estimated RE of 5% (here $\ell = \frac{1}{32!} \approx 3.8 \times 10^{-36}$) using ten independent runs of Algorithm 2.3. This gives the estimate $\widehat{|\mathscr{X}^*|} \approx .96$, whereas the exact value for $|\mathscr{X}^*|$ is 1. The computational cost of Algorithm 2.3 is equivalent to the generation of 20×10^5 permutations.

Table 3: The sequence of levels and the corresponding conditional probabilities for a typical run. The product of the conditional probabilities is here 3.64×10^{-36}.

t	γ_t	$\widehat{c}_t$	t	γ_t	$\widehat{c}_t$
0	9828	0.01033	10	11403	0.01377
1	10413.5	0.00948	11	11414	0.01779
2	10743	0.01206	12	11422	0.01799
3	10964	0.01111	13	11428	0.01631
4	11107.5	0.01116	14	11432	0.02528
5	11209	0.00976	15	11435	0.02236
6	11281	0.00812	16	11437	0.0336
7	11330	0.00899	17	11438	0.11123
8	11364	0.00932	18	11439	0.06579
9	11387	0.01198	19	11440	0.03128

The sequence of levels in Table 3 was estimated via the following procedure with $N_p = 10^4$ and $\varrho = 0.01$.

Algorithm 2.4 (Approximate Selection of Levels)

1. *Set a counter* $t = 1$. *Given the sample size* N_p *and the rarity parameter* $\varrho \in (0,1)$, *initialize by generating* N_p *uniformly distributed permutations over* $\mathscr{X}$.

   Denote the population of permutations by $\mathbf{X}_1^{(0)}, \dots, \mathbf{X}_N^{(0)}$. *Let* $\widehat{\gamma}_0$ *be the* $(1-\varrho)$ *sample quantile of* $S(\mathbf{X}_1^{(0)}), \dots, S(\mathbf{X}_{N_p}^{(0)})$ *and let* $\widetilde{\mathscr{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)}, \dots, \widetilde{\mathbf{X}}_{N_0}^{(0)}\}$ *be the subset of the population* $\{\mathbf{X}_1^{(0)}, \dots, \mathbf{X}_{N_p}^{(0)}\}$ *for which* $S(\mathbf{X}_i^{(0)}) \geqslant \widehat{\gamma}_0$.

2. *Sample uniformly with replacement* N_p *times from the population* $\widetilde{\mathscr{X}}^{(t-1)}$ *to obtain a new sample* $\mathbf{Y}_1, \dots, \mathbf{Y}_{N_p}$.

3. *Same as in Algorithm 2.3, but note that here the resulting sample is only approximately distributed from $g^*(\mathbf{x} \,|\, \widehat{\gamma}_{t-1})$.*
4. *Same as in Algorithm 2.2.*
5. *Same as in Algorithm 2.2.*
6. *Same as in Algorithm 2.2.*

Remark 2.6 Unlike in the previous example, $S(\mathbf{X})$ is not a continuous function and therefore the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(t)}), \ldots, S(\mathbf{X}_{N_p}^{(t)})$ may not be unique. Hence, it is not necessarily true that

$$\frac{1}{N_p} \sum_{i=1}^{N_p} I\left\{S(\mathbf{X}_i^{(t)}) \geqslant \widehat{\gamma}_t\right\} = \varrho$$

for all $t = 0, \ldots, T-1$, even when $N_p \times \varrho$ is an integer.

As a different example consider the case where $n = 10$ and $\gamma = 375$. We ran Algorithm 2.3 ten independent times with $N = 10^3$ and the sequence of levels given in Table 4. Each run required four iterations and thus the total computational cost of Algorithm 2.3 is approximately equivalent to the generation of $4 \times 10 \times 10^3 = 4 \times 10^4$ random permutations. The average over the ten trials gave the estimate $\widehat{|\mathscr{X}^*|} = 2757$ with a RE of 5%. Using full enumeration, the true value was found to be $|\mathscr{X}^*| = 2903$.

Table 4: The sequence of levels and the corresponding conditional probabilities for a typical run. The sequence was determined using Algorithm 2.4 with $N_p = 10^3$ and $\varrho = 0.1$. The product of the conditional probabilities here is 7.97×10^{-4}.

t	γ_t	$\widehat{c}_t$
0	339	0.0996
1	362	0.1161
2	373	0.1222
3	375	0.5643

2.3 Combinatorial Optimization Example

Combinatorial optimization has always been an important and challenging part of optimization theory. A well-known instance of a difficult combinatorial optimization problem is the 0-1 knapsack problem, defined as:

$$\begin{aligned}
&\max_{\mathbf{x}} \sum_{j=1}^{n} p_j x_j, \qquad x_i \in \{0,1\},\\
&\text{subject to}: \sum_{j=1}^{n} w_{ij} x_j \leqslant c_i, \quad i = 1,\ldots,m.
\end{aligned} \tag{12}$$

Here $\{p_i\}$ and $\{w_{ij}\}$ are positive weights and $\{c_i\}$ are positive cost parameters. To make (12) easier to handle as an estimation problem, we note that (12) is equivalent to $\max\limits_{\mathbf{x}\in\{0,1\}^n} S(\mathbf{x})$, where $\mathbf{x} = (x_1,\ldots,x_n)$ is a binary vector and

$$S(\mathbf{x}) = C(\mathbf{x}) + \sum_{j=1}^{n} p_j x_j = \alpha \sum_{i=1}^{m} \min\left\{c_i - \sum_{j=1}^{n} w_{ij} x_j, 0\right\} + \sum_{j=1}^{n} p_j x_j,$$

with $\alpha = (1+\sum_{j=1}^{n} p_j)\big/\max_{i,j}\{c_i - w_{ij}\}$. Note that the constant α is such that if $\mathbf{x}$ satisfies all the constraints in (12), then $C(\mathbf{x}) = 0$ and $S(\mathbf{x}) = \sum_{j=1}^{n} p_j x_j \geqslant 0$. Alternatively, if $\mathbf{x}$ does not satisfy all of the constraints in (12), then $C(\mathbf{x}) \leqslant -(1+\sum_{j=1}^{n} p_j x_j)$ and $S(\mathbf{x}) \leqslant -1$. To this optimization problem we can associate the problem of estimating the rare-event probability

$$\ell(\gamma) = \mathbb{P}\left(S(\mathbf{X}) \geqslant \gamma\right), \quad \gamma \in \left[0, \sum_{j=1}^{n} p_j\right],$$

where $\mathbf{X}$ is a vector of independent Bernoulli random variables with success probability 1/2. An important difference in the optimization setting is that we are not interested *per se* in obtaining an unbiased estimate for the rare-event probability. Rather we only wish to approximately sample from the pdf $g^*(\mathbf{x}\,|\,\gamma) \propto I\{S(\mathbf{x}) \geqslant \gamma\}$ for as large a value of γ as the algorithm finds possible. Given this objective, we combine the preliminary stage, in which a suitable sequence of levels is determined, with the main estimation stage into a single algorithm.

Algorithm 2.5 (Knapsack Optimization)

1. *Set a counter $t = 1$. Given the sample size N and the rarity parameter $\varrho \in (0,1)$, initialize by generating N uniform binary vectors of dimension n. Denote the population of binary vectors by $\mathbf{X}_1^{(0)},\ldots,\mathbf{X}_N^{(0)}$. Let $\widehat{\gamma}_0$ be the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(0)}),\ldots,S(\mathbf{X}_N^{(0)})$ and let $\widetilde{\mathscr{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)},\ldots,\widetilde{\mathbf{X}}_{N_0}^{(0)}\}$ be the subset of the population $\{\mathbf{X}_1^{(0)},\ldots,\mathbf{X}_N^{(0)}\}$ for which $S(\mathbf{X}_i^{(0)}) \geqslant \widehat{\gamma}_0$. Then,*

$$\widetilde{\mathbf{X}}_1^{(0)},\ldots,\widetilde{\mathbf{X}}_{N_0}^{(0)} \overset{approx.}{\sim} g^*(\mathbf{x}\,|\,\widehat{\gamma}_0) \propto I\left\{S(\mathbf{x}) \geqslant \widehat{\gamma}_0\right\}.$$

2. *Sample uniformly with replacement N times from the population $\widetilde{\mathscr{X}}^{(t-1)}$ to obtain a new sample $\mathbf{Y}_1, \ldots, \mathbf{Y}_N$.*

3. *For each $\mathbf{Y} = (Y_1, \ldots, Y_n)$ in $\{\mathbf{Y}_1, \ldots, \mathbf{Y}_N\}$, generate $\widetilde{\mathbf{Y}} = (\widetilde{Y}_1, \ldots, \widetilde{Y}_n)$ as follows:*

   (a) Draw $\widetilde{Y}_1$ from the conditional pdf $g^*(y_1 \,|\, \widehat{\gamma}_{t-1}, Y_2, \ldots, Y_n)$.

   (b) Draw $\widetilde{Y}_i$ from $g^*(y_i \,|\, \widehat{\gamma}_{t-1}, \widetilde{Y}_1, \ldots, \widetilde{Y}_{i-1}, Y_{i+1}, \ldots, Y_n)$, $\quad i = 2, \ldots, n-1$.

   (c) Draw $\widetilde{Y}_n$ from $g^*(y_n \,|\, \widehat{\gamma}_{t-1}, \widetilde{Y}_1, \ldots, \widetilde{Y}_{n-1})$.

   Denote the resulting population of $\mathbf{Y}$s by $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)}$.

4. *Let $\widehat{\gamma}_t$ be the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(t)}), \ldots, S(\mathbf{X}_N^{(t)})$. Let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)}, \ldots, \widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the subset of the population $\{\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)}\}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \widehat{\gamma}_t$. Again, we have*

$$\widetilde{\mathbf{X}}_1^{(t)}, \ldots, \widetilde{\mathbf{X}}_{N_t}^{(t)} \overset{approx.}{\sim} g^*(\mathbf{x} \,|\, \widehat{\gamma}_t).$$

5. *If there is no progress in increasing γ over a number of iterations, that is, if $\widehat{\gamma}_t = \widehat{\gamma}_{t-1} = \cdots = \widehat{\gamma}_{t-s}$ for some user-specified positive integer s, set $T = t$ and go to Step 6; otherwise set $t = t + 1$ and repeat from Step 2.*

6. *Deliver the vector $\mathbf{x}^*$ from the set*

$$\mathbf{X}_1^{(T)}, \ldots, \mathbf{X}_N^{(T)}$$

   for which $S(\mathbf{X}_i^{(T)})$ is maximal as an estimate for the global maximizer of (12).

