The expected bit complexity of the von Neumann rejection algorithm

Luc Devroye[*] and Claude Gravel[†]

March 10, 2016

Abstract

In 1952, von Neumann introduced the rejection method for random variate generation. We revisit this algorithm when we have a source of perfect bits at our disposal. In this random bit model, there are universal lower bounds for generating a random variate with a given density to within an accuracy ϵ derived by Knuth and Yao, and refined by the authors. In general, von Neumann's method fails in this model. We propose a modification that insures proper behavior for all Riemann-integrable densities on compact sets, and show that the expected number of random bits needed behaves optimally with respect to universal lower bounds. In particular, we introduce the notion of an oracle that evaluates the supremum and infimum of a function on any rectangle of $\mathbb{R}^d$, and develop a quadtree-style extension of the classical rejection method.

Keywords: random number generation, random bit model, von Neumann sampling algorithm, tree-based algorithms, random sampling, entropy

AMS subject classifications: 65C10 Random number generation, 68Q25 Analysis of algorithms and problem complexity, 68Q30 Algorithmic information theory, 68Q87 Probability in computer science (algorithm analysis, random structures, phase transitions, etc.), 68W20 Randomized algorithms, 68W40 Analysis of algorithms

1 Introduction

The purpose of this paper is to discuss von Neumann's [10] rejection method to generate a random variable X under the random bit model. In this model, we have access to

[*]School of Computer Science, McGill University, Canada

[†]Department of Computer Science and Operations Research, Université de Montréal, Canada

an infinite sequence of independent random Bernoulli($^1/_2$) bits, and are interested in the complexity as measured by the number of bits used until halting. For integer-value random variables X, the entire story is known. Knuth and Yao [9] obtained lower bounds on the expected number of bits for the exact simulation of X, and exhibited optimal algorithms. On the other hand, for random variables X with a density on $\mathbb{R}^d$, no exact algorithm exists, since any output of an algorithm delivers a function of a (possibly random) number of random bits. Thus, it is necessary to introduce the notion of an ϵ-approximation (see section 3) using Wasserstein L_∞-distance (Devroye and Gravel [4] and Rachev and Rüschendorf [11]).

A naive application of the rejection method—one of the most often used methods in simulation—leads to errors and inconsistencies. To deal with this, we introduce the notion of an oracle that computes the supremum and infimum of a function over any rectangles of $\mathbb{R}^d$ (see section 4). The oracle can be used in conjunction with a quadtree partition of the space to design a rejection algorithm that is guaranteed to deliver an ϵ-approximation (sections 4 and 5). We show that is valid whenever f is Riemann-integrable and derive expected complexity bounds (in terms of the number of random bits consumed) that are close to the universal lower bounds of [4].

We believe that random number generation libraries should offer the possibility of specifying any ϵ—no matter how small—as an input. Current practice entirely disregards the effects of approximations. In this paper and a companion paper [4], we take a first small step towards this goal. Applications in physics (see [8] who requires and develops ϵ-accurate normal generators), quantum computing (see Brassard, Devroye, and Gravel [6]), and other areas of science demand very precise computations. Much more is needed of course, especially when many random variables are combined in scientific computation—how does an ϵ-approximation propagate through a system, for example?

2 The discrete case

Consider two probability vectors $(p_i)_{i\in\mathbb{Z}}$, and $(q_i)_{i\in\mathbb{Z}}$, where

$$\sup_{i\in\mathbb{Z}} \frac{p_i}{q_i} \le C$$

for a finite constant C. Then von Neumann's rejection method can be reformulated as follows, if C is explicitly known:

The algorithm is at the beginning of the next page.

Algorithm 1 Von Neumann's method for discrete distributions in the random bit model

```
1: repeat
2:   Generate $X$ according to $(q_1, q_2, \ldots)$. {In the random bit model, the Knuth-Yao or
     Han-Hoshi algorithm can be used here to generate $X$.}
3:   Generate $U$ uniformly on $[0, 1]$.
4:   if $UCq_X \le p_X$ then
5:     return $X$ {Exit}
6:   end if
7: end repeat
```

The expected number of iterations in this algorithm is C. The returned random variable, X, has distribution $(p_i)_{i\in\mathbb{Z}}$. Knuth and Yao [9] showed that to generate X, the expected number of random bits required by any algorithm is at least the entropy of X,

$$\mathcal{E}(X) \overset{\text{def}}{=} \sum_{i\in\mathbb{Z}} q_i \log_2\left(\frac{1}{q_i}\right).$$

They also exhibited an algorithm that requires an expected number of bits not exceeding $\mathcal{E}(X) + 2$. Note that given X, the decision

$$UCq_X \le p_X$$

can be made using 2 expected random bits because to decide $U \le p_X/Cq_X$ given X follows a geometric law with parameter $1/2$; we compare the i^{th} bit of U with the i^{th} bit of p_X/Cq_X until both don't agree for integers $i > 0$ (see, e.g., Devroye and Gravel [4]). The expected number of random bits needed by this implementation is not more than

$$C\big(\mathcal{E}(X) + 2\big).$$

This is usually quite far from the lower bound of Knuth and Yao, $\sum_{i\in\mathbb{Z}} p_i \log_2 \frac{1}{p_i}$. It is worth to mention an application of the rejection method in the bit model to the simulation of physical phenomena and to communication complexity in Brassard, Devroye, and Gravel [6]. The remainder of the paper is concerned with the case in which X has a density f. Since such X cannot be generated exactly by any algorithm, it has to be approximated in some manner by a discrete random variable, and thus we require an appropriate—and as it turns out, nontrivial—generalization of Algorithm 1.

3 Bisection algorithm

The building block for continuous random generation is the bisection algorithm, which is mathematically equivalent to an algorithm given in Devroye and Gravel [4]. We develop a version here that is convenient for later use. The analysis of Theorem 1 below is new. Consider an algorithm for generating a random variable X with density f on $\mathbb{R}^d$ that has the following property: for a given $\epsilon > 0$, it outputs a random variable $X_\epsilon \in \mathbb{R}^d$—an ϵ-approximation of X—such that there exists a coupling of (X, X_ϵ) with

$$\operatorname{ess\,sup} \|X - X_\epsilon\| \leq \epsilon,$$

where $\|\cdot\|$ is the L_∞-distance in $\mathbb{R}^d$. It is understood that necessarily, X_ϵ is discret since it is a function of a (random) finite number of random bits. We call X_ϵ an ϵ-approximation of X.

Devroye and Gravel [4] showed that if T_ϵ is the random number of bits needed by any algorithm for generating such an approximation X_ϵ, then

$$\mathbf{E}(T_\epsilon) \geq \mathcal{E}(f) + d \log_2 \left(\frac{1}{\epsilon} \right) - d, \tag{1}$$

where $\mathcal{E}(f)$ is the differential entropy of f,

$$\mathcal{E}(f) = \int f \log_2 \left(\frac{1}{f} \right).$$

The lower bound is valid under a technical condition known as Rényi's condition (see Rényi [12] and Csiszàr [2]), namely that the entropy of the discrete random variable $\lfloor X \rfloor$, which takes values on the grid of all integer-valued vectors of $\mathbb{R}^d$, be finite. The lower bound (1) serves as a guide to calibrate and compare the rejection algorithms presented in this paper. It is especially crucial to match its main term, $d \log_2 \left(\frac{1}{\epsilon} \right)$, without an extra multiplicative constant.

