# Optimally Adaptive Integration of Univariate Lipschitz Functions

Ilya Baran* Erik D. Demaine* Dmitriy A. Katz†

**Abstract**

We consider the problem of approximately integrating a Lipschitz function $f$ (with a known Lipschitz constant) over an interval. The goal is to achieve an additive error of at most $\epsilon$ using as few samples of $f$ as possible. We use the adaptive framework: on all problem instances an adaptive algorithm should perform almost as well as the best possible algorithm tuned for the particular problem instance. We distinguish between DOPT and ROPT, the performances of the best possible deterministic and randomized algorithms, respectively. We give a deterministic algorithm that uses $O(\mathrm{DOPT}(f,\epsilon)\cdot\log(\epsilon^{-1}/\mathrm{DOPT}(f,\epsilon)))$ samples and show that an asymptotically better algorithm is impossible. However, any deterministic algorithm requires $\Omega(\mathrm{ROPT}(f,\epsilon)^2)$ samples on some problem instance. By combining a deterministic adaptive algorithm and Monte Carlo sampling with variance reduction, we give an algorithm that uses at most $O(\mathrm{ROPT}(f,\epsilon)^{4/3}+\mathrm{ROPT}(f,\epsilon)\cdot\log(1/\epsilon))$ samples. We also show that any algorithm requires $\Omega(\mathrm{ROPT}(f,\epsilon)^{4/3}+\mathrm{ROPT}(f,\epsilon)\cdot\log(1/\epsilon))$ samples in expectation on some problem instance $(f,\epsilon)$, which proves that our algorithm is optimal.

## 1 Introduction

We consider the problem of approximating a definite integral of a univariate Lipschitz function (with known Lipschitz constant) to within $\epsilon$ using the fewest possible samples. The function is given as a black box: sampling it at a parameter value is the only allowed operation. It is easy to show that $\Theta(\epsilon^{-1})$ samples are necessary and sufficient for a deterministic algorithm in the worst case (see, e.g., [13]). The results in [1] imply a Monte-Carlo method that requires only $\Theta(\epsilon^{-2/3})$ samples in the worst case.

**The Adaptive Framework.** The univariate Lipschitz integration problem becomes more interesting in the adaptive setting. The motivation is that, for a given $\epsilon$, some problem instances have much lower complexity than others. For example, if $f(x)=Lx$, where $L$ is the Lipschitz constant, then evaluating $f$ at the endpoints of the interval over which the integral is taken is sufficient to solve the problem for any $\epsilon$. Thus, it is desirable to have an algorithm that is guaranteed to use fewer samples on easier problem instances. Such an algorithm is called *adaptive*. We formalize this notion by defining the difficulty of a problem as the performance of the best possible algorithm on that problem:

**Definition 1** *Let $\mathcal{P}$ be a class of problem instances. Let $\mathcal{A}$ be the set of all correct algorithms for $\mathcal{P}$ (among some reasonable class of algorithms). Let* $\mathrm{COST}(A,P)$ *be the performance of algorithm $A\in\mathcal{A}$ on problem instance $P\in\mathcal{P}$. Define* $\mathrm{OPT}(P)=\min_{A\in\mathcal{A}}\mathrm{COST}(A,P)$. *We use* DOPT *when $\mathcal{A}$ is the set of deterministic algorithms and* ROPT *when $\mathcal{A}$ is the set of randomized algorithms that are correct on each $P\in\mathcal{P}$ with probability at least* $2/3$.

By definition, for every problem instance $P$, there is an algorithm whose cost on $P$ is $\mathrm{OPT}(P)$. A good adaptive algorithm is a single algorithm whose cost is not much greater than $\mathrm{OPT}(P)$ for *every* problem instance $P$. Therefore, an adaptive guarantee is in general much stronger than a worst-case guarantee.

The ultimate goal of investigating a problem in the adaptive framework is to design an "optimally adaptive" algorithm. Suppose $\mathcal{P}$ is the set of problem instances and each problem instance $P\in\mathcal{P}$ has certain natural parameters, $v_1(P),\ldots,v_k(P)$, with the first parameter $v_1(P)=\mathrm{OPT}(P)$. An algorithm is *optimally adaptive* if its performance on every problem instance $P\in\mathcal{P}$ is within a constant factor of every algorithm's worst-case performance on the family of instances with the same values for the parameters:

*MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge, MA 02139, USA, {ibaran, edemaine}@mit.edu

†Sloan School of Management, Massachusetts Institute of Technology, 50 Memorial Drive, Cambridge, MA 02142, USA, dimdim@mit.edu

$\{P' \in \mathcal{P} \mid v_i(P') = v_i(P) \text{ for all } i\}$. Note that this definition depends on the choice of parameters, so in addition to OPT, we need to choose reasonable parameters, such as $\epsilon$, the desired output accuracy.

**Related Work.** Approximate definite integration is well-studied in numerical analysis (see, e.g., [5]). However, most practical algorithms, do not assume a Lipschitz bound known in advance (or something similar) and therefore cannot guarantee a bound on the error, only convergence on a class of functions. Information-based complexity [12] tends to use assumptions that lead to stronger query complexity guarantees than convergence, but it does not use the adaptive framework. The term "adaptive" in that literature refers to an algorithm's ability to use previous sample results to determine where to sample next. To our knowledge, this paper is the first use of adaptive analysis for integration.

The authors in [14] describe what is essentially our deterministic integration algorithm (they call it "estimation," but they use mean, rather than maximum error) but they only prove that it is optimal in the greedy sense—no global optimality claims are made.

For other problems, optimally adaptive algorithms have been previously designed in the context of set operations [6], aggregate ranking [7], and independent set discovery in [4]. Lipschitz functions also lend themselves well to adaptive algorithms. It is shown in [8] that Piyavskii's algorithm [10] for minimizing a univariate Lipschitz function performs $O(\text{OPT})$ samples. [3] gives an adaptive algorithm for minimizing the distance from a point to a Lipschitz curve that is within a logarithmic factor of OPT. [2] gives adaptive algorithms for several problems on Lipschitz functions.