Remark 2.7 The conditionals of the minimum variance importance sampling pdf in Step 3 can be written as:

$$g^*(y_i \,|\, \widehat{\gamma}_{t-1}, \mathbf{y}_{-i}) \propto I\left\{C(\mathbf{y}) + p_i y_i \geqslant \widehat{\gamma}_{t-1} - \sum_{j \neq i} p_j y_j\right\},$$

where $\mathbf{y}_{-i}$ denotes the vector $\mathbf{y}$ with the i-th element removed. Sampling a random variable $\widetilde{Y}_i$ from such a conditional can be accomplished as follows. Draw $B \sim \mathsf{Ber}(1/2)$, if $S([y_1, \ldots, y_{i-1}, B, y_{i+1}, \ldots, y_n]) \geqslant \widehat{\gamma}_{t-1}$, then set $\widetilde{Y}_i = B$, otherwise set $\widetilde{Y}_i = 1 - B$.

As a particular example, consider the *Sento1.dat* knapsack problem given in

`http : //people.brunel.ac.uk/ mastjjb/jeb/orlib/files/mknap2.txt`

The problem has 30 constraints and 60 variables. We selected $\varrho = 0.01$ and $N = 10^3$, and the algorithm was stopped after no progress was observed ($d = 1$). We ran the algorithm ten independent times. The algorithm always found the optimal solution. A typical evolution of the algorithm is given in Table 5. The total sample size used is 10^4 which is about 8.67×10^{-15} of the total effort needed for the complete enumeration of the 2^{60} possible binary vectors.

Table 5: Typical evolution of Algorithm 2.5. The maximum value for this problem is 6704.

t	$\widehat{\gamma}_t$	t	$\widehat{\gamma}_t$
1	-5708.55	6	6051.74
2	1296.94	7	6346.32
3	3498.60	8	6674.92
4	4567.91	9	6704
5	5503.22	10	6704

3 Main Method

In this section we present a quite general version of the proposed algorithm, which can be applied to a wide variety of rare-event probability estimation, combinatorial optimization, and counting problems. Recall that to estimate the rare-event probability (1), we would like to completely specify or at least sample from the minimum variance importance sampling pdf (5). The main idea of the method is to generate an exact sample from (5) and estimate its normalizing constant $\ell(\gamma)$ simultaneously. To achieve this goal, we sample recursively from the sequence of minimum variance importance sampling pdfs: $\{g_t^*\} \equiv \{g^*(\cdot \,|\, \gamma_t)\}$, where each of the pdfs is associated with a given level set γ_t such that $\gamma_0 < \gamma_1 < \ldots < \gamma_{T-1} < \gamma_T = \gamma$. Initially, we aim to obtain an exact sample from g_0^*, where the level γ_0 is chosen in advance via a preliminary simulation run such that exact sampling from g_0^* is viable using the acceptance-rejection method [22] with proposal f. Note that we can interpret the sample from g_0^* as an empirical approximation to g_0^*, from which a kernel density estimator [3] of g_0^* can be constructed. We can then use the sample from g_0^* (or rather the kernel density approximation based on this sample) to help generate exact samples from g_1^* using the acceptance-rejection method with the kernel density approximation as proposal. Once we have an exact sample from g_1^*, we use it to construct a kernel density approximation to g_1^*, which in its turn will help generate an exact sample from g_2^* and so on. This recursive process is continued until we generate from the final desired minimum variance importance sampling pdf $g_T^* = g^*(\cdot \,|\, \gamma)$. As we have seen, in many cases it is straightforward to generate exact samples from g_t^* using a suitable Markov chain, provided that one already has samples from g_{t-1}^*. In this kernel density approximation interpretation, we choose the kernel of the density estimator to be a Markov transition kernel with stationary distribution g_{t-1}^*. In more generality and detail the steps of the algorithm are given below. We assume a suitable sequence of levels is given. Later on we will present a procedure which can be used to construct such a sequence. Figure 2 shows a simple example illustrating the workings of the algorithm.

Algorithm 3.1 (Main Algorithm for Estimation)
Given the sample size N and a suitable sequence of levels $\{\gamma_t\}_{t=0}^T$, execute the following steps.

1. **Initialization.** *Generate a sample $\mathscr{X}^{(0)} = \{\mathbf{X}_1^{(0)}, \dots, \mathbf{X}_N^{(0)}\}$ of size N (user specified) from the nominal pdf f. Let $\widetilde{\mathscr{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)}, \dots, \widetilde{\mathbf{X}}_{N_0}^{(0)}\}$, $N_0 > 0$ be the largest subset of $\mathscr{X}^{(0)}$ for which $S(\mathbf{X}_i^{(0)}) \geqslant \gamma_0$. We have thus generated a sample*
$$\widetilde{\mathbf{X}}_1^{(0)}, \dots, \widetilde{\mathbf{X}}_{N_0}^{(0)} \sim g^*(\cdot \,|\, \gamma_0)$$
using the acceptance-rejection method with proposal f. The normalizing constant of $g^*(\cdot \,|\, \gamma_0)$ is therefore straightforward to estimate:
$$\widehat{\ell}(\gamma_0) = \frac{1}{N} \sum_{i=1}^{N} I\{S(\mathbf{X}_i^{(0)}) \geqslant \gamma_0\} = \frac{N_0}{N}.$$

2. **Bootstrap sampling and Markov kernel smoothing.** *(To be iterated from $t = 1$.) Given the exact sample $\widetilde{\mathscr{X}}^{(t-1)}$ from g_{t-1}^*, construct the kernel density estimator:*
$$\pi_{t-1}(\mathbf{x} \,|\, \widetilde{\mathscr{X}}^{(t-1)}) = \frac{1}{N_t} \sum_{i=1}^{N_t} \kappa\left(\mathbf{x}, \widetilde{\mathbf{X}}_i^{(t-1)}\right). \tag{13}$$
Here, κ is the transition density of a suitable Markov chain starting from $\widetilde{\mathbf{X}}_i^{(t-1)}$ and with stationary pdf g_{t-1}^*. The aim is to use π_{t-1} to help generate exact samples from the next target pdf g_t^*. This is accomplished as follows. Generate a new population $\mathscr{X}^{(t)} = \{\mathbf{X}_1^{(t)}, \dots, \mathbf{X}_N^{(t)}\}$ such that
$$\mathbf{X}_1^{(t)}, \dots, \mathbf{X}_N^{(t)} \sim \pi_{t-1}. \tag{14}$$
Let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)}, \dots, \widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the largest subset of $\mathscr{X}^{(t)}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t$. Then, we have that
$$\widetilde{\mathbf{X}}_1^{(t)}, \dots, \widetilde{\mathbf{X}}_{N_t}^{(t)} \sim g_t^*$$
and that an unbiased estimate of the conditional probability
$$c_t = \mathbb{P}(S(\mathbf{X}) > \gamma_t \,|\, S(\mathbf{X}) > \gamma_{t-1}), \quad t = 0, 1, \dots, T, \quad \gamma_{-1} = -\infty,$$

is given by

$$\widehat{c}_t = \frac{1}{N}\sum_{i=1}^{N} I\{S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t\} = \frac{N_t}{N}. \tag{15}$$

Therefore, by the product rule of probability theory, an estimate for the normalizing constant of g_t^* is

$$\widehat{\ell}(\gamma_t) = \prod_{k=0}^{t} \widehat{c}_k. \tag{16}$$

3. Stopping Condition. *If $\gamma_t = \gamma_T$, go to Step 3; otherwise, set $t = t+1$ and repeat from Step 2.*

4. Final Estimate. *Deliver the consistent estimate:*

$$\widehat{\ell}(\gamma) = \prod_{t=0}^{T} \widehat{c}_t \tag{17}$$

of $\ell(\gamma)$ — the normalizing constant of $g^*(\cdot \,|\, \gamma)$. The validity of this estimate follows from the product rule of probability theory, namely,

$$\begin{aligned}\ell(\gamma) &= \mathbb{P}(S(\mathbf{X}) \geqslant \gamma_0)\prod_{t=1}^{T}\mathbb{P}(S(\mathbf{X}) \geqslant \gamma_t \,|\, S(\mathbf{X}) \geqslant \gamma_{t-1}), \quad \mathbf{X} \sim f \\ &= \mathbb{E}_f I\{S(\mathbf{X}) \geqslant \gamma_0\}\prod_{t=1}^{T}\mathbb{E}_{g_{t-1}^*} I\{S(\mathbf{X}) \geqslant \gamma_t\}.\end{aligned} \tag{18}$$

Remark 3.1 (Generation from kernel density estimator) Generating a sample from π_{t-1} in (14) is easy. Namely, we choose a point in $\widetilde{\mathscr{X}}^{(t-1)}$ at random, say $\widetilde{\mathbf{X}}_I^{(t-1)}$, and then sample from the Markov transition density $\kappa(\,\cdot\,, \widetilde{\mathbf{X}}_I^{(t-1)})$. For example, the kernel used for the problem described in Figure 2 is

$$\kappa((x,y),(x',y')) = g^*(x \,|\, \gamma_{t-1}, y')g^*(y \,|\, \gamma_{t-1}, x).$$

In other words, the kernel changes the initial state (x', y') to (x, y') by sampling from $g^*(x \,|\, \gamma_{t-1}, y')$ and then (x, y') is changed to (x, y) by sampling from $g^*(y \,|\, \gamma_{t-1}, x)$. This kernel was chosen due to the simplicity of the conditionals of $g^*(x, y \,|\, \gamma)$. There are, of course, other possible choices for the kernel. Observe that since the transition density κ has stationary distribution g_{t-1}^*, we have

$$\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_N^{(t)} \sim g_{t-1}^*.$$

Thus it is important to realize that the only rationale for applying the Markov kernel within our algorithm is not to achieve asymptotic stationarity toward the pdf g_{t-1}^* as in standard MCMC methods, but only to increase the diversity (that is, reduce the correlation) within the population $\widetilde{\mathscr{X}}^{(t-1)}$.