We require an auxiliary set of results on bisection algorithms for generating a random variate that is an ϵ-approximation of a random variable X with a continuous (not necessarily absolutely continuous) distribution function G on a compact interval $[a, b]$ of length $L \stackrel{\text{def}}{=} |b - a|$. The bisection Algorithm 2 assumes that we have access to both G and G^{-1}.

The algorithm is at the beginning of the next page.

Algorithm 2 The bisection algorithm in the random bit model (Devroye and Gravel [4])

```
1:  $J \leftarrow [G(a), G(b)] = [0, 1]$
2:  repeat
3:    if $|I| \overset{\text{def}}{=} b - a \leq 2\epsilon$ then
4:      $X_\epsilon \leftarrow \frac{a+b}{2}$
5:      return $X_\epsilon$ {Exit}
6:    else
7:      $B \leftarrow$ random unbiased bit.
8:      $z \leftarrow G^{-1}\left(\frac{G(a)+G(b)}{2}\right)$.
9:      if $B = 0$ then
10:       $I \leftarrow [a, z]$
11:       $J \leftarrow \left[G(a), \frac{G(a)+G(b)}{2}\right] = \left[G(a), G(z)\right]$
12:     else
13:       $I \leftarrow [z, b]$
14:       $J \leftarrow \left[\frac{G(a)+G(b)}{2}, G(b)\right] = \left[G(z), G(b)\right]$
15:     end if
16:   end if
17: end repeat
```

Theorem 1. *For the bisection Algorithm 2 applied to any distribution with support on an interval of length L, and halted as soon as an interval of length less than or equal to 2ϵ is reached, we have*

(i) If X_ϵ denotes the center of the halting interval, then there exists a coupling between X and X_ϵ such that $\|X - X_\epsilon\| \leq \epsilon$.

(ii) If T_ϵ denotes the number of bits used, then

$$\mathbf{E}(T_\epsilon) \leq 3 + \log_2^+\left(\frac{L}{2\epsilon}\right),$$

where $\log_2^+(x) = \max\{0, \log_2(x)\}$.

Remark 1. We note here that in general this bound cannot be improved by more than 3 bits. Just consider the uniform distribution on $[0, L]$. Since all intervals have length to $\frac{L}{2^i}$ after i were used, we have

$$T_\epsilon = \min\left\{i \geq 0 : \frac{L}{2^i} \leq 2\epsilon\right\} = \max\left\{0, \left\lceil \log_2 \frac{L}{2\epsilon} \right\rceil\right\}.$$

Proof of Theorem 1. The bisection method yields, very naturally, a full binary tree, i.e., one in which all internal nodes have two children. Each internal node corresponds to a subinterval of $[a,b]$ of length greater than 2ϵ, the root represents the original interval $[a,b]$ of length L, and leaves represent intervals of length less than or equal to 2ϵ that cause an exit.

Upon exit, the random variable X_ϵ can be coupled with X such that $\|X - X_\epsilon\| \leq \epsilon$, because at every iteration, the random binary choice picks the correct interval, $[a,z]$ or $[z,b]$, with the correct probability $1/2$. One could thus define X as the limit of I when the algorithm is run without halting. Since X_ϵ is the midpoint of an interval of length at most 2ϵ that also contains X, we must have $\|X - X_\epsilon\| \leq \epsilon$. This shows part (i).

To prove part (ii), let us denote the set of leaves by $\mathcal{L}$, and the set of internal nodes (i.e., all non-leaf nodes) by $\mathcal{I}$. The depth of node u is denoted by $\mathrm{d}(u)$. It is of course possible that $\mathcal{I}$ and $\mathcal{L}$ are both infinite. However, one has that in all cases,

$$\sum_{u\in\mathcal{L}} \frac{1}{2^{\mathrm{d}(u)}} \leq 1,$$

because the leaves form a non-overlapping covering of $[a,b]$ so that the bisection method—a random walk started at the root and halted when a leaf is reached—always stops. Also,

$$\mathbf{E}(T_\epsilon) = \sum_{u\in\mathcal{L}} \frac{\mathrm{d}(u)}{2^{\mathrm{d}(u)}}.$$

In the next chain of inequalities, we define

$$N_\ell = \sum_{v\in\mathcal{I}} \mathbb{1}_{\{\mathrm{d}(v)=\ell\}}, \;\; \ell \geq 0,$$

i.e., the number of internal nodes at depth ℓ in the tree. Using Kraft's inequality (see, e.g., Cover and Thomas [1]), which states that for any binary tree,

$$\sum_{u\in\mathcal{L}} \frac{1}{2^{\mathrm{d}(u)}} \leq 1,$$

we have, with $A(u)$ denoting the set of ancestors u and $D(v)$ denoting the set of descendants of v, that, since $d(u) = \sum \mathbb{1}\{v \in A(u) \setminus \{u\}\}$,

$$\begin{aligned}
\mathbf{E}(T_\epsilon) &= \sum_{u\in\mathcal{L}} \sum_{\substack{v\in A(u)\\ v\neq u}} \frac{1}{2^{\mathrm{d}(v)}} \frac{1}{2^{\mathrm{d}(u)-\mathrm{d}(v)}} \\
&= \sum_{v\in\mathcal{I}} \frac{1}{2^{\mathrm{d}(v)}} \sum_{\substack{u\in D(v)\\ u\in\mathcal{L}}} \frac{1}{2^{\mathrm{d}(u)-\mathrm{d}(v)}}
\end{aligned}$$

$$\le \sum_{v\in\mathcal{I}} \frac{1}{2^{\mathrm{d}(v)}} \quad \text{(by Kraft's inequality)}$$
$$= \sum_{\ell=0}^{\infty} \frac{N_\ell}{2^\ell}.$$

Observe that at depth ℓ, all intervals associated with nodes are disjoint, and thus, since each interval node corresponds to an interval strictly larger than 2ϵ,

$$N_\ell < \frac{L}{2\epsilon}.$$

Also, $N_\ell \le 2^\ell$ because we have a binary tree. Hence,

$$N_\ell \le \min\left\{\lfloor {}^{L}/_{2\epsilon}\rfloor, 2^\ell\right\}.$$

We deduce, using

$$\ell_0 = \max\left\{0, \left\lceil \log_2 \frac{L}{2\epsilon}\right\rceil\right\},$$

that

$$\begin{aligned}\sum_{\ell=0}^{\infty} \frac{N_\ell}{2^\ell} &\le \sum_{\ell=0}^{\ell_0} 1 + \sum_{\ell=\ell_0+1}^{\infty} \left\lfloor \frac{L}{2\epsilon}\right\rfloor \frac{1}{2^\ell} \\ &= \ell_0 + 1 + \left\lfloor \frac{L}{2\epsilon}\right\rfloor \frac{1}{2^{\ell_0}} \\ &\le \log_2^+\left(\frac{L}{2\epsilon}\right) + 3.\end{aligned}$$