**Our Results.** In this paper we give a deterministic algorithm that makes $O(\text{DOPT} \cdot \log(\epsilon^{-1}/\text{DOPT}))$ samples. We also prove a matching lower bound on deterministic algorithms. When comparing to ROPT, however, we show that any deterministic adaptive algorithm uses $\Omega(\text{ROPT}^2)$ samples on some problem instance. We present a randomized adaptive algorithm, LIPSCHITZ-MC-INTEGRATE, that always uses $O(\text{ROPT}^{4/3} + \text{ROPT} \cdot \log(\epsilon^{-1}))$ samples and prove a matching lower bound.

We therefore give optimally adaptive algorithms for the Lipschitz integration problem in the deterministic and randomized settings. Although the algorithms are simple, in both cases analyzing their adaptive performance is nontrivial. To our knowledge, LIPSCHITZ-MC-INTEGRATE is the first randomized optimally adaptive algorithm. Also, a simple corollary of the randomized lower bound is that the non-adaptive algorithm based on the results in [1] is optimal in the worst case.

In practice, problems where we know the Lipschitz constant and nothing else about the function do not often occur. However, to simplify implementation and analysis, practitioners sometimes make no use of the extra knowledge and only focus on the Lipschitz condition (e.g. for implicit surface ray tracing in [9]). Although adaptive optimality is not meaningful in such a case, an adaptive upper bound does provide a meaningful guarantee.

Some of the results in this paper, primarily in Sections 3 and 4, are based on the first author's master's thesis [2].

## 2 Problem Basics

We start by giving a precise formulation of the problem we consider:
Problem LIPSCHITZ-INTEGRATION:

| | |
|---|---|
| **Given:** | $(f, a, b, L, \epsilon)$ |
| **Such that:** | $f\colon [a, b] \to \mathbb{R}$ |
| | and for $x_1, x_2 \in [a, b]$, $\lvert f(x_2) - f(x_1)\rvert \le L\lvert x_2 - x_1\rvert$ |
| **Compute:** | $I \in \mathbb{R}$ such that $\left\lvert I - \int_a^b f(x)\,dx \right\rvert \le \epsilon$ |

A randomized algorithm needs to be correct with probability at least 2/3. Sampling $f$ is the only way to obtain information about it.

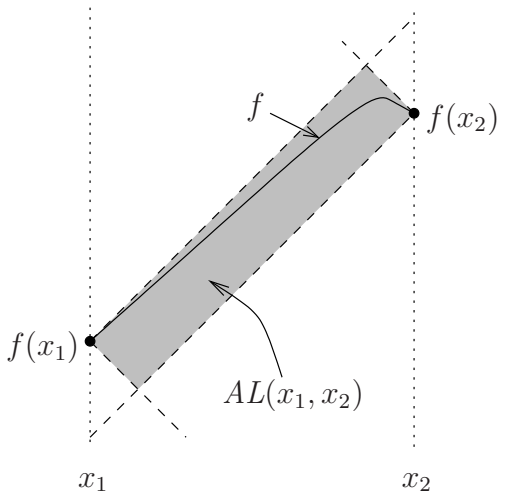


Figure 1: Illustration of area looseness. Lipschitz bounds are dashed.

Some input parameters can be eliminated without loss of generality. The problem instance $(f, a, b, L, \epsilon)$ is equivalent to the problem instance $(\hat{f}, 0, 1, 1, \epsilon/L(b-a)^2)$ where $\hat{f}(x) = f\left(\frac{x-a}{b-a}\right)/L(b-a)$, so we can assume without loss of generality that $a = 0$, $b = 1$, and $L = 1$.

We now develop some basic tools we will need for discussing and analyzing the algorithms. Essentially, we show how to make use of the Lipschitz condition to bound the error of our estimates.

When we sample $f$ at a point, the Lipschitz condition tells us that $f$ lies between the lines of slope 1 and $-1$ crossing at that point. If we sample $f$ at two points, the four bounding lines define a rectangle in which $f$ must lie between these two points. We thus have upper and lower bounds for the integral of $f$ on the interval between these points, and the difference between these bounds is precisely the area of the rectangle (see Figure 1). We call this difference *area looseness*, and it depends on both the length of the interval and the values of $f$ at the sampled points. A greater difference between values of $f$ (a steeper function) results in a smaller area looseness. Area looseness is twice the worst-case integration error of approximating $f$ by a straight line on that interval. Formally, we define area looseness as follows:

**Definition 2** *Given a Lipschitz function $f$ on $[0,1]$, define the* area looseness *of a subinterval $[x_1, x_2]$ of $[0,1]$ as*

$$AL_f(x_1, x_2) = \frac{(x_2 - x_1)^2 - (f(x_1) - f(x_2))^2}{2}.$$

*When it is clear which $f$ we are talking about, we simply write $AL(x_1, x_2)$.*

Our analysis relies on area looseness being well behaved. The following proposition shows that it has the properties one would expect a bound on integration error to have and that an additional sample in the middle of the interval decreases total area looseness quickly.

**Proposition 1** *Area-looseness has the following properties:*
*(1) $0 \le AL(x_1, x_2) \le (x_2 - x_1)^2/2$.*
*(2) If $x_1' \le x_1 < x_2 \le x_2'$ then $AL(x_1, x_2) \le AL(x_1', x_2')$.*
*(3) If $x \in [x_1, x_2]$, then $AL(x_1, x) + AL(x, x_2) \le AL(x_1, x_2)$.*
*(4) $AL\big(x_1, \frac{x_1+x_2}{2}\big) + AL\big(\frac{x_1+x_2}{2}, x_2\big) \le AL(x_1, x_2)/2$.*

**Proof:** (1) Note that $(x_2 - x_1)^2 \ge (f(x_1) - f(x_2))^2 \ge 0$, where the first inequality follows form the Lipschitz condition. Therefore, (1) holds.