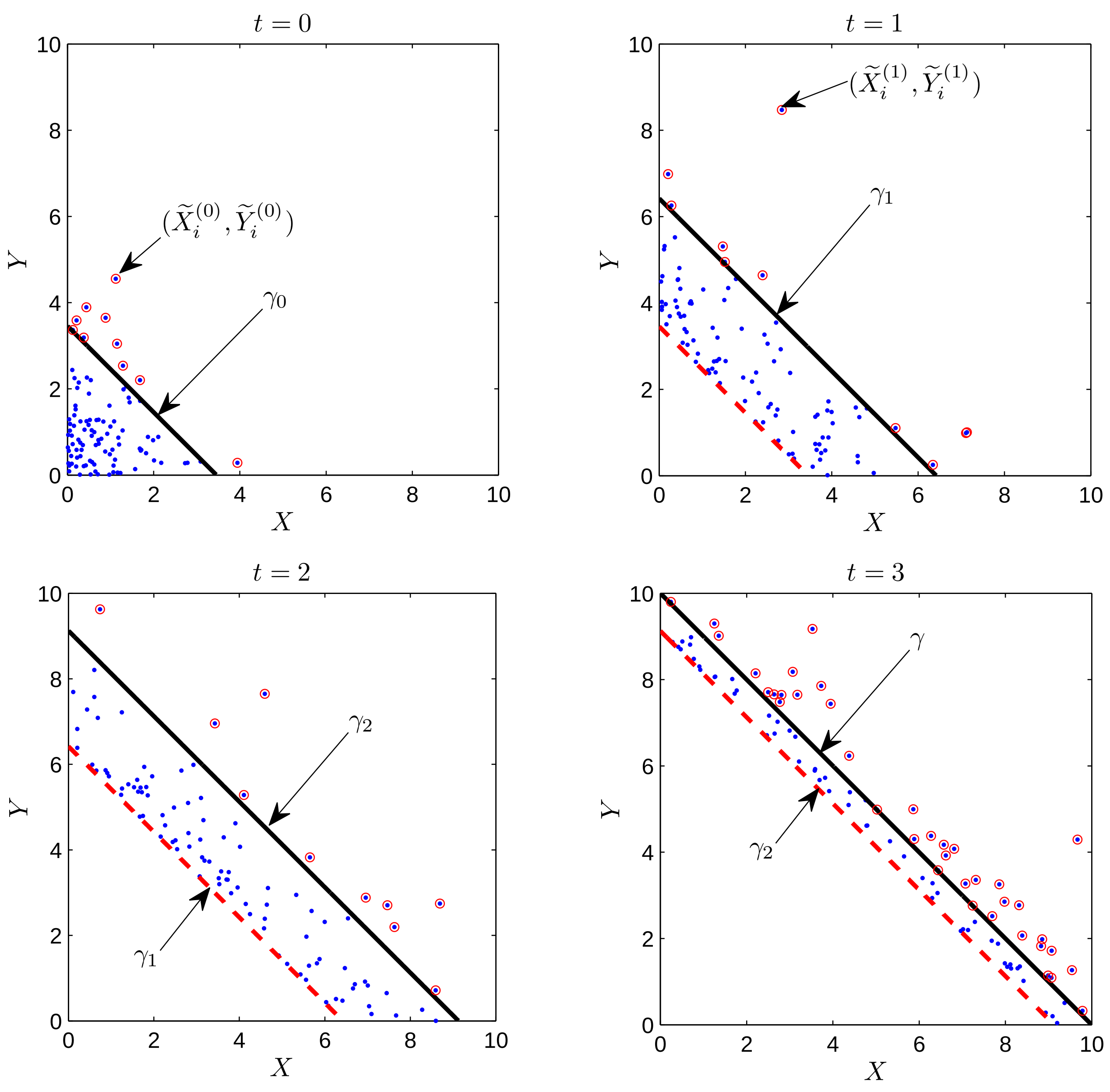

Figure 2: Illustration of the workings of the proposed algorithm on the problem of sampling from $g^*(x, y \mid \gamma) = \mathrm{e}^{-x-y} I\{x + y \geqslant \gamma\}/\ell(\gamma)$, $x, y > 0$ and estimating $\ell(\gamma)$ with $\gamma = 10$. The sample size is $N = 100$ and the sequence of levels is 3.46, 6.42, 9.12 and 10. The top left figure ($t = 0$) shows the initial 100 points from the nominal pdf f and a segment of the straight line $x + y = \gamma_0$. The points above this line segment (encircled) belong to the population $\widetilde{\mathcal{X}}^{(0)}$, which has pdf $g^*(\cdot; \gamma_0)$. These points help to generate the population $\mathcal{X}^{(1)}$ of 100 points depicted on the next Figure ($t = 1$), still with pdf g_0^*. The encircled points above the threshold γ_1 have pdf g_1^* and they help us to construct a kernel density estimator and sample from g_2^*. Continuing this recursive process, we eventually generate points above the threshold γ, as in the bottom right Figure ($t = 3$). There all the points have pdf g_2^*. The points above $\gamma = 10$ (encircled) are drawn exactly from the target pdf $g^*(x, y \mid \gamma)$.

Remark 3.2 (Consistency of $\widehat{\ell}$) The $\{\widehat{c}_i\}$ are unbiased estimators of the conditional probabilities $\{c_i\}$. The estimators are, however, dependent and therefore the final product estimator (17) is only asymptotically unbiased.

Remark 3.3 (Burn-in and correlation) It may be desirable that the population from the kernel approximation π_{t-1} is iid, but in practice the Markov structure of the kernel κ will make the samples in the population correlated. This correlation could have averse effect on the performance of the sampler. In such cases it may be beneficial to apply the transition kernel a number times, say b, in order to reduce the correlation amongst the samples and to ensure that the sampler is exploring the sample space properly. Thus the parameter b can be interpreted as a burn-in parameter. Note, however, that the burn-in parameter in Algorithm 3.1 is not related to achieving stationarity as in standard MCMC, only reducing the correlation structure of the bootstrap sample.

The algorithm assumes that we have a suitable sequence of levels such that the conditional probabilities

$$c_t = \mathbb{P}(S(\mathbf{X}) > \gamma_t \,|\, S(\mathbf{X}) > \gamma_{t-1}), \quad t = 0, 2, \ldots, T, \quad \mathbf{X} \sim f(\mathbf{x}),$$

are not too small. In many cases it will not be obvious how to construct such a sequence a priori, but we can always use a pilot simulation run to estimate a suitable sequence $\{\widehat{\gamma}_t\}_{t=0}^{T}$ at negligible computational cost. The idea is to sample from the sequence of pdfs $g^*(\mathbf{x} \,|\, \widehat{\gamma}_t),\ t = 1, \ldots, T$, where $\widehat{\gamma}_t$ is an estimate of the true γ_t, which has been plugged into $g^*(\mathbf{x} \,|\, \cdot)$. Here γ_t is the $(1-\varrho)$ percentile of the distribution of $S(\mathbf{X})$ given that $S(\mathbf{X}) > \gamma_{t-1},\ \mathbf{X} \sim f(\mathbf{x})$. This approach is sometimes known as the *plug-in* approach [8].

Algorithm 3.2 (Estimation of Levels) *Given the sample size N_p for the preliminary simulation run and the user-specified rarity parameter ϱ, execute the following steps.*

1. **Initialization.** *Set $t = 1$. Generate a sample $\mathcal{X}^{(0)} = \{\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_{N_p}^{(0)}\}$ of size N_p from the nominal pdf f. Let $\widehat{\gamma}_0$ be the $(1-\varrho)$ sample quantile of the population $S(\mathbf{X}_1^{(0)}), \ldots, S(\mathbf{X}_{N_p}^{(0)})$. Again let $\widetilde{\mathcal{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)}, \ldots, \widetilde{\mathbf{X}}_{N_0}^{(0)}\}$, $N_0 > 0$ be the set of points in $\mathcal{X}^{(0)}$ for which $S(\mathbf{X}_i^{(0)}) \geqslant \widehat{\gamma}_0$.*

2. **Bootstrap sampling and kernel smoothing.** *Given the population $\widetilde{\mathcal{X}}^{(t-1)}$, construct the kernel density estimator π_{t-1} of g_{t-1}^* given in (13), and generate a new population $\mathcal{X}^{(t)} = \{\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_{N_p}^{(t)}\}$ such that*

$$\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_{N_p}^{(t)} \sim \pi_{t-1}.$$

Let

$$\widehat{\gamma}_t = \min(\widehat{a}, \gamma), \tag{19}$$

where $\widehat{a}$ is the $(1-\varrho)$ sample quantile of $S(\mathbf{X}_1^{(t)}),\ldots,S(\mathbf{X}_{N_p}^{(t)})$. In other words, $\widehat{a}$ is an estimate of the root of the equation

$$\mathbb{P}(S(\mathbf{X}) \geqslant a) = \varrho, \quad \mathbf{X} \sim \pi_{t-1}.$$

Let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)},\ldots,\widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the set of points in $\mathscr{X}^{(t)}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t$.

3. **Estimate of sequence of levels.** *If $\widehat{\gamma}_t = \gamma$, set $T = t$ and deliver the sequence of levels*

$$\widehat{\gamma}_0,\ldots,\widehat{\gamma}_{T-1},\gamma;$$

otherwise, set $t = t+1$ repeat from Step 2.

Remark 3.4 (Optimization) After a minor modification, we can use Algorithm 3.2 not only to estimate a sequence of levels, but also for optimization. For continuous and discrete optimization problems we are not typically interested in estimating ℓ, but in increasing γ as much as possible. Thus Step 3 above is modified as follows.

3. **Estimate of extremum.** *If $\widehat{\gamma}_t = \widehat{\gamma}_{t-1} = \cdots = \widehat{\gamma}_{t-s}$ for some user-specified positive integer s, set $T = t$ and deliver the vector $\mathbf{x}^*$ from the set*

$$\mathbf{X}_1^{(T)},\ldots,\mathbf{X}_N^{(T)}$$

for which $S(\mathbf{X}_i^{(T)})$ is maximal as an estimate for the global maximizer of $S(\mathbf{x})$; otherwise set $t = t+1$ and repeat from Step 2.

Note that we will use N_p to denote the sample size when the algorithm is used for the construction of a suitable sequence of levels, otherwise we will use N.

For some optimization problems, it can be beneficial to keep track of the best solution vector overall, across all the iterations of the algorithm.

Remark 3.5 Due to the way we determine the level sets $\{\widehat{\gamma}_t\}_{t=0}^{T-1}$, we have

$$\mathbb{P}(S(\mathbf{X}) \geqslant \widehat{\gamma}_t \,|\, S(\mathbf{X}) \geqslant \widehat{\gamma}_{t-1}) \approx \varrho,\ \forall t = 1,\ldots,T-1,$$

with exact equality only in continuous problems (assuming $N \times \varrho$ is an integer). Note, however, that the last conditional probability $\mathbb{P}(S(\mathbf{X}) \geqslant \gamma \,|\, S(\mathbf{X}) \geqslant \widehat{\gamma}_{T-1})$, $\mathbf{X} \sim f$ may be quite different from ϱ.

Remark 3.6 If the target density is

$$g^*(\mathbf{x} \,|\, \gamma) \propto f(\mathbf{x}) I\{S(\mathbf{x}) \leqslant \gamma\},$$

that is, the inequality within the indicator function is reversed, then Algorithm 3.2 is modified as follows. Instead of taking the $(1-\varrho)$ sample quantile, we take the ϱ-th sample quantile of the population of scores in Step 1 and 2. All instances of $S(\cdot) \geqslant \cdot$ must be replaced with $S(\cdot) \leqslant \cdot$, and $\widehat{\gamma}_t = \min(\widehat{a},\gamma)$ in (19) is replaced with $\widehat{\gamma}_t = \max(\widehat{a},\gamma)$.