To see this, treat the cases $L < 2\epsilon$ and $L \ge 2\epsilon$ separately. □

4 A rejection algorithm for densities with compact support

In this section, we assume that f is Riemann-integrable and supported on $[0,1]^d$. Note that this is equivalent to the assumption that f is almost-everywhere continuous, bounded, and supported on $[0,1]^d$. Our algorithm requires an oracle—a black box—that for any closed rectangle $R \subseteq \mathbb{R}^d$ gives

$$\sup_{x\in R} f(x) \quad \text{and} \quad \inf_{x\in R} f(x).$$

If $R = \{x\}$, then that oracle coincides with a standard function evaluation. Without the possibility of computing infimum and supremum of the density f over compact subintervals of the domain of f, sampling absolutely continuous distribution using the rejection method seems to be impossible in total generality. We will also track complexity in terms of the number of uses of the oracle before halting, and call it T_ϵ. One use of the oracle reveals

$$C \overset{\text{def}}{=} \sup_{x\in[0,1]^d} f(x),$$

which is a finite number by assumption (Riemann-integrable functions are bounded by definition). At once, we have a simple bound for applying the rejection method: $f(x) \le C$. Algorithm 3 shows the original algorithm by von Neumann [10] (see also Devroye [3]) of the standard rejection algorithm.

Algorithm 3 Von Neumann's original rejection algorithm

```
1: repeat
2:   Generate $X$ uniformly on $[0,1]^d$.
3:   Generate $U$ uniformly on $[0,1]$.
4:   if $UC \le f(X)$ then
5:     return $X$
6:   end if
7: end repeat
```

Since we cannot generate X and U with infinite precision, at least two modifications are needed. One modification is to take into account the precision ϵ desired for X, and another modification is to take into account that bits of U are generated sequentially. Appendix B, which is in this article for pedagogical purpose, gives more insights on wrong and naive modifications that someone could be tempted to do. We can consider the initial rectangle

$$R_0 = [0,1]^d \times [0,C],$$

and its 2^{d+1} child rectangles defined by the 2^{d+1} quadrants centered at the center x_0 of R_0:

$$x_0 = \Big(\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2}, \frac{C}{2}\Big).$$

In the data structure literature, such a partition, when applied recursively, leads to a quadtree (see, e.g., Samet [13]). Any subsequent rectangle can be split again about its center point, and so forth, in the manner of infinite quadtree Q on $R_0 \subseteq \mathbb{R}^d$ as illustrated by Figures 1 and 2.

The figures are at the beginning of the next page.

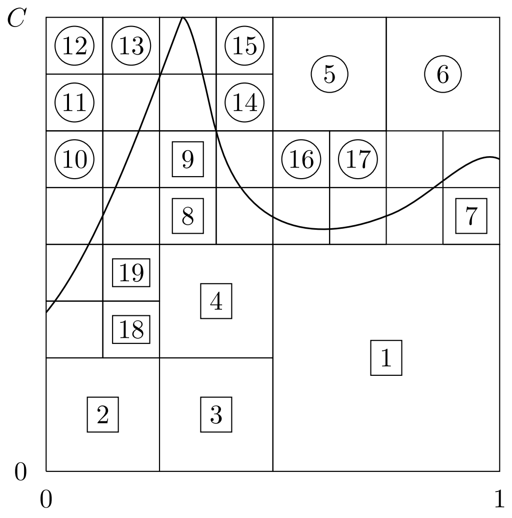

Figure 1: Decomposition of $[0, 1] \times [0, C]$ into its quadtree tree structure.

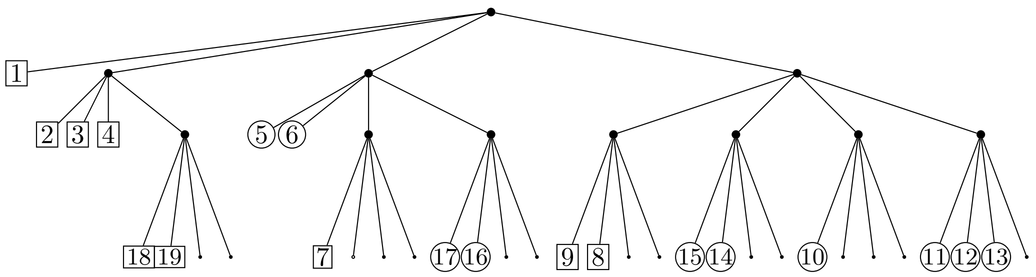

Figure 2: Quadtree for decomposition on Figure 1.

In von Neumann's algorithm, deciding if $UC \leq f(X)$ for $(X, U) \in R_0$ is equivalent to finding a rectangle R in the quadtree Q with the property that either

$$R \subseteq \big\{(x, y) \in R_0 : \ \ y \leq f(x)\big\} \quad \text{(we accept since } UC \leq f(X))$$

or

$$R \subseteq \big\{(x, y) \in R_0 : \ \ y > f(x)\big\} \quad \text{(we reject since } UC > f(X)).$$

However, this must be done carefully, without overlapping rectangles. Thus, we must trim Q, and associate the exiting rectangles R with the leaves. Thus,

$$\big\{(x,y)\in R_0:\ \ y\le f(x)\big\}=\bigcup\{R:\ \ R \text{ is an accepting rectangle}\}, \tag{2}$$

and

$$\big\{(x,y)\in R_0:\ \ y> f(x)\big\}=\bigcup\{R:\ \ R \text{ is a rejecting rectangle}\}. \tag{3}$$

Below, we will see that Riemann-integrability of f suffices for this decomposition. When a rejecting rectangle is found, the procedure is repeated. When an accepting rectangle is found, say,

$$\prod_{i=1}^{d}[a_i,b_i]\times[\alpha,\beta],$$

it suffices to generate $X_\epsilon\in\prod_{i=1}^{d}[a_i,b_i]$ such that

$$\|X^\star - X_\epsilon\|\le\epsilon \tag{4}$$

where $X^\star$ is uniform on $\prod_{i=1}^{d}[a_i,b_i]$ and coupled with X_ϵ. It is easy to see by the triangle inequality that X_ϵ is then coupled with X having density f such that

$$\|X-X_\epsilon\|\le\epsilon.$$

To achieve (4), use bisection for each dimension separately. By Theorem 1, the expected number of bits needed is bounded by

$$\begin{aligned}3+\sum_{i=1}^{d}\log_2^+\left(\frac{b_i-a_i}{\epsilon}\right)&=3+\sum_{\substack{i=1\\ b_i-a_i>\epsilon}}^{d}\log_2\left(\frac{b_i-a_i}{\epsilon}\right)\\&\le 3+d\log_2\left(\frac{1}{\epsilon}\right)\end{aligned}$$

for $\epsilon\le 1$. If $\epsilon>1$, then bisection requires no bit, so that we conclude that the expected number of bits does not exceed

$$3+d\log_2^+\left(\frac{1}{\epsilon}\right).$$

We note that the checks

$$R\subseteq\{(x,y)\in\mathbb{R}^d\times\mathbb{R}:\ \ f(x)\le y\} \tag{5}$$

and

$$R\subseteq\{(x,y)\in\mathbb{R}^d\times\mathbb{R}:\ \ f(x)> y\} \tag{6}$$

can be carried out using our oracle. Let

$$\begin{aligned}
f^+ &= \sup\big\{f(x) : \ (x,y) \in R \text{ for some } y\big\}, \\
f^- &= \inf\big\{f(x) : \ (x,y) \in R \text{ for some } y\big\}, \\
y^- &= \inf\{y : \ (x,y) \in R \text{ for some } x\big\}, \\
y^+ &= \sup\{y : \ (x,y) \in R \text{ for some } x\big\}.
\end{aligned}$$

Then (5) holds if $f^+ \leq y^-$, and (6) holds if $f^- \geq y^+$.