(2) This follows directly from (1) and (3).

(3) The Lipschitz condition implies that $|x_1 - x| \geq |f(x_1) - f(x)|$ and $|x_2 - x| \geq |f(x_2) - f(x)|$, and therefore $|(x_1 - x)(x_2 - x)| \geq |(f(x_1) - f(x))(f(x_2) - f(x))|$. In addition, $(x_1 - x)(x_2 - x) \leq 0$, so $(f(x_1) - f(x))(f(x_2) - f(x)) \geq (x_1 - x)(x_2 - x)$. Multiplying through, we get that

$$f(x_1)f(x_2) + f(x)^2 - f(x_1)f(x) - f(x_2)f(x) \geq x_1x_2 + x^2 - xx_1 - xx_2.$$

Rearranging, we get that

$$-x_1x_2 + f(x_1)f(x_2) \geq x^2 - xx_1 - xx_2 - f(x)^2 + f(x)f(x_1) + f(x)f(x_2).$$

Adding $\frac{x_2^2+x_1^2-f(x_1)^2-f(x_2)^2}{2}$ to both sides to complete the squares, we get that

$$\frac{(x_2 - x_1)^2 - (f(x_1) - f(x_2))^2}{2} \geq \frac{(x - x_1)^2 - (f(x_1) - f(x))^2}{2} + \frac{(x_2 - x)^2 - (f(x_1) - f(x))^2}{2}.$$

From the definition of $AL$, the last inequality is equivalent to (3).

(4) Let $x_m = (x_1 + x_2)/2$. The proposition claims that if $f$ is Lipschitz, then

$$\frac{(x_2 - x_1)^2 - (f(x_2) - f(x_1))^2}{2} \geq (x_m - x_1)^2 - (f(x_m) - f(x_1))^2 + (x_2 - x_m)^2 - (f(x_2) - f(x_m))^2.$$

Because $(x_m - x_1) = (x_2 - x_m) = (x_2 - x_1)/2$, the claim can be written as

$$\frac{(x_2 - x_1)^2 - (f(x_2) - f(x_1))^2}{2} \geq \frac{(x_2 - x_1)^2}{2} - (f(x_m) - f(x_1))^2 - (f(x_2) - f(x_m))^2.$$

Thus, we need to show that

$$(f(x_2) - f(x_1))^2 \leq 2(f(x_m) - f(x_1))^2 + 2(f(x_2) - f(x_m))^2.$$

Notice that the right hand side can be written as $af(x_m)^2+bf(x_m)+c$, where $a = 4$ and $b = -4f(x_1)-4f(x_2)$. This expression has a single global minimum at $f(x_m) = -b/2a$, which is $f_m = (f(x_1)+f(x_2))/2$. Therefore, we have

$$2(f_m - f(x_1))^2 + 2(f(x_2) - f_m)^2 \leq 2(f(x_m) - f(x_1))^2 + 2(f(x_2) - f(x_m))^2.$$

But we can rewrite the left hand side as

$$2\left(\frac{f(x_2) - f(x_1)}{2}\right)^2 + 2\left(\frac{f(x_2) - f(x_1)}{2}\right)^2 = (f(x_2) - f(x_1))^2,$$

which gives us the claim. □

For the lower bounds, both on OPT and on adaptive algorithms, we need "extremal" Lipschitz functions, whose integral is either maximal or minimal, given the samples. We call these functions $HI$ and $LO$. We also define *looseness*, the maximum difference between $HI$ and $LO$ over an interval.

**Definition 3** *Given a Lipschitz function $f$, and $0 \leq a < b \leq 1$, define the Lipschitz functions $HI_a^b$ and $LO_a^b$ on $[a, b]$ asfollows:*

$$HI_a^b(x) = \min(f(a) + x - a, f(b) + b - x)$$

$$LO_a^b(x) = \max(f(a) - x + a, f(b) - b + x)$$

*Also define looseness $L_f$ as $L_f(a, b) = b - a - |f(b) - f(a)|$.*

See Figure 2. We now present some properties of our definitions.

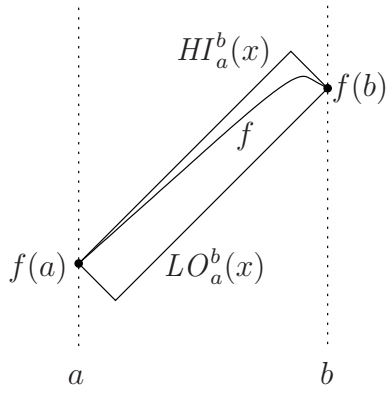


Figure 2: Illustration of $HI$ and $LO$.

**Proposition 2** *Given a Lipschitz function $f$, the functions $HI_a^b$ and $LO_a^b$ have the following properties:*

*(1) If $g$ is Lipschitz, $g(a) = f(a)$, and $g(b) = f(b)$, then for $x \in [a,b]$, $HI_a^b(x) \geq g(x) \geq LO_a^b(x)$.*

*(2)* $AL(a,b)/(b-a) \leq \max_{x\in[a,b]}(HI_a^b(x) - LO_a^b(x)) = L(a,b) \leq 2AL(a,b)/(b-a)$

*(3) The functions have the following integrals:*

$$\int_a^b HI_a^b(x)\,dx = (b-a)\frac{f(a)+f(b)}{2} + AL(a,b)/2$$

$$\int_a^b LO_a^b(x)\,dx = (b-a)\frac{f(a)+f(b)}{2} - AL(a,b)/2$$

.