3.1 The Holmes-Diaconis-Ross Algorithm

As pointed out in the introduction, the idea of using the product rule of probability theory to estimate rare-event probabilities has been widely used by the simulation community, in particular, Holmes and Diaconis [6], and Ross [15] use it for discrete counting problems. The Holmes-Diaconis-Ross algorithm (HDR) reads as follows. Suppose we are given a suitable sequence of levels $\{\gamma_t\}_{t=-1}^{T}$ (with $\gamma_{-1} = -\infty$) and a sample size b. For each $t \in \{0, \ldots, T\}$, find a point $\mathbf{Y}$ such that $S(\mathbf{Y}) \geqslant \gamma_{t-1}$. Let $\mathbf{Y}_1, \ldots, \mathbf{Y}_b$ be the output of b consecutive steps of a suitable Gibbs or Metropolis-Hastings sampler, starting from $\mathbf{Y}$ and with stationary pdf $g^*(\mathbf{y} \,|\, \gamma_{t-1})$. Use $\widehat{c}_t = \frac{1}{b}\sum_{i=1}^{b} I\{S(\mathbf{Y}_i) \geqslant \gamma_t\}$ as an approximation to the conditional probability $c_t = \mathbb{P}_f(S(\mathbf{Y}) \geqslant \gamma_t \,|\, S(\mathbf{Y}) \geqslant \gamma_{t-1})$. Deliver the final estimate $\widehat{\ell}(\gamma_T) = \prod_{t=0}^{T} \widehat{c}_t$. More formally, the approach is summarized in the following algorithm.

Algorithm 3.3 (HDR Algorithm for Rare-event Probability Estimation)
Given a sequence of levels $\{\gamma_t\}_{t=0}^{T}$ ($\gamma_{-1} = -\infty$) and the sample size b, for each $t \in \{0, 1, \ldots, T\}$ execute the following steps.

1. *Construct an arbitrary point $\mathbf{Y}^{(0)} = (Y_1^{(0)}, \ldots, Y_n^{(0)})$ such that $S(\mathbf{Y}^{(0)}) \geqslant \gamma_{t-1}$.*

2. *For $m = 1, \ldots, b$, generate a discrete uniform random variable, I, on the set $\{1, \ldots, n\}$, and set $\mathbf{Y}^{(m)} = (Y_1^{(m-1)}, \ldots, Y_I, \ldots, Y_n^{(m-1)})$, where Y_I is drawn from the conditional pdf*

$$g^*(y_I \,|\, \gamma_{t-1}, \mathbf{Y}_{-I}^{(m-1)}).$$

   Here $\mathbf{Y}_{-I}^{(m-1)}$ is the same as $\mathbf{Y}^{(m-1)}$, but with $Y_I^{(m-1)}$ removed.

3. *Calculate the ergodic estimate*

$$\widehat{c}_t = \frac{1}{b}\sum_{m=1}^{b} I\left\{S(\mathbf{Y}^{(m)}) \geqslant \gamma_t\right\}.$$

Algorithm 3.3 delivers the collection of independent estimates $\{\widehat{c}_t\}_{t=0}^{T}$, so that a natural estimate for the rare-event probability is $\widehat{\ell}(\gamma_T) = \prod_{t=0}^{T} \widehat{c}_t$.

The HDR method differs from our approach in the following aspects.

1. Although the HDR methods requires the user to supply a sequence of levels such that $\{c_t\}$ are not too small, it does not provide a practical method for the automatic construction of such a sequence. The sequence is thus chosen in an ad hoc way. In contrast, Algorithm 3.2 delivers a sequence of levels for any value of the rarity-parameter $\varrho \in (0, 1)$.

2. In the HDR algorithm each conditional probability c_t is estimated using the output from a single Markov chain. In contrast, our method runs a population of Markov chains, implicitly defined via the kernel density estimator (13).

3. The HDR algorithm does not achieve *exact* stationarity with respect to $g^*(\mathbf{x} \mid \gamma_{t-1})$, because, for each $t \in \{0, \ldots, T\}$, the Markov chain in Step 2 is started from an *arbitrary* initial state, which is not necessarily in stationarity with respect to $g^*(\mathbf{x} \mid \gamma_{t-1})$. The HDR algorithm thus relies on a burn-in period to achieve approximate stationarity with respect to the target pdf. In other words, one obtains a sample approximately distributed according to the target $g^*(\mathbf{x} \mid \gamma_{t-1})$, and hence $\widehat{c}_t$ is not an unbiased estimate for c_t. Algorithm 3.1 avoids this problem by sampling from the minimum variance pdfs $\{g^*(\cdot \mid \gamma_t)\}_{t=0}^T$ sequentially in the order $g^*(\cdot \mid \gamma_0), g^*(\cdot \mid \gamma_1), \ldots, g^*(\cdot \mid \gamma_T)$, making it possible for the kernel density estimator (13) to reuse past samples that are already in stationarity with respect to the target. Note that in Ross's description [15] of the HDR method, sampling from the sequence of minimum variance pdfs $\{g^*(\cdot \mid \gamma_t)\}_{t=0}^T$ may be carried out in any order. For example, one could sample from $g^*(\cdot \mid \gamma_8)$ prior to sampling from $g^*(\cdot \mid \gamma_3)$.

Although the HDR algorithm does not generate samples in exact stationarity, it has the advantage that the resulting conditional estimates $\{\widehat{c}_t\}$ are completely independent of each other. Thus we emphasize that the idea of starting the Markov chains in Step 2 from arbitrary initial points (so that sampling from the target pdfs in a nested sequential order is not essential) has its own merit.

3.2 Approximate Relative Error Analysis

One can provide a rough approximation for the relative error of the estimator $\widehat{\ell}(\gamma)$ of $\ell(\gamma)$ prior to performing any computer simulation. Recall that $c_t = \mathbb{P}(S(\mathbf{X}) \geqslant \gamma_t \mid S(\mathbf{X}) \geqslant \gamma_{t-1}),\ t = 0, \ldots, T,\ \ \gamma_{-1} = -\infty$ are the conditional probabilities of reaching the next level, which are estimated via $\widehat{c}_t = \frac{1}{N}\sum_{i=1}^N I\{S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t\}$ in Algorithm 3.1. If the samples generated from each g_t^* were independent (this could, for example, be ensured to a close approximation by choosing a large burn-in parameter for the Markov kernel to reduce the dependence), we can write the variance of the estimator as follows.

$$\mathrm{Var}[\widehat{\ell}(\gamma)] = \prod_{t=0}^{T} \mathbb{E}_{g_t^*}[\widehat{c}_t^2] - \ell^2(\gamma) = \prod_{t=0}^{T} \left(\mathrm{Var}[\widehat{c}_t] + \widehat{c}_t^2\right) - \ell^2(\gamma).$$

Hence, if $R(\widehat{\ell})$ denotes the relative error of the estimator $\widehat{\ell}$, we can write:

$$R^2(\widehat{\ell}(\gamma)) = \prod_{t=0}^{T(\varrho,\gamma)-1} \left(1 + \frac{c_t^{-1} - 1}{N}\right)\left(1 + R^2(\widehat{\ell}_T)\right) - 1.$$

Observe that T—the number of terms in the sequence of levels—is an increasing function of γ and of ϱ. Also, $c_t \to \varrho$ as $N \to \infty$. Thus, for a fixed ϱ, we have $T \to \infty$ as $\gamma \to \infty$. Suppose we wish to have a fixed small RE for any value of γ. We are interested in how large N has to be and how it depends on γ. For example, if $N \approx 100\, T(\varrho;\gamma)/\varrho$ as $\gamma \to \infty$, then the relative error:

$$R^2(\widehat{\ell}) \approx \mathrm{e}^{(1-\varrho)/100} - 1, \quad \gamma \to \infty.$$

More specifically, if $\varrho = 0.1$, then $c_t \approx 0.1$ and $\lim_{\gamma\to\infty} R^2(\widehat{\ell}) \approx 0.0090$. Thus we find that, for a fixed ϱ, the relative error is approximately constant if

$$N = O(T(\gamma)).$$

Therefore, the complexity of the algorithm is determined from the dependence of $T(\gamma)$ on γ. If, for example, T grows linearly in γ, then the complexity of the algorithm is quadratic, because the total number of samples used in the simulation effort is $N \times (T+1)$. In general, for large N, $R^2(\widehat{\ell})$ as a function of γ is approximated by:

$$\begin{aligned} R^2(\widehat{\ell}(\gamma)) &\approx \prod_{t=0}^{\ln_\varrho^{\ell(\gamma)/\ell_T} - 1} \left(1 + \frac{c_t^{-1} - 1}{N}\right) \left(1 + R^2(\widehat{\ell}_T)\right) - 1 \\ &\approx \left(1 + \frac{\varrho_t^{-1} - 1}{N}\right)^{\ln_\varrho^{\ell(\gamma)/\ell_T}} \left(1 + R^2(\widehat{\ell}_T)\right) - 1. \end{aligned}$$

This formula can be used to make rough predictions about the performance of Algorithm 3.1 prior to the actual computer simulation.

4 Applications

In this section we present some applications of the proposed method.

4.1 Traveling Salesman Problem

The objective of the TSP is to find the shortest tour in a complete weighted graph containing n nodes. The problem can be formulated as minimizing the cost function

$$S(\mathbf{x}) = \sum_{i=1}^{n-1} C_{x_i,x_{i+1}} + C_{x_n,x_1},$$

where $\mathbf{x} = (x_1, \ldots, x_n)$ is a permutation of $(1, \ldots, n)$, x_i is the $i-$th node (or city) to be visited in a tour represented by $\mathbf{x}$, and C_{ij} is a cost (or distance associated with the edges of the graph) from node (or city) i to node j. We now discuss how we apply the proposed algorithm to find the optimal tour. The sequence of distributions from which we wish to sample is

$$g^*(\mathbf{x} \,|\, \gamma_t) \propto I\{S(\mathbf{x}) \leqslant \gamma_t\}, \quad \mathbf{x} \in \mathscr{X}, \quad t = 1, 2, \ldots,$$

where $\mathscr{X}$ is the set of all possible tours (i.e., the set of all possible permutations of $(1, \ldots, n)$), and $\{\gamma_t\}$ is an increasing sequence of levels. To apply Algorithm 3.2, we need only specify the Markov transition pdf in Step 2. The rest of the steps remain the same, taking Remark 3.6 into account. The Markov transition density κ for the TSP is specified via the following conditional sampling step. Given a tour $\mathbf{x}$, which is an outcome from $g^*(\mathbf{x}; \gamma_t)$, the conditional sampling consists of updating $\mathbf{x}$ to $\tilde{\mathbf{x}}$ with probability $I\{S(\tilde{\mathbf{x}}) \leqslant \gamma_t\}$, where $\tilde{\mathbf{x}}$ has the tour between the x_i-th and x_j-th cities $(j \neq i)$ reversed. For example, if $n = 10$ and $\mathbf{x} = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$ and $i = 4, j = 9$, then we accept the state $\tilde{\mathbf{x}} = (1, 2, 3, 9, 8, 7, 6, 5, 4, 10)$ with probability $I\{S(\tilde{\mathbf{x}}) \leqslant \gamma_t\}$; otherwise we do not update $\mathbf{x}$. The conditional sampling is therefore similar to the *2-opt* heuristic commonly employed in conjunction with simulated annealing [22]. Note, however, that the *2-opt* move here is not a mere heuristic, because the transition density κ has the desired stationary density.