Algorithm 4 Generation of an ϵ-approximate random variable with density on $[0,1]^d$

```
1:  $R \leftarrow [0,1]^d \times \big[0, \sup_{x\in[0,1]^d} f(x)\big]$
2:  Decision $\leftarrow$ None
3:  repeat
4:      Call the oracle that returns $\inf_{x\in R^\star} f(x)$ and $\sup_{x\in R^\star} f(x)$ which are in turn used
        by the following branching statement. {Here and below $R^\star$ denotes the projection
        of $R$ onto $\mathbb{R}^d$, i.e., $R^\star = \{x : \ (x,y) \in R \text{ for some } y\}$.}
5:      if $R \subseteq \{(x,y) \in \mathbb{R}^d \times \mathbb{R} : \ f(x) \leq y\}$ then
6:          Decision $\leftarrow$ Accept
7:      else if $R \subseteq \{(x,y) \in \mathbb{R}^d \times \mathbb{R} : \ f(x) > y\}$ then
8:          Decision $\leftarrow$ Reject
9:      else
10:         $x^\star \leftarrow$ center of $R$
11:         Select one vertex $v$ of $R$ uniformly at random and replace $R$ by the rectangle
            with $v$ and $x^\star$ as opposing vertices.
12:     end if
13: until Decision $\neq$ None
14: if Decision $=$ Reject then
15:     Goto line (1) {Restart the algorithm an average of $\sup_{x\in[0,1]^d} f(x)$ times.}
16: else
17:     Use bisection to generate an $\epsilon$-approximation $X_\epsilon$ of a uniform variable in $R^\star$.
18:     return $X_\epsilon$
19: end if
```

Theorem 2. *Let f be a Riemann-integrable density on $[0,1]^d$. Then the quadtree Q partitions $R_0 \overset{def}{=} [0,1]^d \times [0, \sup f]$ in a collection of leaf rectangles R for which (2) and (3) hold. Thus, Algorithm 4 halts with probability one and delivers an ϵ-approxiation X_ϵ of X, a random variable with density f.*

Proof of Theorem 2. Let T be the number of iterations of the algorithm before halting. In other words, T is the depth of the leaf rectangle R reached by randomly walking down the quadtree. That walk costs $T(d+1)$ random bits. We show that

$$\lim_{k\to\infty} \mathbf{P}\{T > k\} = 0,$$

and thus, (2) and (3) must hold. In the partition of R_0 into $2^{(d+1)k}$ level-k rectangles (each of Lebesgue measure $\frac{1}{2^k}\frac{1}{2^k}\cdots\frac{1}{2^k}\frac{C}{2^k}$), there are N_k cells R—those for which we cannot decide—that are "visited" by f, i.e. for which

$$\sup_{(x,y)\in R} f(x) \geq \inf\{y : \ (x,y) \in R \text{ for some } x\}$$

and

$$\inf_{(x,y)\in R} f(x) \leq \sup\{y : \ (x,y) \in R \text{ for some } x\}.$$

Then

$$\mathbf{P}\{T > k\} = \frac{N_k}{2^{(d+1)k}}.$$

For every rectangle R, let us define its projection, $R^\star$, onto $\mathbb{R}^d$, i.e.

$$R^\star = \{x : \ (x,y) \in R \text{ for some } y\}.$$

Then, for a fixed dimension, we can group the 2^k cells $R^\star$ with the same projection and verify that of these 2^k cells, at most

$$\left(\frac{\sup_{x\in R^\star} f(x) - \inf_{x\in R^\star} f(x)}{C}\right)2^k + 2$$

are visited by f. Since there are 2^{dk} rectangles $R^\star$, let $\mathcal{P}_k^\star$ be the collection of all projections $R^\star$, and then

$$\begin{aligned} N_k &\leq \sum_{R^\star\in\mathcal{P}_k^\star} \left(\left(\frac{\sup_{x\in R^\star} f(x) - \inf_{x\in R^\star} f(x)}{C}\right)2^k + 2\right) \\ &= \frac{(I^+ - I^-)}{C} 2^{(d+1)k} + 2\cdot 2^{dk}, \end{aligned}$$

where we use the Riemann approximations I^+ and I^- of the integral of f:

$$\begin{aligned} I_k^+ &= \sum_{R^\star\in\mathcal{P}_k^\star} \Big(\sup_{x\in R^\star} f(x)\Big)\lambda(R^\star), \\ I_k^- &= \sum_{R^\star\in\mathcal{P}_k^\star} \Big(\inf_{x\in R^\star} f(x)\Big)\lambda(R^\star), \end{aligned}$$

$$\lambda(R^\star) = \text{Lebesgue measure of } R^\star = \frac{1}{2^{dk}}.$$

Therefore,

$$\mathbf{P}\{T > k\} \le \frac{2}{2^k} + \frac{I_k^+ - I_k^-}{C}.$$

Since f is Riemann-integrable, this tends to 0 as $k \to \infty$.

□

We call f a monotone density on $[0,1]^d$ if it decreases along at least one of the dimensions, i.e., there exists $i \in \{1, \ldots, d\}$ such that for all vectors $(x_1, \ldots, x_i, \ldots, x_d) \in [0,1]^d$, $(x_1, \ldots, x_i', \ldots, x_d) \in [0,1]^d$, $x_i \le x_i'$, then $f(x_1, \ldots, x_i, \ldots, x_d) \ge f(x_1, \ldots, x_i', \ldots, x_d)$. As before, let N_k be the number of cells at level k that are visited by f. Then,

$$N_k \le 2 \cdot 2^k \cdot 2^{(d-1)k}$$

because the domain of f is divided into 2^{dk} cells and the 2^k cells along the i^{th} dimension give rise to a walk. This walk along the i^{th} is at most of length $2 \cdot 2^k$ as illustrated on Figure 3.

We have

$$\mathbf{P}\{T > k\} = \frac{N_k}{2^{(d+1)k}} \le \frac{2}{2^k}, \ \ k \ge 0,$$

and thus,

$$\mathbf{E}(T) = \sum_{k=0}^{\infty} \mathbf{P}\{T > k\} \le 4.$$

In other words, for monotone densities, the inner loop of the algorithm has a uniform performance guarantee.