**Proof:** (1) This inequality follows immediately from the Lipschitz condition on $g$.

(2) We prove the equation for when $f(a) < f(b)$. The proof is analogous for the other case. Note that $HI_a^b(x) \leq f(a)+x-a$ and $LO_a^b \geq f(b)-b+x$. Therefore, $HI_a^b(x) - LO_a^b(x) \leq f(a)-f(b)+b-a = L(a,b)$. On the other hand $HI_a^b((a+b)/2) - LO_a^b((a+b)/2) = (f(a)+(b-a)/2) - (f(b)-(b-a)/2) = L(a,b)$.

For the inequalities, we have: $AL(a,b) = \frac{L(a,b)(b-a+|f(a)-f(b)|)}{2}$. But $b-a \leq b-a+|f(a)-f(b)| \leq 2(b-a)$. Therefore, $AL(a,b) \leq L(a,b)(b-a) \leq 2AL(a,b)$.

(3) We prove the equation $\int_a^b HI_a^b(x)\,dx = (b-a)\frac{f(a)+f(b)}{2} + AL(a,b)/2$. The proof of the second equation is analogous. Note that for $f(a)+x-a \leq f(b)+b-x$ precisely when $x \leq (f(b)-f(a)+a+b)/2$.

Therefore, the integral may be written as

$$\int_a^b HI_a^b(x)\,dx = \int_a^{(f(b)-f(a)+a+b)/2} f(a)+x-a\,dx + \int_{(f(b)-f(a)+a+b)/2}^b f(b)+b-x\,dx =$$

$$= \frac{3f(a)+f(b)+b-a}{4}\cdot\frac{f(b)-f(a)+(b-a)}{2}+$$

$$+\frac{f(a)+3f(b)+b-a}{4}\cdot\frac{f(a)-f(b)+(b-a)}{2} =$$

$$= f(a)\cdot\frac{f(b)-f(a)+(b-a)}{4} + f(b)\cdot\frac{f(a)-f(b)+(b-a)}{4}+$$

$$+\frac{f(a)+f(b)+b-a}{4}\cdot(b-a) =$$

$$= (b-a)\frac{f(a)+f(b)}{2} + \frac{(f(b)-f(a))(f(a)-f(b))}{4} + \frac{(b-a)^2}{4} =$$

$$= (b-a)\frac{f(a)+f(b)}{2} + AL(a,b)/2,$$

as desired. □

**Proposition 3** *Given a Lipschitz function $f$, looseness has the following properties:*
*(1)* $0 \le L(a,b) \le b-a$
*(2) If* $a' \le a \le b \le b'$, *then* $L(a,b) \le L(a',b')$.
*(3) If* $x_1 \le x_2 \le \cdots \le x_n$, *then* $\sum_{i=1}^{n-1} L(x_i,x_{i+1}) \le L(x_1,x_n)$.

**Proof:** (1) This follows immediately from the Lipschitz condition.
(2) By (3) and (1), $L(a',b') \ge L(a',a)+L(a,b)+L(b,b') \ge L(a,b)$
(3) By definition, $\sum_{i=1}^{n-1} L(x_i,x_{i+1}) = x_n - x_1 - \sum_{i=1}^{n-1}|f(x_{i+1})-f(x_i)|$. But by the triangle inequality, $\sum_{i=1}^{n-1}|f(x_{i+1})-f(x_i)| \ge |f(x_n)-f(x_1)|$. □

# 3 Proof Sets

In order to compare the running time of an algorithm on a problem instance to DOPT, we define the concept of a proof set for a problem instance. A set $P$ of points in $[0,1]$ is a *proof set* for problem instance $(f,\epsilon)$ and output $x$ if for every $f'$ that is equal to $f$ on $P$, $x$ is a correct output on $(f',\epsilon)$. In other words, sampling $f$ at a proof set proves the correctness of the output. We say that a set of samples is a proof set for a particular problem instance without specifying the output if some output exists for which it is a proof set.

It is clear from the definition that sampling a proof set is the only way a deterministic algorithm can guarantee correctness: if an algorithm doesn't sample a proof set for some problem instance, we can feed it a problem instance that has the same value on the sampled points, but for which the output of the algorithm is incorrect. Conversely an algorithm can terminate as soon as it has sampled a proof set and always be correct. Thus, DOPT is equal to the size of a smallest proof set.

In order to analyze the deterministic algorithm, we will compare the number of samples it makes to the size of a proof set $P$. We will need some tools for doing this.

Let $P$ be a nonempty finite set of points in $[0,1]$. Consider the execution of an algorithm which samples a function at points on the interval $[0,1)$ (if it samples at 1, ignore that sample). Let $s_1, s_2, \ldots, s_n$ be the sequence of samples that the algorithm performs in the order that it performs them. Let $I_t$ be the set of unsampled intervals after sample $s_t$, i.e., the connected components of $[0,1)-\{s_1,\ldots,s_t\}$, except make each element of $I_t$ half-open by adding its left endpoint, so that the union of all the elements of $I_t$ is $[0,1)$. Let $[l_t,r_t)$ be the element of $I_{t-1}$ that contains $s_t$.

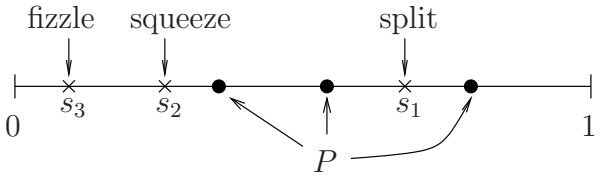


Figure 3: Different types of samples.