For this and most other combinatorial optimization problems we apply the transition kernel with b number of burn-in times as described in Remark 3.3. In the course of the conditional sampling the score function is updated as follows $(j > i)$:

$$S(\tilde{\mathbf{x}}) = S(\mathbf{x}) - C_{x_{i-1},x_i} - C_{x_j,x_{j+1}} + C_{x_{i-1},x_j} + C_{x_i,x_{j+1}}.$$

We now present a number of numerical experiments which demonstrate the performance of the algorithm. Table 6 summarizes the results from a number of benchmark problems from the TSP library:

`http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/`

The experiments were executed using Matlab 7 on a laptop PC with a 2GHz Intel Centrino Duo CPU.

Table 6: Case studies for the symmetric TSP. The experiments were repeated ten times and the average minimum and maximum were recorded. The parameters of the proposed algorithm are $\varrho = 0.5$, $N = 10^2$ and $b = n \times 50$, where n is the size of the problem. The CPU times are in seconds.

`file`	γ^*	min	mean	max	CPU	$\bar{T}$
`burma14`	3323	3323	3323	3323	3.5	45.8
`ulysses16`	6859	6859	6859	6859	4.7	53.2
`ulysses22`	7013	7013	7013	7013	9.9	79.7
`bayg29`	1610	1610	1610	1610	16	113
`bays29`	2020	2020	2020	2020	16	110.7
`dantzig42`	699	699	699	699	38	188.2
`eil51`	426	426	427.6	430	57	249.1
`berlin52`	7542	7542	7542	7542	60	232.5
`st70`	675	675	675.8	680	116	370.6
`eil76`	538	538	543.9	547	145	428.6
`pr76`	108159	108159	108216	108304	130	372.3

Observe that the algorithm finds the optimal solution in all cases out of ten trials. In some cases, the algorithm found the optimal solution ten out of ten times. The number of iterations necessary to solve a problem increases with the size of the problem and is roughly equal to $\log_\varrho(n!)$, which is bounded from above by $n \log_\varrho(n)$. Table 7 gives some medium-scale examples. Again note the good performance of the algorithm in finding the optimal tour.

Table 7: Medium-scale case studies for the symmetric TSP. The experiments were repeated ten times and the average, minimum, and maximum were recorded. The parameters of the algorithm are $\varrho = 0.5$, $N = 10^2$ and $b = n \times 50$, where n is the size of the problem. The CPU times are in seconds.

`file`	γ^*	min	mean	max	CPU	$\bar{T}$
`a280`	2579	2581	2594.4	2633	5763	2026.7
`ch130`	6110	6110	6125.9	6172	1096	742.8
`eil101`	629	643	647.6	654	703	620.4
`gr120`	6942	6956	6969.2	6991	935	668.4
`gr137`	69853	69853	69911.8	70121	1235	781
`kroA100`	21282	21282	21311.2	21379	634	527.2
`kroB100`	22141	22141	22201	22330	634	526.7
`kroC100`	20749	20749	20774.4	20880	636	528.8
`kroD100`	21294	21294	21295.5	21309	630	529
`kroE100`	22068	22068	22123.1	22160	650	526.5
`lin105`	14379	14379	14385.6	14401	712	561.2
`pr107`	44303	44303	44305.3	44326	780	569.3
`pr124`	59030	59030	59048.4	59076	1018	691.5
`pr136`	96772	97102	97278.2	97477	1180	760
`pr144`	58537	58537	58640.9	59364	1406	827.6
`pr152`	73682	73682	73833	74035	1620	886.6
`rat99`	1211	1211	1213.2	1218	614	561.5
`rd100`	7910	7910	7910.8	7916	622	534.7
`si175`	21407	21013	21027.2	21051	2030	1066.3
`st70`	675	676	677.6	681	297	369.3
`swiss42`	1273	1273	1273	1273	103	180
`u159`	42080	42080	42383	42509	1688	934.4

4.2 Quadratic Assignment Problem

The quadratic assignment problem (QAP) is one of the most challenging problems in optimization theory. It has various applications, such as computer chip design, optimal resource allocation, and scheduling. In the context of optimal allocation, the objective is to find an assignment of a set of n facilities to a set of n locations such that the total cost of the assignment is minimized. The QAP can be formulated as the problem of minimizing the cost function

$$S(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} F_{ij}\, D_{x_i,x_j},$$

where $\mathbf{x} = (x_1, \ldots, x_n)$ is a permutation on $(1, \ldots, n)$, F is an $n \times n$ *flow* matrix, that is, F_{ij} represents the flow of materials from facility i to facility j and D is an $n \times n$ distance matrix, such that D_{ij} is the distance between location i and location j. We assume that both F and D are symmetric matrices. We now explain how we apply Algorithm 3.2 to the QAP. We again need only specify the Markov transition density κ in Step 2, since the rest of the steps are obvious. The transition density is specified by the following conditional sampling procedure.

1. Given the permutation $\mathbf{Y} = (Y_1, \ldots, Y_n)$, draw a pair of indices (I, J) such that $I \neq J$ and both I and J are uniformly distributed over the integers $1, \ldots, n$.

2. Given $(I, J) = (i, j)$, generate the pair $(\widetilde{Y}_i, \widetilde{Y}_j)$ from the conditional bivariate pdf
$$g^*(\widetilde{y}_i, \widetilde{y}_j \,|\, \widehat{\gamma}_{t-1}, \mathbf{Y}_{-i,-j}), \quad (\widetilde{y}_i, \widetilde{y}_j) \in \{(y_i, y_j), (y_j, y_i)\},$$
where $\mathbf{Y}_{-i,-j}$ is the same as $\mathbf{Y}$, except that the i-th and j-th elements are removed.

3. Set $Y_i = \widetilde{Y}_i$ and $Y_j = \widetilde{Y}_j$.

Sampling from $g^*(\widetilde{y}_i, \widetilde{y}_j \,|\, \widehat{\gamma}_{t-1}, \mathbf{Y}_{-i,-j})$ is accomplished as follows. Given the state $\mathbf{y}$, we update $\mathbf{y}$ to $\widetilde{\mathbf{y}}$ with probability $I\{S(\widetilde{\mathbf{y}}) \leqslant \gamma_{t-1}\}$, where $\widetilde{\mathbf{y}}$ is identical to $\mathbf{y}$ except that the i-th and j-th positions are exchanged. For example, if $\mathbf{y} = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$ and $i = 3, j = 7$, then $\widetilde{\mathbf{y}} = (1, 2, 7, 4, 5, 6, 3, 8, 9, 10)$. In the course of the conditional sampling, the score function is updated as follows:

$$S(\widetilde{\mathbf{y}}) = S(\mathbf{y}) + 2 \sum_{k \neq i,j} (f_{kj} - f_{ki})(D_{y_k,y_i} - D_{y_k,y_j}).$$

To achieve satisfactory performance we repeat the conditional sampling $b = n$ times. In other words, the the Markov kernel κ is applied with b burn-in steps. We now present a number of numerical experiments which demonstrate the performance of the algorithm. Table 8 summarizes the results from a number of benchmark problems from the TSP library:

`http://www.seas.upenn.edu/qaplib/inst.html`

Table 8: Case studies for the symmetric QAP. The experiments were repeated ten times and the average, minimum, and maximum were recorded. The parameters of the algorithm are $\varrho = 0.5$, $N = 10^3$ and $b = n$, where n is the size of the problem.

`file`	γ^*	min	mean	max	CPU	$\bar{T}$
`chr12a.dat`	9552	9552	9552	9552	21	45.2
`chr12b.dat`	9742	9742	9742	9742	21	45.4
`chr12c.dat`	11156	11156	11159	11186	20	42.5
`chr15a.dat`	9896	9896	9942.8	10070	36	53
`chr15b.dat`	7990	7990	8100	8210	36	53.4
`chr15c.dat`	9504	9504	10039	10954	36	53.4
`chr18a.dat`	11098	11098	11102.4	11142	60	64
`chr18b.dat`	1534	1534	1534	1534	54	57.3
`chr20a.dat`	2192	2192	2344	2406	75	66.9
`chr20b.dat`	2298	2352	2457.8	2496	70	64.5
`chr20c.dat`	14142	14142	14476.8	14812	85	77.7
`chr22a.dat`	6156	6156	6208.6	6298	105	81.4
`chr22b.dat`	6194	6194	6290.4	6362	97	75.3
`chr25a.dat`	3796	3796	4095.6	4286	147	90.1

The algorithm, although quite slower, works for large scale asymmetric QAP instances as well. For the instance `Lipa90b.dat` of size 90, we obtained the optimal solution tour ($S(\mathbf{x}^*) = 12490441$) ten out of ten times with algorithmic parameters $N = 10^2$, $\varrho = 0.5$, $b = 900$. The average iterations for convergence (using the same stopping criterion as above) was 505 and the average CPU time was 10426 seconds.

4.3 Tail Probabilities for Sums of Independent Random Variables

Consider the problem of estimating the tail probability

$$\ell(\gamma) = \mathbb{P}\left(\sum_{i=1}^{n} X_i \geqslant \gamma\right), \tag{20}$$

where γ is large enough to make $\ell(\gamma)$ a rare-event probability and $X_i \sim f(X_i),\ i = 1, \ldots, n$ independently. The sequence of minimum variance importance sampling pdfs for this problem is

$$g^*(\mathbf{x} \mid \gamma_t) \propto f(\mathbf{x}) I\{S(\mathbf{x}) \geqslant \gamma_t\}, \quad t = 1, 2, \ldots$$

where $f(\mathbf{x}) = \prod_{i=1}^n f(x_i)$ and $S(\mathbf{x}) = \sum_{i=1}^n x_i$. We now need only specify the Markov transition density κ in Step 2 of Algorithm 3.1. The rest of the steps remain the same. The transition density is specified via the following conditional sampling procedure.

Given $\mathbf{X}$ from the population $\widetilde{\mathcal{X}}^{(t-1)}$, update to the state $\widetilde{\mathbf{X}} = (\widetilde{X}_1, \ldots, \widetilde{X}_n)$ as follows.