For a complete analysis of algorithm A, we need to consider the total number N of trials before deciding. The number of uses of the oracle is given by

$$\sum_{i=1}^{N} T_i,$$

where T_i is the number of iterations in the i-th trial—these T_i's are i.i.d. The number of random bits used is

$$(d+1)\sum_{i=1}^{N} T_i$$

during the first phase (the decision phase), and is bounded by

$$3 + d \log_2^+ \left(\frac{1}{2\epsilon}\right)$$

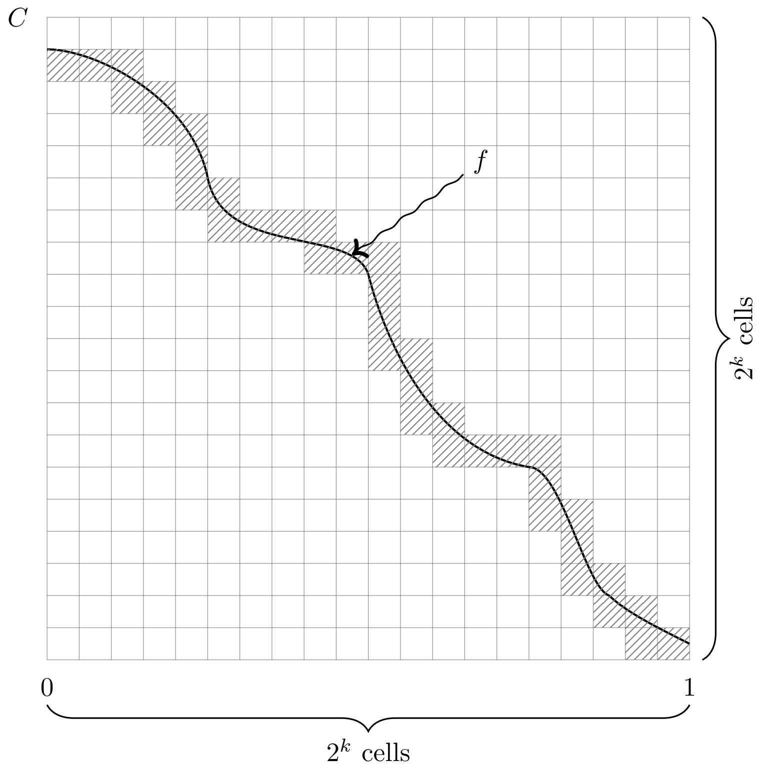

Figure 3: The number of cells visited by the monotone curve f is at most $2 \cdot 2^k$ cells.

in the second phase (the bisection phase). Since $\mathbf{E}(N) = C = \sup_{x\in[0,1]^d} f(x)$, we see that the expected number of uses of the oracle is

$$C\mathbf{E}(T)$$

and that the expected number of random bits required is bounded from above by

$$(d+1)C\mathbf{E}(T) + 3 + d\log_2^+\left(\frac{1}{2\epsilon}\right).$$

As noted earlier, for any coordinate-wise monotone density on $[0,1]^d$,

$$\mathbf{E}(T) \leq 4.$$

However, for general Riemann-integrable densities we cannot insure that $\mathbf{E}(T)$ converges. We will address that point below.

Theorem 3. *Let f be a Riemann-integrable density on $[0,1]^d$, with $C \overset{def}{=} \sup_{x\in[0,1]^d} f(x)$.*

1. *If f is monotone in at least one coordinate, then the expected number of uses of the oracle is not more than*
$$4C,$$
and the expected number of bits needed to generate an ϵ-approximation X_ϵ is not more than
$$4C(d+1) + 3 + d\log_2^+\left(\frac{1}{2\epsilon}\right).$$

2. *If I_k^+ and I_k^- are the Riemann approximation of $\int f$ for regular grids of size 2^{dk} i.e., each coordinate is split into 2^k equal intervals, then the expected number of uses of the oracle is not more than*
$$4C + \mathcal{A}(f),$$
where
$$\mathcal{A}(f) \overset{def}{=} \sum_{k=0}^{\infty} \left(I_k^+ - I_k^-\right),$$
and the expected number of random bits used by Algorithm 4 is not more than
$$4C(d+1) + (d+1)\mathcal{A}(f) + 3 + d\log_2^+\left(\frac{1}{2\epsilon}\right).$$

Proof of Theorem 3. Just recall the estimates of $\mathbf{E}(T)$ obtained above and recall the upper bound $\mathbf{P}\{T > k\}$ in terms of $I_k^+ - I_k^-$. □

Remark 2. Part (2) of the Theorem 3 is only useful if $\mathcal{A}(f) < \infty$. For most densities, $\mathcal{A}(f) < \infty$, as can be seen from this simple sufficient condition. Let the modulus of continuity be
$$\omega(\delta) = \sup_{\|x-y\|\le\delta} |f(x) - f(y)|, \ \ \delta > 0.$$
We have $\mathcal{A}(f) < \infty$ if
$$\sum_{k=0}^{\infty} \omega\left(\frac{\sqrt{d}}{2^k}\right) < \infty$$
because
$$\begin{aligned} I_k^+ - I_k^- &= \frac{1}{2^{dk}} \sum_{R^\star\in\mathcal{P}_k^\star} \left(\sup_{x\in R^\star} f(x) - \inf_{x\in R^\star} f(x)\right) \\ &\le \frac{1}{2^{dk}} \sum_{R^\star\in\mathcal{P}_k^\star} \sup_{x,x'\in R^\star} |f(x) - f(x')| \end{aligned}$$

$$\leq \omega\left(\frac{\sqrt{d}}{2^k}\right).$$

It suffices that as $\delta \downarrow 0$, $\omega(\delta) = O\big({}^1\!/_{\log^{1+\alpha}(\delta^{-1})}\big)$ or $\omega(\delta) = O(\delta^\alpha)$ for some $\alpha > 0$.

Remark 3. The number of bits used by our algorithm behaves as $d \log_2^+ \left(\frac{1}{\epsilon}\right) + O(1)$ as $\epsilon \downarrow 0$, and thus matches the lower bound mentioned earlier.

5 A rejection algorithm for densities with non-compact support

In general, we use von Neumann's rejection method when we know a density g for which random variate generation is "easy", and can verify that

$$\sup_{x\in\mathbb{R}} \frac{f(x)}{g(x)} = C < \infty,$$

where C is a known constant:

Algorithm 5 General rejection algorithm

```
repeat
   Generate $X$ with density $g$
   Generate $U$ uniformly on $[0,1]$
   if $Cg(X)U < f(X)$ then
      return $X$ {Exit: $X$ is accepted}
   end if
end repeat
```

The expected number of iterations of Algorithm 5 is C. We offer a generalization of Algorithm 5 for this situation under a certain number of assumptions. Assume for now that $d = 1$, and that we can compute both G and G^{-1}, where G is the c.d.f. for g. Assume furthermore that we have an oracle for

$$\sup_{x\in R} \frac{f(x)}{g(x)} \quad \text{and} \quad \inf_{x\in R} \frac{f(x)}{g(x)}$$

for all intervals R of $\mathbb{R}$. One may be able to work things out with oracles for $\sup f$, $\inf f$, $\sup g$, and $\inf g$ as well but we opt to take the more convenient approach.