Then sample $s_t$ is a:

$$\begin{array}{rll} \textit{split} & \text{if} & [l_t, s_t) \cap P \neq \emptyset \text{ and } [s_t, r_t) \cap P \neq \emptyset \\ \textit{squeeze} & \text{if} & [l_t, s_t) \cap P \neq \emptyset \text{ or } [s_t, r_t) \cap P \neq \emptyset, \text{ but not both} \\ \textit{fizzle} & \text{if} & [l_t, r_t) \cap P = \emptyset. \end{array}$$

These definitions are, of course, relative to $P$. See Figure 3. We can now bound the number of samples of different types:

**Proposition 4** *The number of splits is at most* $|P| - 1$.

**Proof:** Let $a_t$ be the number of elements of $I_t$ that intersect $P$. If $s_t$ is a split, then $a_t - a_{t-1} = 1$, otherwise $a_t = a_{t-1}$. Because the elements of $I_t$ are disjoint, $a_t \leq |P|$. Unless $P = \{1\}$ (in which case, there can be no splits), $a_0 = 1$, so at most $|P| - 1$ splits can occur. □

To bound the number of squeezes, we will first want the following easy fact:

**Proposition 5** *Let* $a_i$, *for* $1 \leq i \leq |P|$, *and* $b$ *be positive real numbers. Suppose that* $\sum_{i=1}^{|P|} a_i \leq \epsilon^{-1}$. *Then* $\sum_{i=1}^{|P|} \log_b a_i \leq |P| \log_b(\epsilon^{-1}/|P|)$.

**Proof:** By the arithmetic-geometric mean inequality, $\sqrt[|P|]{\prod_{i=1}^{|P|} a_i} \leq \frac{\sum_{i=1}^{|P|} a_i}{|P|} \leq \frac{\epsilon^{-1}}{|P|}$. Taking the logarithm with base $b$ of both sides gives us $\frac{\sum_{i=1}^{|P|} \log_b a_i}{|P|} \leq \log_b(\epsilon^{-1}/|P|)$. Multiplying both sides by $|P|$ gives us the desired result. □

**Proposition 6** *Suppose that for all* $i$ *and* $j$ *with* $i \neq j$, $|s_i - s_j| > \epsilon$ *and that for all* $t$, $s_t = (l_t + r_t)/2$. *Then if* $|P| \leq \epsilon^{-1}/2$, *the number of squeezes is at most* $|P| \log_2(\epsilon^{-1}/|P|)$.

**Proof:** Let $IP_t = \{[a, b) \in I_t \mid [a, b) \cap P \neq \emptyset\}$. If $J \in IP_t$, define $S(J)$ to be the number of squeezes that have occurred to intervals containing $J$:

$$S(J) = \sum_{i=1}^{t} \begin{cases} 1 \text{ if } s_i \text{ is a squeeze and } J \subset [r_i, l_i) \\ 0 \text{ otherwise.} \end{cases}$$

We claim the invariant that for all $t$,

$$\sum_{[a,b) \in IP_t} (b - a) \cdot 2^{S([a,b))} = 1. \tag{1}$$

We prove (1) by induction on $t$. The base case $t = 0$ is clear because unless $P$ is trivial, $IP_0 = \{[0, 1)\}$ and $S([0, 1)) = 0$ because there have been no squeezes. For the inductive step, assume (1) holds for $t - 1$. If $s_t$ is a fizzle, no intervals containing points of $P$ are affected and the sum remains the same. If $s_t$ is a split, interval $[l_t, r_t)$ is replaced by $[l_t, s_t)$ and $[s_t, r_t)$ in $IP_t$ and $S([l_t, s_t)) = S([s_t, r_t)) = S([l_t, r_t))$ because no new squeezes have occurred and the sum remains unchanged. Finally, if $s_t$ is a squeeze, the interval $[l_t, r_t)$ in $IP_t$ is replaced by an interval $J$ of half the length, but $S(J) = S([l_t, r_t)) + 1$, so the sum remains the same.

Now, by assumption, each element of $I_t$ is longer than $\epsilon$, so, writing (1) for $t = n$ gives:

$$\sum_{[a,b)\in IP_n} \epsilon \cdot 2^{S([a,b))} < \sum_{[a,b)\in IP_n} (b-a) \cdot 2^{S([a,b))} = 1, \quad \text{which means} \quad \sum_{[a,b)\in IP_n} 2^{S([a,b))} < \epsilon^{-1}.$$

Using Proposition 5, we obtain $\sum_{[a,b)\in IP_n} S([a,b)) < |P| \log_2(\epsilon^{-1}/|P|)$, which implies that the total number of squeezes is at most $|P| \log_2(\epsilon^{-1}/|P|)$. □

We now characterize proof sets for LIPSCHITZ-INTEGRATION.

**Proposition 7** *Let $P = \{x_1, x_2, \ldots, x_n\}$ such that $0 \le x_1 < x_2 < \cdots < x_n \le 1$. Then $P$ is a proof set for problem instance $(f, \epsilon)$ if and only if $x_1^2 + (1 - x_n)^2 + \sum_{i=1}^{n-1} AL(x_i, x_{i+1}) \le 2\epsilon$.*

**Proof:** The value $M = x_1 \cdot f(x_1) + (1 - x_n) \cdot f(x_n) + \sum_{i=1}^{n-1} \left( (x_{i+1} - x_i) \cdot \frac{f(x_i)+f(x_{i+1})}{2} \right)$ is within $\epsilon$ of $\int_0^1 f(x)\,dx$ because

$$\left| x_1 \cdot f(x_1) - \int_0^{x_1} f(x)\,dx \right| \le \frac{x_1^2}{2}, \qquad \left| (1 - x_n) \cdot f(x_n) - \int_{x_n}^1 f(x)\,dx \right| \le \frac{(1-x_n)^2}{2},$$

and for each $i$, $\left| (x_{i+1} - x_i) \cdot \frac{f(x_i)+f(x_{i+1})}{2} - \int_{x_i}^{x_{i+1}} f(x)\,dx \right| \le \frac{1}{2} \cdot AL(x_i, x_{i+1})$ by Proposition 2.