1. Draw $\widetilde{X}_1$ from the conditional pdf

$$g^*(\widetilde{x}_1 \mid \gamma_{t-1}, X_2, \ldots, X_n) \propto f(\widetilde{x}_1)\, I\left\{\widetilde{X}_1 \geqslant \gamma_t - \sum_{i=2}^n X_i\right\}.$$

2. Draw $\widetilde{X}_i,\ i = 2, \ldots, n-1$ from

$$g^*(\widetilde{x}_i \mid \gamma_{t-1}, \widetilde{X}_1, \ldots, \widetilde{X}_{i-1}, X_{i+1}, \ldots, X_n) \propto f(\widetilde{x}_i)\, I\left\{\widetilde{X}_i \geqslant \gamma_t - \sum_{j \neq i}^n X_j\right\}.$$

3. Draw $\widetilde{X}_n$ from $g^*(\widetilde{x}_n \mid \gamma_{t-1}, \widetilde{X}_1, \ldots, \widetilde{X}_{n-1})$.

As a numerical example, consider the case where $n = 10$, $\gamma = 60$, and $f(x_i)$ is an exponential density with pdf e^{-x_i}, $x_i \geqslant 0$. We ran Algorithm 3.1 ten independent times with $N = 10^4$ and the sequence of levels depicted in Figure 3. From the ten trials we obtained the estimate $\widehat{\ell}(60) = 2.81 \times 10^{-16}$ with estimated RE of 3.4%. The sequence of levels was obtained from Algorithm 3.2 with $\varrho = 0.1$, $N_p = 10^4$, using the same kernel. Note that plotting the progress over the iterations, as in Figure 3, provides an empirical estimate of the rate function for the large-deviation probability considered here. For this problem the exact probability is known to be $\ell(\gamma) = \mathrm{e}^{-\gamma} \sum_{k=1}^{n-1} \frac{\gamma^k}{k!} \approx \mathrm{e}^{-\gamma}\gamma^{n-1}/(n-1)!$. Therefore $\ell \approx 2.8515 \times 10^{-16}$ and our estimate has an exact relative error of 3.4%. Figure 3 illustrates that the algorithm accurately samples from the minimum variance importance sampling pdf. The line of best fit $\log_{10}\widehat{\ell}(\gamma) = -0.33\gamma + 4.4$ indicates that the decay of $\log_{10}(\ell)$ as a function of γ is linear and hence the method has $O(\gamma^2)$ complexity. Figure 4 shows the empirical joint distribution of X_1 and $\sum_{j \neq 1} X_j$ at the penultimate step $(T-1)$. Observe that if $\sum_{j \neq 1} X_j$ is close or above $\gamma = 60$, then it is likely that X_2 will be small and essentially be simulated from the nominal pdf $f(x_2)$. This is what we would intuitively expect to happen if we sample from the minimum variance IS pdf $g^*(\cdot \mid \gamma_{T-1})$.

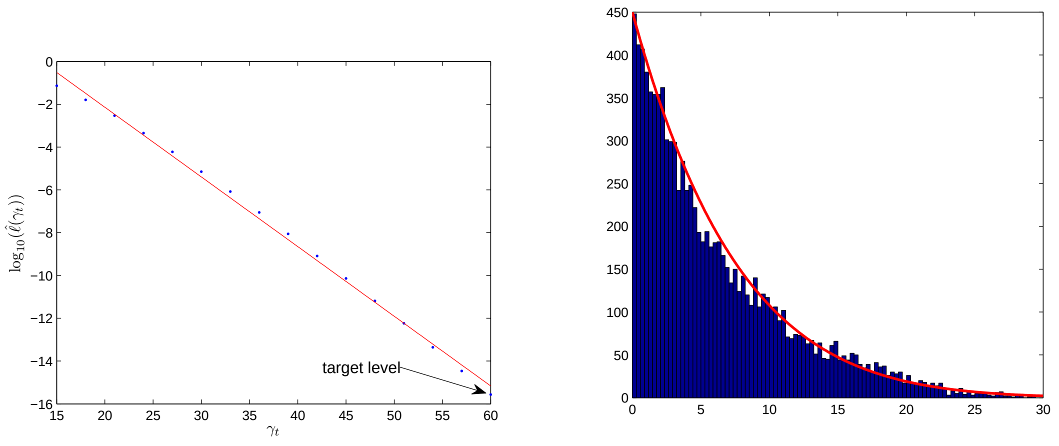

Figure 3: Left: typical evolution of the algorithm. Right: the empirical distribution of X_1 on the final iteration. The solid line depicts the exact analytical marginal pdf of X_1 (scaled). The histogram and the analytical pdf agree quite well, suggesting that the method indeed generates samples from $g^*(\cdot \mid \gamma)$.

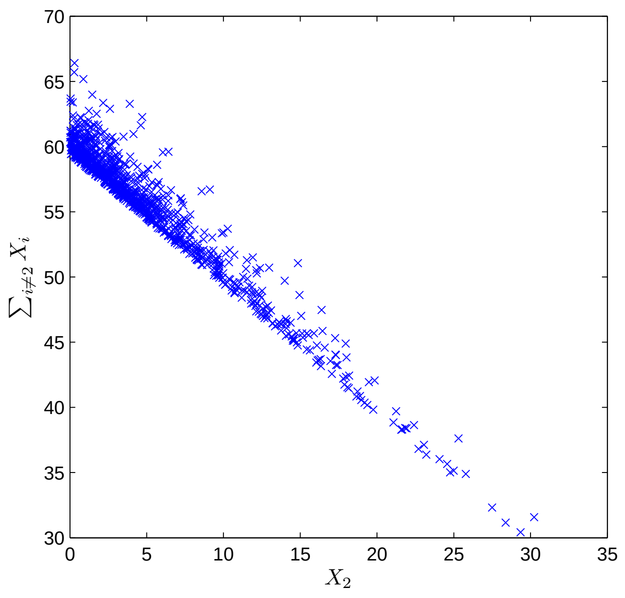

Figure 4: Empirical joint distribution of X_2 and $\sum_{j\neq 2} X_j$ at the penultimate step $(T-1)$.

4.4 Sum of Heavy-tailed Random Variables

Consider estimating the tail probability (20) in Section 4.3 when $f(x_i)$ is the $\mathsf{Weib}(0.2, 1)$ pdf and $n = 5$. Running Algorithm 3.1 independently ten times with $N = 10^4$, $\gamma = 10^6$, using Algorithm 3.2 with $N_p = 10^4$ and $\varrho = 0.1$ to determine the levels, we obtained $\widehat{\ell} = 6.578 \times 10^{-7}$ with an estimated relative error of 0.065. The total simulation effort was $10 \times N \times (T+2) = 8 \times 10^5$, including the estimation of the sequence of levels. A nice result is that the complexity of the algorithm can be sub-linear. Since $T = O(\ln(\ell(\gamma)))$ and $\ell(\gamma) = O(\mathrm{e}^{-\gamma^a})$ for large γ, for a fixed RE we have $N = O(\gamma^a)$. Hence the complexity is $O(\gamma^{2a})$, and it is sub-linear for $a < 0.5$, that is, the heavier the tail, the smaller the T needed to reach a fixed threshold γ and the smaller the sample size needed to achieve a fixed RE.

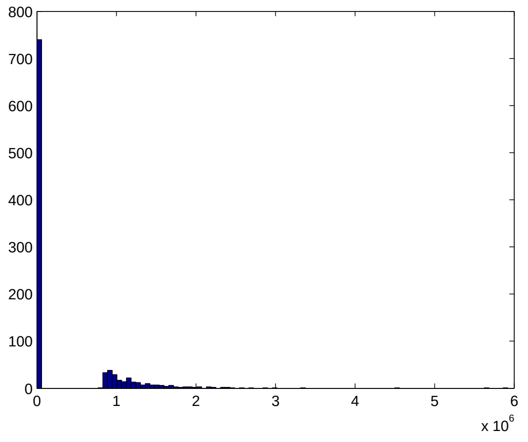

Figure 5: A histogram of the marginal distribution of X_1, where $\mathbf{X} \sim g^*(\mathbf{x} \mid 10^6)$ is simulated by Algorithm 3.1.

Figures 5 and 6 suggest that the algorithm is sampling accurately from the minimum variance IS pdf. Note that most of the observations on Figure 5 are simulated from the nominal pdf $f(x_1)$, with the rest simulated conditional on being close to the rare-event threshold $\gamma = 10^6$. The joint distribution depicted on Figure 6 suggests that when one of X_i is large, the rest are small. This means that the rare-event set is reached by having a single large variable in the sum, whilst all others are simulated under the nominal pdf. Again observe that an advantage of the proposed method is that the intermediate iterations that are needed to reach the desired threshold can be used to build an approximation to the rate function of the large-deviations probability. An example of such a construction is given in Figure 7, where we have applied Algorithm 3.1 for a problem with $\gamma = 10^{10}$ (again $n = 5$, $N = 10^4$, $a = 0.2$ and the sequence of levels was determined using Algorithm 3.2 with $\varrho = 0.1$ and $N_p = 10^4$). The estimate was $\widehat{\ell}(\gamma) = 1.95 \times 10^{-43}$. The line of best fit confirms the theoretical result that $\ell(\gamma) = O(\mathrm{e}^{-\gamma^a})$. In this case we can deduce from the empirical data that $\ell(\gamma) \to \mathrm{e}^{-\ln(10)(0.43\gamma^a - 0.65)} \approx 4.4668\ \mathrm{e}^{-0.99\gamma^a}$. This is close to the large-deviations analytical estimate of $n\mathrm{e}^{-\gamma^a}$, but note that whilst the large deviation estimate is valid asymptotically, the empirical estimate is a good approximation for finite values of γ.

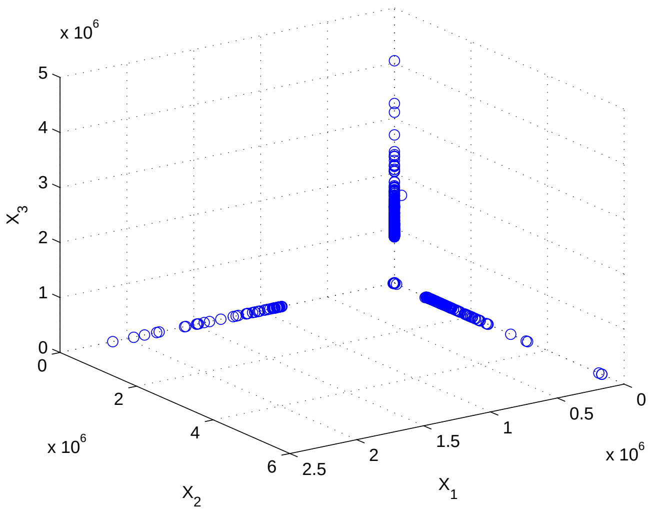

Figure 6: The empirical joint distribution of X_1, X_2, X_3 under $g^*(\mathbf{x} \mid 10^6)$.