Define $C = \sup_{x\in\mathbb{R}} \frac{f(x)}{g(x)}$, which is known thanks to our oracle. The goal is to decompose

$$\big\{(x, y) : \; y \leq f(x)\big\},$$

as before, into regions for which random variate generation is "easy". For a bounded density on $[0,1]$, we are content with the rectangles. This can be mimicked by transforming the x-axis with

$$x \mapsto G(x)$$

since G is monotone and continuous. Using this transformation, we note that if X has density g, then $G(X)$ is uniform on $[0,1]$. Furthermore, note that if $u = G(x)$, then

$$\frac{f \circ G^{-1}(u)}{g \circ G^{-1}(u)} = \frac{f(x)}{g(x)} \overset{\text{def}}{=} \tilde{f}(u),$$

where $\tilde{f}$ is a density on $[0,1]$ on which we can use our sup-inf oracle. Since $\tilde{f} \le C$, we can use the quadtree method of Algorithm 4 to select a rectangle R_i with probability $\lambda(R_i)$ in the decomposition

$$\big\{(u,v): \ v \le \tilde{f}(u), \ u \in [0,1]\big\} = \bigcup_{i \in \mathbb{N}} \big\{R_i : \ R_i \text{ is an accepting rectangle}\big\}.$$

This decomposition is valid, and the procedure halts with probability one, if $\tilde{f}$ is Riemann-integrable. Put differently, it works if

$$\tilde{f}(u) = \frac{f \circ G^{-1}(u)}{g \circ G^{-1}(u)}, \ \ 0 < u < 1,$$

is Riemann-integrable. The expected number of bits required in the decision phase of the algorithm (selecting a leaf of the quadtree) is bounded as in Theorem 3, when applied to $\tilde{f}$. It is bounded by

$$\Delta \overset{\text{def}}{=} 2\left(4C + \sum_{k=0}^{\infty} \left(I_k^+ - I_k^-\right)\right),$$

where I_k^+ and I_k^- are the Riemann approximation of $\int_0^1 \tilde{f}(u)\mathrm{d}u$ for a grid of 2^k equal intervals that partition $[0,1]$.

We only need to analyze the second phase—the method to generate an ϵ-approximation once a rectangle R has been selected, i.e., the method that starts by accepting rectangle $R = [u_1, u_2] \times [v_1, v_2] \subseteq [0,1] \times [0,C]$ as illustrated by Figure 4 and outputs X_ϵ such that $\|X_\epsilon - G^{-1}(U)\| \le \epsilon$ where X, X_ϵ are coupled and (U,V) is uniform in R. This can be achieved by bisection, noting that $G^{-1}(U)$ has distribution function G restricted to $[G^{-1}(u_1), G^{-1}(u_2)]$.

A figure illustrating the principle is at the beginning of the next page.

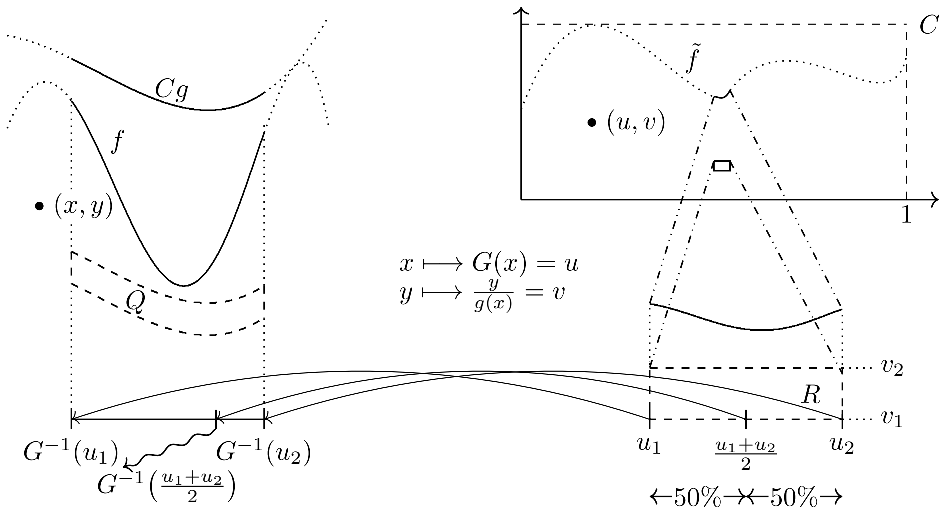

Figure 4: An accepting uniform random rectangle R and the bisection of its back-transformation Q: $(u,v) \in R$ if and only if $(x,y) \in Q$.

Remark 4. Note that with $x_1 = G^{-1}(u_1)$ and $x_2 = G^{-1}(u_2)$, we have

$$\lambda(R) = (u_2 - u_1)(v_2 - v_1) = \int_{u_1}^{u_2} (v_2 - v_1)\mathrm{d}u = \int_{x_1}^{x_2} (v_2 - v_1)(g(x)\mathrm{d}x) = \lambda(Q).$$

Also if (u,v) is such that $v < \tilde{f}(u)$, then the corresponding (x,y) point is such that

$$y = vg(x) < \tilde{f}(u)g(x) = \left(\frac{f(x)}{g(x)}\right)g(x) = f(x),$$

and similarly for $v > \tilde{f}(u)$.

The inversion Algorithm 6 of Devroye and Gravel [4], which is extension of a similar method for the discrete case first proposed by Han and Hoshi [7] is summarized as follows:

The algorithm is at the beginning of the next page.

Algorithm 6 Inversion/Bisection

```
Input: $u_1$, $u_2$ such that $u_2 > u_1$ {$u_2 - u_1$ is the width of an accepting rectangle.}
1:  $x_1 \leftarrow G^{-1}(u_1)$
2:  $x_2 \leftarrow G^{-1}(u_2)$
3:  repeat
4:    if $|x_2 - x_1| \leq 2\epsilon$ then
5:      $X_\epsilon \leftarrow \frac{x_1+x_2}{2}$ {Midpoint}
6:      return $X_\epsilon$
7:    else
8:      $\gamma \leftarrow \frac{u_1+u_2}{2}$
9:      $B \leftarrow$ random unbiased bit {To choose a random side.}
10:     if $B = 0$ then
11:       $u_2 \leftarrow \gamma$
12:       $x_2 \leftarrow G^{-1}(u_2)$
13:     else
14:       $u_1 \leftarrow \gamma$
15:       $x_1 \leftarrow G^{-1}(u_1)$
16:     end if
17:   end if
18: end repeat
```

This algorithm picks a uniform subinterval and, if permitted to run forever, would produce a random variable with distribution function G restricted to $[G^{-1}(u_1), G^{-1}(u_2)]$ as is illustrated in Figure 4. So, for random variate generation, we only replace line (17) of Algorithm 4 by Algorithm 6 where $[u_1, u_2] = R^*$, and note that elsewhere in Algorithm 4, f must be replaced by $\tilde{f}$.