In the other direction, if $x_1^2 + (1-x_n)^2 + \sum_{i=1}^{n-1} AL(x_i, x_{i+1}) > 2\epsilon$, then two functions, $f_{HI}$ and $f_{LO}$, can be constructed such that $f_{HI}(x_i) = f(x_i) = f_{LO}(x_i)$ but $\int_0^1 f_{HI}(x)\,dx - \int_0^1 f_{LO}(x)\,dx > 2\epsilon$, which means that no matter what value an algorithm outputs after sampling $P$, that value will be incorrect for either $f_{HI}$ or $f_{LO}$. □

# 4 Deterministic Algorithm and Analysis

Proposition 7, together with Proposition 1 immediately shows the correctness of a trivial algorithm. Let $n = \lceil \epsilon^{-1}/4 \rceil$ and let the algorithm make $n$ samples, at $\frac{1}{2n}, \frac{3}{2n}, \ldots, \frac{2n-1}{2n}$ and output the integral $M$ as in the proof of Proposition 7. It is correct because the area-looseness of every interval is at most $(1/n)^2/2$. Because there are $n - 1$ intervals, the total area-looseness of all of them is at most $(n-1)/(2n^2)$. Also, $x_1^2 = (1-x_n)^2 = 1/(2n)^2$, so $x_1^2 + (1-x_n)^2 + \sum_{i=1}^{n-1} AL(x_i, x_{i+1}) = n/(2n^2) \le 2\epsilon$. Therefore, $\Theta(\epsilon^{-1})$ samples are always sufficient (and if, for instance, $f$ is a constant, necessary).

We now give a deterministic adaptive algorithm.

**Algorithm** LIPSCHITZ-INTEGRATE

**1.** Sample $f(0)$ and $f(1)$.
**2.** Do while the total area-looseness of unsampled intervals is greater than $2\epsilon$:
  **3.** Find the interval $(x, y)$ between two adjacent sampled points that maximizes $AL(x, y)$.
  **4.** Sample $f((x + y)/2)$.
**5.** Output the integral of the linear interpolation through the samples.

In terms of implementation, the algorithm maintains the total area-looseness of the current unsampled intervals, the unsampled intervals themselves in a linked list, and uses a priority queue to choose the unsampled interval with the largest area-looseness at every step and sample in the middle of it.

The correctness of the algorithm is clear from Proposition 7: the algorithm stops precisely when the total area-looseness of the unsampled intervals is no more than $2\epsilon$. We need to analyze the algorithm's performance.

**Theorem 1** *Algorithm* LIPSCHITZ-INTEGRATE *makes $O(\mathrm{DOPT} \cdot \log(\epsilon^{-1}/\mathrm{DOPT}))$ samples on problem instance $(f, \epsilon)$.*

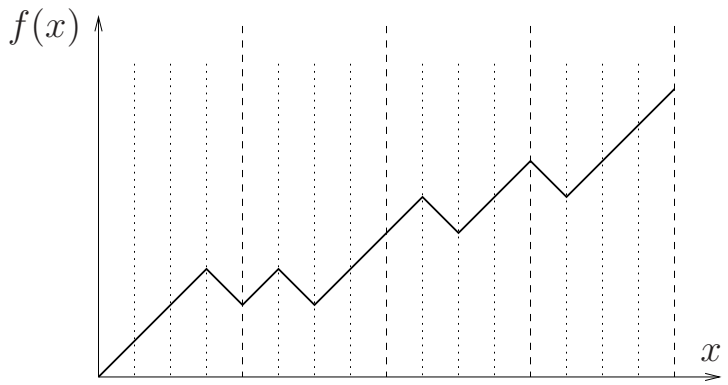


Figure 4: Lower bound construction for deterministic algorithms with $n = 16$ and $k = 4$.

**Proof:** We will actually compare the number of samples to $\mathrm{DOPT}(f, \epsilon/2)$ rather than to $\mathrm{DOPT}(f, \epsilon)$. We can do this because if we take a proof set for $\mathrm{DOPT}(f, \epsilon)$ and sample in the middle of every unsampled interval, then by Proposition 1 (4), we will obtain a proof set for $\mathrm{DOPT}(f, \epsilon/2)$. Thus, $\mathrm{DOPT}(f, \epsilon/2) \leq 2 \cdot \mathrm{DOPT}(f, \epsilon) + 1$. So let $P$ be a proof set for $(f, \epsilon/2)$ of size $\mathrm{DOPT}(f, \epsilon/2)$.

First, we argue that no interval of length smaller than $4\epsilon$ is ever subdivided. Suppose for contradiction that among $n$ intervals $I_1, \ldots, I_n$ of lengths $a_1, \ldots, a_n$, interval $I_k$ with $a_k < 4\epsilon$ is chosen for subdivision. By Proposition 1 (1), $AL(I_i) \leq a_i^2/2$, so $\sqrt{AL(I_k)} \leq 2\epsilon$. On the other hand, $\sum a_i = 1$, so $\sum \sqrt{AL(I_i)} \leq 1$. Multiplying the inequalities, we get $\sum AL(I_i) \leq \sum \sqrt{AL(I_i)AL(I_k)} \leq 2\epsilon$. But this implies that the algorithm should have terminated, which is a contradiction.

Now, we count the number of samples relative to $P$. The number of splits is $O(|P|)$ by Proposition 4. The above paragraph shows that we can use Proposition 6 to conclude that there are $O(|P| \log(\epsilon^{-1}/|P|))$ squeezes. We now show that there are $O(|P|)$ fizzles and so prove the theorem.

A fizzle occurs when an interval not containing a point of $P$ is chosen for subdivision. Consider the situation after $n$ points have been sampled. Let the sampled points be $0 = x_1 \leq x_2 \leq \cdots \leq x_n = 1$. Because the total area-looseness of intervals between points of $P$ is at most $\epsilon$, by repeated application of Proposition 1 (2,3), we have $\sum_{[x_i, x_{i+1}) \cap P = \emptyset} AL(x_i, x_{i+1}) \leq \epsilon$. The algorithm has not terminated, so the total area-looseness must be more than $2\epsilon$, which implies that $\sum_{[x_i, x_{i+1}) \cap P \neq \emptyset} AL(x_i, x_{i+1}) > \epsilon$. Because there are at most $|P|$ elements in the sum on the left hand side, the largest element must be greater than $\epsilon/|P|$. Therefore, there exists a $k$ such that $[x_k, x_{k+1})$ contains a point of $P$ and $AL(x_k, x_{k+1}) > \epsilon/|P|$. So if a fizzle occurs, the area-looseness of the chosen interval must be at least $\epsilon/|P|$.