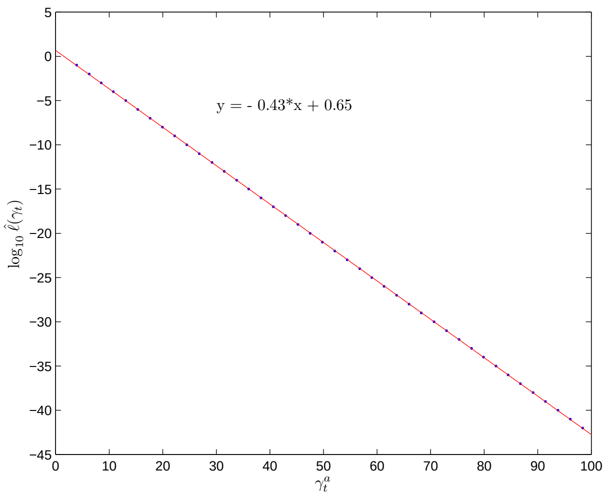

Figure 7: Results from Algorithm 3.1 on a problem with $\gamma = 10^{10}$, $n = 5$, $N = 10^4$, $a = 0.2$ and a sequence of levels determined using Algorithm 3.2.

4.5 Rare-Event Simulation over Disjoint Sets

A rare-event probability estimation problem of recent interest (see, e.g., [4]) is the estimation of probabilities of the form

$$\ell(\gamma) = \mathbb{P}\left(\left\{\sum_{i=1}^{n} X_i > \gamma\right\} \bigcup \left\{\sum_{i=1}^{n} X_i < -a\gamma\right\}\right), \quad a > 1, \tag{21}$$

where $X_i \sim f(x_i),\ i = 1, \ldots, n$ independently, and γ is large enough to make ℓ small. The minimum variance importance sampling pdf, from which the proposed algorithm samples, is

$$g^*(\mathbf{x} \,|\, \gamma) \propto I\left(\left\{\sum_{i=1}^{n} x_i > \gamma\right\} \bigcup \left\{\sum_{i=1}^{n} x_i < -a\gamma\right\}\right) \prod_{i=1}^{n} f(x_i).$$

The conditional pdfs are then a mixture of left and right truncated univariate pdfs ($i = 1, \ldots, n$):

$$g^*(x_i \,|\, \gamma, \mathbf{X}_{-i}) \propto f(x_i)\left(I\left\{x_i > \gamma - \sum_{j \neq i} X_j\right\} + I\left\{x_i < -a\gamma - \sum_{j \neq i} X_j\right\}\right). \tag{22}$$

We can easily sample from these conditionals using the inverse-transform method. Note that this problem cannot immediately be written in the form (1). This is why we present a detailed version of Algorithm 3.2. For definiteness we choose each $f(x_i)$ to be a Gaussian pdf with mean 0 and standard deviation 1 so that we can compute ℓ analytically and assess the quality of the estimate delivered by the proposed method.

Algorithm 4.1 (Rare-event simulation over disjoint sets)

1. *Set a counter* $t = 1$. *Given* N_p *and* ϱ, *initialize by generating*

$$\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_{N_p}^{(0)} \sim f(\mathbf{x}).$$

Here $f(\mathbf{x}) \propto e^{-\sum_{i=1}^{n} x_i^2/2}$. *Let* $\widehat{\gamma}_0$ *be the* $(1 - \varrho/2)$ *sample quantile of the population*

$$S(\mathbf{X}_1^{(0)}), \ldots, S(\mathbf{X}_{N_p}^{(0)}), \frac{-S(\mathbf{X}_1^{(0)})}{a}, \ldots, \frac{-S(\mathbf{X}_{N_p}^{(0)})}{a}.$$

In other words, $\widehat{\gamma}_0$ *is an estimate of the root of*

$$\mathbb{P}(\{S(\mathbf{X}) > \gamma\} \cup \{S(\mathbf{X}) < -a\gamma\}) = \varrho, \qquad \mathbf{X} \sim f(\mathbf{x}).$$

Let $\widetilde{\mathcal{X}}^{(0)} = \{\widetilde{\mathbf{X}}_1^{(0)}, \ldots, \widetilde{\mathbf{X}}_{N_0}^{(0)}\}$ *be the subset of the population* $\{\mathbf{X}_1^{(0)}, \ldots, \mathbf{X}_{N_p}^{(0)}\}$ *for which* $S(\mathbf{X}_i^{(0)}) \geqslant \widehat{\gamma}_0$ *or* $S(\mathbf{X}_i^{(0)}) \leqslant -a\widehat{\gamma}_0$.

2. *Sample uniformly with replacement N_p times from the population $\widetilde{\mathscr{X}}^{(t-1)}$ to obtain a new sample $\mathbf{Y}_1, \ldots, \mathbf{Y}_{N_p}$.*

3. *For each $\mathbf{Y} = (Y_1, \ldots, Y_n)$ in $\{\mathbf{Y}_1, \ldots, \mathbf{Y}_{N_p}\}$, generate $\widetilde{\mathbf{Y}} = (\widetilde{Y}_1, \ldots, \widetilde{Y}_n)$ as follows:*

   (a) Draw $\widetilde{Y}_1$ from the conditional pdf $g^*(\widetilde{y}_1 \,|\, \widehat{\gamma}_{t-1}, Y_2, \ldots, Y_n)$, where g^* is defined in (22).

   (b) Draw $\widetilde{Y}_i$ from $g^*(\widetilde{y}_i \,|\, \widehat{\gamma}_{t-1}, \widetilde{Y}_1, \ldots, \widetilde{Y}_{i-1}, Y_{i+1}, \ldots, Y_n), \quad i = 2, \ldots, n-1.$

   (c) Draw $\widetilde{Y}_n$ from $g^*(\widetilde{y}_n \,|\, \widehat{\gamma}_{t-1}, \widetilde{Y}_1, \ldots, \widetilde{Y}_{n-1})$.

   Denote the population of $\widetilde{\mathbf{Y}}$s thus obtained by $\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_{N_p}^{(t)}$.

4. *Set*
$$\widehat{\gamma}_t = \min\{\gamma, \widehat{d}\},$$
   where $\widehat{d}$ is the $(1 - \varrho/2)$ sample quantile of
$$S(\mathbf{X}_1^{(t)}), \ldots, S(\mathbf{X}_{N_p}^{(t)}), -\frac{S(\mathbf{X}_1^{(t)})}{a}, \ldots, -\frac{S(\mathbf{X}_{N_p}^{(t)})}{a}.$$
   Let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)}, \ldots, \widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the subset of the population $\{\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_{N_p}^{(t)}\}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \widehat{\gamma}_t$ or $S(\mathbf{X}_i^{(t)}) \leqslant -a\widehat{\gamma}_t$.

5. *If $\widehat{\gamma}_t = \gamma$, set $T = t$ and deliver the sequence of levels*
$$\widehat{\gamma}_0, \ldots, \widehat{\gamma}_{T-1}, \gamma,$$
   otherwise set $t = t + 1$ repeat from Step 2.

Algorithm 3.1 remains the same except that we let $\widetilde{\mathscr{X}}^{(t)} = \{\widetilde{\mathbf{X}}_1^{(t)}, \ldots, \widetilde{\mathbf{X}}_{N_t}^{(t)}\}$ be the subset of the population $\{\mathbf{X}_1^{(t)}, \ldots, \mathbf{X}_{N_p}^{(t)}\}$ for which $S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t$ or $S(\mathbf{X}_i^{(t)}) \leqslant -a\gamma_t$, and not just $S(\mathbf{X}_i^{(t)}) \geqslant \gamma_t$.

As a numerical example, consider the case where $n = 10$, $\gamma = 20$, and $a = 1.05$. First, we used Algorithm 4.1 with $N_p = 10^4$ and $\varrho = 0.1$ and obtained a sequence with 33 levels. Next, we applied Algorithm 3.1 with $N = 10^4$ ten independent times and obtained the estimate of $\widehat{\ell}(\gamma) = 1.41 \times 10^{-10}$ with an estimated RE of 5.5%. The total simulation effort is thus roughly equivalent to the generation of $33 \times N \times n \times 10 + N_p \times n = 3,31 \times 10^6$ random variables from univariate truncated Gaussian density.

Figure 8 shows the paths of a random walk constructed from the sample population at the final step of the algorithm. The empirical ratio of the number of paths that are above γ to the number of paths that end below $-a\gamma$ is 7.15.

This empirical estimate is in close agreement with the exact ratio of the tail probabilities:

$$\frac{\mathbb{P}_f(S_{10} > \gamma)}{\mathbb{P}_f(S_{10} < a\gamma)} \approx 8.14.$$

Thus the splitting of computational resources in the estimation of the probabilities over the two disjoint sets is close to optimal, that is, close to the minimum variance allocation of resources.

The Figure 9 shows the empirical joint distribution of X_1 and $\sum_{j\neq 1} X_j$ together with the contours of the exact joint pdf which is proportional to

$$\exp\left\{-\frac{x^2}{2} - \frac{y^2}{2(n-1)}\right\} \times I(\{x+y>\gamma\} \cup \{x+y<-a\gamma\}).$$

There is close agreement between the distribution of the points and the contours of the exact density, suggesting that we are sampling accurately from the zero variance importance sampling pdf.

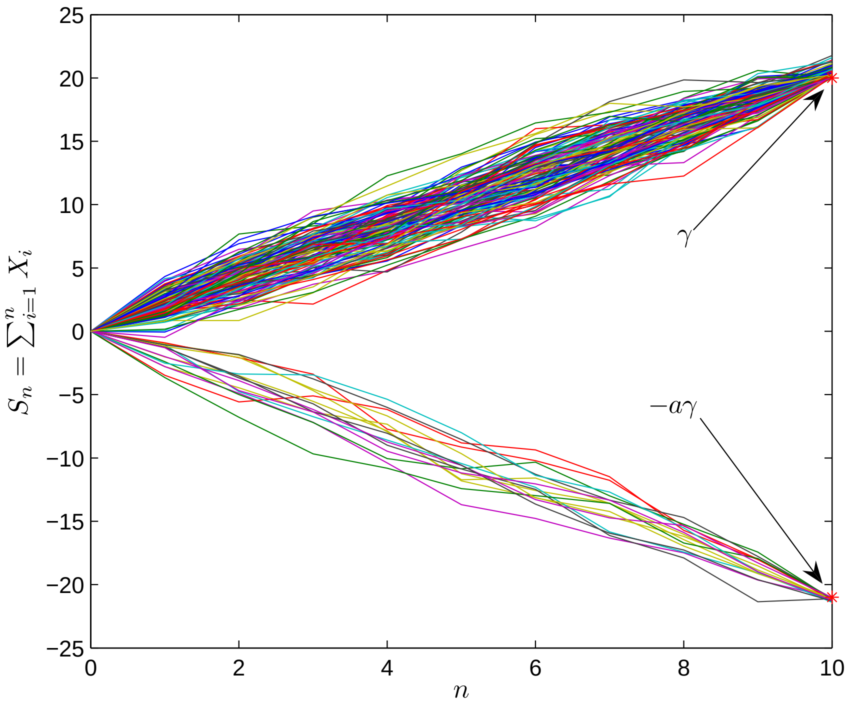

Figure 8: The paths of the random walk $S_n = \sum_{i=1}^n X_i,\ S_0 = 0,\ n = 0, \dots, 10,$ under the zero variance pdf $g^*(\mathbf{x} \,|\, \gamma)$.