Theorem 4. *The expected number of bits used by Algorithm 6 is not more than*

$$3 + \sum_{j \in \mathbb{Z}} F(I_j) \log_2 \left(\frac{1}{F(I_j)} \right),$$

where $I_j = [2\epsilon j, 2\epsilon(j+1))$, and $F\big([a,b)\big) \overset{def}{=} F(b) - F(a)$. In particular, if that sum is finite for $\epsilon = 1$ and if $\int f \log_2 \big(1/f\big) > -\infty$, then, as $\epsilon \downarrow 0$, the expected number of bits does not exceed

$$\log_2 \left(\frac{1}{\epsilon} \right) + \int f \log_2 \left(\frac{1}{\epsilon} \right) + 5 + o(1).$$

Remark 5. Theorem 4 establishes that Algorithm 6 is optimal to within an additive constant. In particular, its main term, $\log_2 \big(1/\epsilon\big)$, and second term, the differential entropy $\int f \log_2 \big(1/f\big)$, match the lower bound.

Remark 6. The expected number of bits required in the decision phase of the algorithm, Δ, is finite under smoothness conditions on $\tilde{f}$. It depends also on C, but clearly not on ϵ.

Proof of Theorem 4. Let us denote an accepting rectangle R_i and its projection by $R_i^\star$. So, if $R_i^\star = [u_i, v_i]$, then $R_i = [u_i, v_i] \times [\alpha_i, \alpha_i + Cq_i]$, where $0 \le \alpha_i \le \alpha_i + Cq_i \le C$, $q_i \in [0,1]$. The probability mass of R_i is $p_i \overset{\text{def}}{=} (v_i - u_i)Cq_i$.

By the mapping G^{-1}, R_i gets mapped to a contiguous region Q_i, of projection $Q_i^\star \overset{\text{def}}{=} [a_i, b_i]$, with

$$a_i = G^{-1}(u_i) \ , \ b_i = G^{-1}(v_i),$$

and thus,

$$v_i - u_i = \int_{a_i}^{b_i} g = G(Q_i^\star) = \frac{p_i}{Cq_i}.$$

Here we use the notation $G([a_i, b_i]) = G(b_i) - G(a_i)$. We also note that for all x,

$$\sum_{i:\, x \in Q_i^\star} Cq_i g(x) = f(x).$$

Define a regular 2ϵ-grid on $\mathbb{R}$ with intervals $I = [2\epsilon j, 2\epsilon(j+1))$ for all $j \in \mathbb{Z}$.

If we exit with rectangle R_i, then the bisection phase of the algorithm takes an expected number of bits bounded by

$$3 + \sum_{j \in \mathbb{Z}} \xi_{ji} \log_2 \left(\frac{1}{\xi_{ji}}\right),$$

where

$$\xi_{ji} = \frac{G\big(I_j \cap Q_i^\star\big)}{G(Q_i^\star)} \ , \ j \in \mathbb{Z}$$

is a probability vector in j. This result is due to the observation that the bisection method is equivalent to the algorithm analyzed by Devroye and Gravel [4] and Gravel [5] in the context of the discrete distribution algorithm by Han and Hoshi [7]. Thus, the expected number of bits, averaged over all R_i, is not more than

$$3 + \sum_{i \in \mathbb{Z}} p_i \sum_{j \in \mathbb{Z}} \xi_{ji} \log_2 \left(\frac{1}{\xi_{ji}}\right). \tag{7}$$

By the concavity of $u \log_2 \big(1/u\big)$ in u, we have by Jensen's inequality that (7) is not more than

$$3 + \sum_{j \in \mathbb{Z}} \left(\sum_{i \in \mathbb{Z}} p_i \xi_{ji}\right) \log_2 \left(\frac{1}{\sum_{i \in \mathbb{Z}} p_i \xi_{ji}}\right).$$

But

$$\begin{aligned}\sum_{i\in\mathbb{Z}} p_i\xi_{ji} &= \sum_{i\in\mathbb{Z}} p_i \frac{G\big(I_j \cap Q_i^\star\big)}{G(Q_i^\star)}\\ &= \sum_{i\in\mathbb{Z}} Cq_i G(I_j \cap Q_i^\star)\\ &= \sum_{i\in\mathbb{Z}} Cq_i \int_{I_j} \mathbb{1}_{\{x\in Q_i^\star\}} g(x)\mathrm{d}x\\ &= \int_{I_j}\left(\sum_{i\in\mathbb{Z}} Cq_i \mathbb{1}_{\{x\in Q_i^\star\}}\right) g(x)\mathrm{d}x\\ &= \int_{I_j} f(x)\mathrm{d}x\\ &\stackrel{\text{def}}{=} F(I_j),\end{aligned}$$

where F is the distribution function of f, and $F(I_j) = F(b_j) - F(a_j)$. Thus the expected number of bits in the bisection phase does not exceed

$$3 + \sum_{j\in\mathbb{Z}} F(I_j)\log_2\left(\frac{1}{F(I_j)}\right). \tag{8}$$

In the last term, we recognize the entropy defined by the probability vector $\big(F(I_j)\big)_{j\in\mathbb{Z}}$.

A theorem due to Csiszár [2] established that if $\big(F(I_j)\big)_{j\in\mathbb{Z}}$ has a finite entropy for some $\epsilon > 0$, and $\int f \log_2\big(1/f\big) > -\infty$, then as $\epsilon \downarrow 0$,

$$(8) \leq \int f \log_2 \frac{1}{f} + \log_2 \frac{1}{\epsilon} + 5 + o(1).$$

The "5" can be replaced by "3" if in addition f is bounded and decreasing on its support, $[0,\infty)$ (see Gravel and Devroye [4]).

□

6 Conclusion and outlook

The extension of our results to dimensions greater than one for densities with unbounded support should pose no big problems. With the oracles introduced in our modification of von Neumann's method, we believe that it is impossible to design a rejection algorithm for densities that are not Riemann-integrable, so the question of the design of a universally valid rejection algorithm under the random bit model remains open.

Acknowledgment

The authors would like to thank all three referees for their feedback. They are invited to Luc's house for drinks if they wish to come out of the closet. Luc Devroye's research was supported by an NSERC Discovery Grant. Claude Gravel thanks Gilles Brassard.

References

[1] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley, New-York, 1991.

[2] I. Csiszár, *Some remarks on the dimension and entropy of random variables*, Acta Mathematica Academiae Scientiarum Hungarica, 12 (1961), pp. 399–408.

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

[4] L. Devroye and C. Gravel, *Sampling with arbitrary precision*, 2015. `http://arxiv.org/abs/1502.02539`.

[5] C. Gravel, *Échantillonnage de distributions non uniformes en précision arbitraire et protocoles d'échantillonnage exact distribué des distributions discrètes quantiques*, PhD thesis, Université de Montréal, 2015. `https://papyrus.bib.umontreal.ca/xmlui/handle/1866/12337`.