Now let $S_t$ be the set of samples made by the algorithm after time $t$. Define $A_t$ as follows: let $\{y_1, y_2, \ldots, y_n\} = S_t \cup P$ with $0 = y_1 \leq y_2 \leq \cdots \leq y_n$ and let $A_t = \sum_{i=1}^{n-1} AL(y_i, y_{i+1})$. Clearly, $A_t \geq 0$, $A_t \geq A_{t+1}$ (by Proposition 1 (3)), and therefore, $A_t \leq A_0 \leq 2\epsilon$. Every fizzle splits an interval between adjacent $y$'s into two. Because the area-looseness of the interval before the split was at least $\epsilon/|P|$, by Proposition 1 (4), $A_t$ decreases by at least $\epsilon/(2|P|)$ as a result of every fizzle. Therefore, there can be at most $4|P|$ fizzles during an execution. □

We prove a matching lower bound, showing that the logarithmic factor is necessary and that LIPSCHITZ-INTEGRATE is optimally adaptive:

**Theorem 2** *For any deterministic algorithm and for any $\epsilon > 0$ and any integer $k$ such that $0 < k < \epsilon^{-1}/2$, there exists a problem instance $(f, \epsilon)$ of* LIPSCHITZ-INTEGRATION *with $\mathrm{DOPT}(f, \epsilon) = O(k)$ on which that algorithm performs $\Omega(k \log(\epsilon^{-1}/k))$ samples.*

**Proof:** Let $n = k\lfloor 1/\sqrt{2\epsilon k} \rfloor$. Divide the parameter space $[0, 1]$ into $n$ equal regions and group them into $k$ groups of $n/k$ regions each. In each group, let $n/k - 1$ regions have slope 1 and let 1 region have slope $-1$ (see Figure 4 for an illustration). Sampling at 0, 1, and every point where the slope of $f$ changes is a

proof set because the area looseness of all unsampled intervals is 0. This implies that for any such function, DOPT $\leq 2k + 2 = O(k)$.

Now we show that any algorithm will need additional samples on some function of this type. Consider a group, and consider an interval $I$ consisting of the region on which $f$ has negative slope in that group and an adjacent region in that group. Because the value of $f$ is the same on the endpoints of $I$, $AL(I) = 2/n^2 \geq 4\epsilon/k$. Therefore, an algorithm that does not find the negative-slope region or an adjacent one in at least $k/2$ groups will not be correct on some inputs. Finding the negative-slope region in each group is a binary search that needs $\Omega(\log(n/k)) = \Omega(\log(\epsilon^{-1}/k))$ samples and this needs to be done independently for $\Omega(k)$ groups, giving us the bound. □

# 5 Algorithm LIPSCHITZ-MC-INTEGRATE

A standard strategy in a Monte Carlo integration algorithm is to sample at a point picked uniformly at random from an interval. The expected value of such a sample, scaled by the length of the interval, is precisely the value of the integral over the interval, so the goal is to minimize the variance. When the function is Lipschitz, the variance of the integral estimate based on such a sample can be as high as a constant times the fourth power of the length of the interval. However, if we use the fact that when the area looseness of an interval is low, we approximately know the function, we can adjust the sample to get an unbiased estimator of the integral over that interval whose variance is the square of the area looseness in the worst case. Procedure MC-SAMPLE shows how to do this.

**Procedure** MC-SAMPLE$(x_1, x_2)$:

**1.** Let $x$ be a random number, uniformly chosen from $[x_1, x_2]$
**2.** If $f(x_1) \leq f(x_2)$, then SAMPLE $\leftarrow \left(f(x) - x + \frac{x_1+x_2}{2}\right)$
**3.** Else SAMPLE $\leftarrow \left(f(x) + x - \frac{x_1+x_2}{2}\right)$
**4.** Return SAMPLE $\cdot\, (x_2 - x_1)$

**Proposition 8** MC-SAMPLE$(x_1, x_2)$ *returns an unbiased estimator of* $\int_{x_1}^{x_2} f(x)\,dx$ *that has variance at most* $AL^2(x_1, x_2)$.

**Proof:** Because $x$ is uniform, $E[x - (x_1 + x_2)/2] = 0$ so $E[\text{SAMPLE}] = E[f(x)]$. Also, because $x$ is uniform, the definition of expectation implies that $E[f(x)] = \int_{x_1}^{x_2} f(x)/(x_2 - x_1)\,dx$, so the estimator is unbiased.

We bound the variance when $f(x_1) \leq f(x_2)$. The other case is symmetric. The variance of the return value is equal to $\text{var}(\text{SAMPLE}) \cdot (x_2 - x_1)^2 = \text{var}(f(x) - x) \cdot (x_2 - x_1)^2$. Because $f$ is Lipschitz, $f(x_2) - (x_2 - x) \leq f(x) \leq f(x_1) + (x - x_1)$, so $f(x_2) - x_2 \leq f(x) - x \leq f(x_1) - x_1$. If a random variable is always within an interval, its variance can be at most one quarter of the square of its length. Therefore, $\text{var}(f(x) - x) \leq (x_2 - x_1 + f(x_1) - f(x_2))^2/4 = L(x_1, x_2)^2/4$. By Proposition 2 (2), $L(x_1, x_2)^2(x_2 - x_1)^2/4 \leq AL^2(x_1, x_2)$. □

In order to compute the integral over $[0, 1]$, we would like an estimator for that integral with low variance. If we split $[0, 1]$ into intervals whose total $AL^2$ is small and run MC-SAMPLE on each interval, we will get such an estimator, as shown in the following corollary.