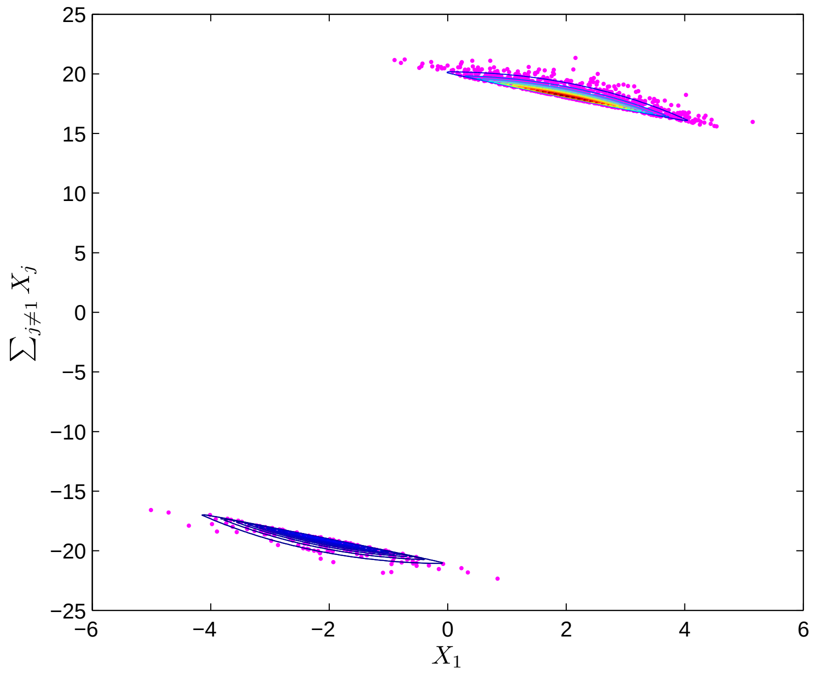

Figure 9: The empirical joint distribution of X_1 and $\sum_{j\neq 1} X_j$ together with the contours of the exact joint pdf.

5 Conclusions and Future Work

We have presented an algorithm for rare-event simulation and multi-extremal optimization. In contrast to standard importance sampling methods, the algorithm does not attempt to approximate the optimal importance sampling pdf (5) using a parametric model, but rather it aims to sample directly from the minimum variance pdf. Using a kernel density estimator with a suitable Markov transition kernel, we obtain samples from (5) in stationarity. Moreover, we obtain a consistent estimator of the normalizing constant of (5). The success of the algorithm depends on the convenient decomposition of the sample space into nested sets and the construction of a suitable Markov kernel κ. We have successfully applied the method to a variety of rare-event simulation, counting, and combinatorial optimization problems. In some cases, most notably for continuous high-dimensional rare-event probability estimation, we obtained superior numerical results compared to standard importance sampling methods.

An important advantage of the method is that the rare-event estimate does not suffer from the Likelihood Ratio degeneracy problems associated with standard importance sampling techniques. Thus, the performance of the algorithm does not deteriorate for rare-event simulation problems of large dimensions.

We obtained a representation of the rare-event probability as a product of conditional probabilities that are not too small, that is, we obtained a representation of the rare-event probability on a logarithmic scale. This representation has an important advantage when estimating very small probabilities, since the floating point accuracy of standard computer arithmetic and catastrophic cancellation errors limit the precision with which the LR terms in (4) can be evaluated.

The proposed algorithms require few parameters to be tuned, namely, the sample sizes N, N_p, the parameter b and and the rarity-parameter ϱ.

All proposed algorithms are highly parallelizable in a way similar to evolving a population of independent Markov chains.

The empirical performance of the proposed algorithm suggests that the proposed methodology has polynomial complexity. We will explore the issue of theoretical complexity in subsequent work. Finally, note that a disadvantage of the proposed methodology is that the use of the Markov kernel introduces dependence in the sample population, which makes an estimate of the RE from a single simulation run very difficult. This dependence can be eliminated with a large burn-in period, but this is precisely what we have been avoiding using the exact sampling method proposed in this paper. The dependence of the estimator $\widehat{\ell}$ on the parameter b is thus not yet clear and will be the subject of subsequent work.

As future work, we intend to exploit the exact sampling achieved with the method to generate random instances from the pdf of the Ising model for cases where standard MCMC samplers mix very slowly [2].

In addition, we will explore the benefits of estimating each conditional probability c_i independently. For example, we could obtain independent estimators of all the conditional probabilities $\{c_i\}$ in the following way. First, generate a population from $f(\mathbf{x})$ and use it to obtain an estimate $\widehat{\gamma}_0$ for γ_0, where $\mathbb{P}_f(S(\mathbf{X}) \geqslant \gamma_0 \,|\, S(\mathbf{X}) \geqslant \gamma_{-1}) = \varrho$. Next, simulate a new population from $f(\mathbf{x})$ and use it to obtain an estimate for $c_0 = \mathbb{P}_f(S(\mathbf{X}) \geqslant \widehat{\gamma}_0 \,|\, S(\mathbf{X}) \geqslant \gamma_{-1})$ and to help generate an exact sample from $g^*(\cdot \,|\, \widehat{\gamma}_0)$. Use the exact sample from $g^*(\cdot \,|\, \widehat{\gamma}_0)$ to obtain an estimate for γ_1, such that $\mathbb{P}_f(S(\mathbf{X}) \geqslant \gamma_1 \,|\, S(\mathbf{X}) \geqslant \widehat{\gamma}_0) = \varrho$. Next, given $\widehat{\gamma}_0$ and $\widehat{\gamma}_1$, simulate afresh from f and $g^*(\cdot \,|\, \widehat{\gamma}_0)$ to obtain an estimate for $c_1 = \mathbb{P}_f(S(\mathbf{X}) \geqslant \widehat{\gamma}_1 \,|\, S(\mathbf{X}) \geqslant \widehat{\gamma}_0)$ and generate an exact sample from $g^*(\cdot \,|\, \widehat{\gamma}_1)$. Use the sample from $g^*(\cdot \,|\, \widehat{\gamma}_1)$ to obtain an estimate for γ_2, such that $\mathbb{P}_f(S(\mathbf{X}) \geqslant \gamma_2 \,|\, S(\mathbf{X}) \geqslant \widehat{\gamma}_1) = \varrho$. Similarly, at step t, given $\{\widehat{\gamma}_i\}_{i=0}^{t-1}$, we simulate from all $\{g^*(\cdot \,|\, \widehat{\gamma}_i)\}_{i=0}^{t}$ afresh to obtain the estimates $\widehat{c}_{t-1}$ and $\widehat{\gamma}_t$. We terminate the procedure once we reach the desired level γ. The conditional probabilities $\{\widehat{c}_i\}$ are thus estimated independently and hence $\widehat{\ell} = \prod_{t=0}^{T} \widehat{c}_t$ will be unbiased for finite sample sizes. The procedure has the added advantage that it combines the estimation of the levels and the conditional probabilities within a single algorithm. The obvious disadvantage is the increased computational cost. As future work we will investigate if the possibility of estimating $\widehat{\ell}$ without bias justifies the extra computation cost.

We will also explore and compare the performance of the HDR algorithm on a range of estimation problems in order to determine how the dependence of the estimators $\{\widehat{c}_i\}$ affects the relative error of $\widehat{\ell}(\gamma_T)$.

In addition, preliminary results suggest that the method can be useful for noisy combinatorial optimization problems, such as the noisy TSP problem [22], and for continuous non-convex optimization problems with multiple extrema.

References

[1] Z. I. Botev. Nonparametric density estimation via diffusion mixing, `http://espace.library.uq.edu.au/view/UQ:120006`. *Technical Report, The University of Queensland*, 2007.

[2] Z. I. Botev. Three examples of a practical exact markov chain sampling, `http://espace.library.uq.edu.au/view/UQ:130865`. *Preprint, The University of Queensland*, 2007.

[3] Z. I. Botev, D. P. Kroese, and T. Taimre. Generalized cross-entropy methods for rare-event simulation and optimization. *Simulation*, 2008.

[4] J. A. Bucklew. *Introduction to Rare Event Simulation*. Springer-Verlag, New York, 2004.

[5] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986.

[6] P. Diaconis and S. Holmes. Three examples of Monte Carlo Markov chains: At the interface between statistical computing, computer science, and statistical mechanics. *Discrete Probability and Algorithms (Aldous et al, ed.)*, (43–56), 1995.

[7] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York, 2001.

[8] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, New York, 1994.

[9] G. Propp J and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 1 & 2:223–252, 1996.

[10] H. Kahn and T. E. Harris. *Estimation of Particle Transmission by Random Sampling*. National Bureau of Standards Applied Mathematics Series, 1951.

[11] D. P. Kroese, S. Porotsky, and R. Y. Rubinstein. The cross-entropy method for continuous multi-extremal optimization. *Methodology and Computing in Applied Probability*, 2006.

[12] D. Lieber, R. Y. Rubinstein, and D. Elmakis. Quick estimation of rare events in stochastic networks. *IEEE Transaction on Reliability*, 46:254–265, 1997.

[13] S. Olafsson. Two-stage nested partitions method for stochastic optimization. *Methodology and Computing in Applied Probability*, 6:5–28, 2004.

[14] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, New York, 2nd edition, 2004.

[15] S. M. Ross. *Simulation*. Academic Press, New York, 3rd edition, 2002.

[16] R. Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 2:127–190, 1999.

[17] R. Y. Rubinstein. The stochastic minimum cross-entropy method for combinatorial optimization and rare-event estimation. *Methodology and Computing in Applied Probability*, 7:5–50, 2005.

[18] R. Y. Rubinstein. How Many Needles are in a Haystack, or How to Solve #P-Complete Counting Problems Fast, journal = Methodology and Computing in Applied Probability. 8(1):5 – 51, 2006.

[19] R. Y. Rubinstein. How to deal with the curse of dimensionality of likelihood ratios in monte carlo simulation. 2007.

[20] R. Y. Rubinstein. Semi-iterative minimum cross-entropy algorithms for rare-events, counting, combinatorial and integer programming. *Methodology and Computing in Applied Probability*, 10:in print, 2008.

[21] R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method: A unified approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*. Springer Verlag, New York, 2004.

[22] R. Y. Rubinstein and D. P. Kroese. *Simulation and the Monte Carlo Method*. John Wiley & Sons, New York, second edition, 2007.

[23] M. Villén-Altamirano and J. Villén-Altamirano. RESTART: A method for accelerating rare event simulations. In J. W. Cohen and C. D. Pack, editors, *Proceedings of the 13th International Teletraffic Congress, Queueing, Performance and Control in ATM*, pages 71–76, 1991.