[6] G. Brassard, L. Devroye and C. Gravel, *Exact classical simulation of the quantum-mechanical GHZ distribution*, IEEE Transactions on Information Theory, 62 (2016), pp. 876–890.

[7] T. S. Han and M. Hoshi, *Interval algorithm for random number generation*, IEEE Transactions on Information Theory, 43 (1997), pp. 599–611.

[8] C. F. F. Karney, *Sampling exactly from the normal distribution*, ACM Trans. Math. Softw., 42 (2016), pp. 1–14.

[9] D. E. Knuth and A. C.-C. Yao, *The complexity of nonuniform random number generation*, in Algorithms and Complexity: New Directions and Recent Results., J. F. Traub, ed., New York, 1976, Carnegie-Mellon University, Computer Science Department, Academic Press, pp. 357–428. Reprinted in Knuth's *Selected Papers on Analysis of Algorithms* (CSLI, 2000).

[10] J. von Neumann, *Various techniques used in connection with random digits. Monte Carlo Methods*, National Bureau of Standards, 12 (1951), pp. 36–38.

[11] S. T. Rachev and L. Rüschendorf, *Mass Transportation Problems: Volume 1: Theory*, Springer (Probability and Its Applications), 1998.

[12] A. Rényi, *On the dimension and entropy of probability distributions*, Acta Mathematica Academiae Scientiarum Hungarica, 10 (1959), pp. 193–215.

[13] H. Samet, *Foundations of Multidimensional and Metric Data Structures*, Morgan Kaufmann, Elsevier/Morgan Kaufmann, San Mateo, 2006.

A Riemann integrability and the sup/inf oracle

In this section, we contruct a family of densities that are not Riemann-integrable for which the oracle is useless. Let $\delta \in [0, 1/3)$ be a parameter, and let $I_\delta \subseteq [0, 1]$ be a Cantor-like set constructed below. Setting

$$f_\delta(x) = \frac{1}{\lambda(I_\delta)} \mathbb{1}_{\{x \in I_\delta\}},$$

where λ is the Lebesgue measure, we have (see below) $\lambda(I_\delta) = (1 - 3\delta)/(1 - 2\delta)$. The Lebesgue measure of the set of discontinuities is $1 - \lambda(I_\delta)$ and is zero only if $\delta = 0$. The case of $\delta = 0$ corresponds to the usual uniform density on $[0, 1]$ that is Riemann-integrable. All cases of $\delta \in (0, 1/3)$ are Lebesgue integrable but not Riemann-integrable because the set of discontinuities is non-zero and yet possess a cumulative distribution function. For every $\epsilon > 0$ and every $x \in I_\delta$ we have

$$\inf_{[x-\epsilon, x+\epsilon]} f(x) = 0,$$

and

$$\sup_{[x-\epsilon, x+\epsilon]} f(x) = \frac{1 - 2\delta}{1 - 3\delta}.$$

Since these boundaries are invariant under changes of ϵ, Algorithm 4, when used for rejection, say, from a uniform density, has an infinite loop with positive probability. It is, therefore, essential that Riemann-integrable densities are considered as that the supremum and infimum returned by the oracle over given small intervals converge to each other as intervals shrink.

For the construction of I_δ, we recursively remove middle *open* subintervals of geometrically decreasing sizes. Let $I_{j,k} \subset [0, 1]$ for $j \in \mathbb{N} \setminus \{0\}$ and $k = 0, \ldots, 2^j - 1$ be the 2^j *closed* subintervals that are *left* once the middle parts are removed from the previous subintervals $I_{j-1,k}$ with $k = 0$ corresponding to the leftmost subinterval and so on. Initially, $I_{0,0} = [0, 1]$. For all $j \in \mathbb{N} \setminus \{0\}$ and $k \in \{0, \ldots, 2^j - 1\}$, the length of a removed middle part is δ^j. For all $j \in \mathbb{N} \setminus \{0\}$, the total length *not* removed at the j-th step is $2^j \lambda(I_{j,0})$ because the subintervals are of the same length. Let I_δ be the limiting subset of $[0,1]$ that is left, i.e.,

$$I_\delta = \bigcap_{j=1}^{\infty} \bigcup_{k=0}^{2^j-1} I_{j,k}.$$

We compute $\lambda(I_\delta)$ as follows:

$$\begin{aligned}
\lambda(I) &= \lim_{j\to\infty} \sum_{k=0}^{2^j-1} \lambda(I_{j,k}) \text{ (by the definition of } I_\delta) \\
&= \lim_{j\to\infty} 2^j \lambda(I_{j,0}), \\
\lambda(I_{1,0}) &= \frac{1}{2} - \frac{\delta}{2}, \\
\lambda(I_{j,0}) &= \frac{1}{2}\lambda(I_{j-1,0}) - \frac{\delta^j}{2} \\
&= \frac{1}{2^j} - \frac{\delta}{2^j} - \frac{\delta^2}{2^{j-1}} - \ldots - \frac{\delta^j}{2} \text{ for } j \geq 1,
\end{aligned}$$

and therefore

$$2^j \lambda(I_{j,0}) = 1 - \delta \sum_{i=0}^{j-1} (2\delta)^i = 1 - \delta \left(\frac{(2\delta)^j - 1}{2\delta - 1} \right),$$

so that

$$\lambda(I_\delta) = \frac{1-3\delta}{1-2\delta}.$$

B A naive modification to the general rejection method that is incorrect

A trivial, but incorrect, modification to Algorithm 3 would be:

```
1: repeat
2:     By bisection, generate $X_\epsilon$ on $[0,1]^d$ such that $X_\epsilon$ is an $\epsilon$-approximation of $X$.
3:     Generate $U$ uniformly on $[0,1]$.
4:     Decide "$U \leq \frac{1}{C} f(X_\epsilon)$" by using two bits in expected value.
5:     If the condition from the previous line is satisfied, then return $X_\epsilon$.
6: end repeat
```

This attempt leads to failure. Let A be the support of X_ϵ, which is necessarily contained in a fixed countable subset of $[0,1]^d$. Then given any Riemann-integrable density

g, take a finite subset $A^\star$ of A, and modify g on $A^\star$ by setting

$$f(x) = \begin{cases} g(x) & \text{if } x \notin A^\star, \\ C & \text{if } x \in A^\star. \end{cases}$$

We have that f is still a Riemann-integrable density bounded by C (and its set of discontinuities is of measure 0 because $A^\star$ is countable) but since $f(X_\epsilon) = C$ if $X_\epsilon \in A^\star$, we accept all $X_\epsilon \in A^\star$, regardless of the density g we started with. Since $\mathbf{P}\{X_\epsilon \in A^\star\} > 0$, we make an error with positive probability.

If we set

$$f(x) = \begin{cases} g(x) & \text{if } x \notin A, \\ 0 & \text{if } x \in A, \end{cases}$$

then g is no longer Riemann-integrable, and in that case, $f(X_\epsilon) = 0$ with probability one, and therefore, the algorithm loops forever, regardless of the choice of g.

This simple example shows the necessity of the oracle and of the condition of Riemann-integrability.