**Corollary 1** *Let* $0 = x_1 < x_2 < \cdots < x_n = 1$ *and suppose* $\sum_{i=1}^{n-1} AL^2(x_i, x_{i+1}) \leq \epsilon^2/3$. *Let* $\hat{I} = \sum_{i=1}^{n-1} \text{MC-SAMPLE}(x_i, x_{i+1})$. *Let* $I = \int_0^1 f(x)\,dx$. *Then* $\Pr[|\hat{I} - I| \geq \epsilon] \leq 1/3$.

**Proof:** By Proposition 8, $\hat{I}$ is an unbiased estimator of $I$ with variance at most $\sum_{i=1}^{n-1} AL^2(x_i, x_{i+1})$, which is at most $\epsilon^2/3$. From Chebyshev's inequality, we obtain that $\Pr[|\hat{I} - I| > \sqrt{3}\sqrt{\epsilon^2/3}] \leq 1/3$, as necessary. □

The remaining difficulty is to find a small number of intervals whose total $AL^2$ is smaller than $\epsilon^2/3$. Note that the deterministic adaptive algorithm in Section 4 finds a small number of intervals whose total $AL$ is smaller than $\epsilon$. We show that we can use the same idea here. Thus, to obtain a randomized adaptive algorithm, we use a deterministic adaptive algorithm to get a rough idea of the function and then use Monte Carlo sampling with variance reduction (MC-SAMPLE) to improve our estimate of the integral.

**Algorithm** LIPSCHITZ-MC-INTEGRATE:

**1.** Sample $f(0)$ and $f(1)$.
**2.** Do while the total $AL^2$ of unsampled intervals is greater than $\epsilon^2/3$:
  **3.** Find the interval $(x, y)$ between two adjacent sampled points that maximizes $AL^2(x, y)$.
  **4.** Sample $f((x + y)/2)$.
**5.** Run MC-SAMPLE on each interval between adjacent sampled points and output the sum.

Correctness is guaranteed by Corollary 1 because the algorithm exits the loop in lines 2–4 only when the total $AL^2$ of intervals between sampled points is no more than $\epsilon^2/3$.

# 6 Performance Analysis

**Theorem 3** *On problem instance* $(f, \epsilon)$ *algorithm* LIPSCHITZ-MC-INTEGRATE *performs* $O(\mathrm{ROPT}^{4/3}(f, \epsilon) + \mathrm{ROPT}(f, \epsilon) \log(1/\epsilon))$ *samples.*

We prove this theorem with three lemmas. The first gives a lower bound on ROPT in terms of a set of points. The second two give an upper bound on the runtime of the algorithm in terms of this set of points.

For the analysis of the algorithm, let $f$ be the Lipschitz function input to LIPSCHITZ-MC-INTEGRATE.

**Lemma 1** *Given* $f$*, there exists a set of points* $0 = x_1 < x_2 < \cdots < x_n = 1$ *such that for* $1 \le i \le n - 2$*,* $AL(x_i, x_{i+1}) = 3\epsilon$*, and* $AL(x_{n-1}, x_n) \le 3\epsilon$*. Furthermore,* $\mathrm{ROPT}(f, \epsilon) \ge (n - 2)/3$.

**Proof:** We begin by constructing a set of points that satisfies the conditions. Obviously, $x_1$ should be 0. Suppose we have constructed the first $k$ points and $x_k \neq 1$. If $AL(x_k, 1) \le 3\epsilon$, set $x_{k+1} = 1$ and we are done. Otherwise, notice that $f$ is continuous, so $AL$ is also continuous. By Proposition 1 (1), $AL(x_k, x_k) = 0$. Therefore, by the intermediate value theorem, there is an $x \in [x_k, 1]$ such that $AL(x_k, x) = 3\epsilon$ and we set $x_{k+1}$ to be that $x$.

Consider an algorithm $A$ that is correct with probability at least 2/3 on all inputs and consider its executions on $f$. Let $e_i$ for $1 \le i \le n - 2$ be the expected number of samples $A$ performs in $(x_i, x_{i+1})$. We claim that in order for $A$ to be correct, it must have $e_i \ge 1/3$ for all $i$ and therefore, the total expected number of samples is $\sum_{i=1}^{n-2} e_i \ge (n - 2)/3$.

Suppose for contradiction, that $e_i < 1/3$ for some $i$. Then, by Markov's inequality, the probability that $A$ samples in $(x_i, x_{i+1})$ is less than 1/3. Now consider two functions defined as follows: $\hat{f}_1(x) = \hat{f}_2(x) = f(x)$ everywhere except $(x_i, x_{i+1})$ and $\hat{f}_1(x) = LO_{x_i}^{x_{i+1}}(x)$ and $\hat{f}_2(x) = HI_{x_i}^{x_{i+1}}(x)$ on $(x_i, x_{i+1})$. By Proposition 2 (3), $\int_0^1 \hat{f}_2(x)dx - \int_0^1 \hat{f}_1(x) = AL(x_i, x_{i+1}) = 3\epsilon$, so no output is correct for both $\hat{f}_1$ and $\hat{f}_2$. Suppose, that we feed $\hat{f}_1$ and $\hat{f}_2$ with probability 1/2 each as input to $A$. Conditioned on $A$ not sampling in $(x_i, x_{i+1})$, the output of $A$ is independent of which function was input. Therefore, conditioned on $A$ not sampling in $(x_i, x_{i+1})$, the probability of error is at least 1/2. Because $\hat{f}_1 = \hat{f}_2 = f$ not on $(x_i, x_{i+1})$, the probability of $A$ not sampling on $(x_i, x_{i+1})$ is greater than 2/3, so the probability of error is greater than 1/3, which implies that $A$ is invalid. □

Because the number of samples in step 5 is smaller (by 1) than the number of samples in steps 1–4, we only focus on the samples in steps 1–4. For the analysis, we split the execution of the algorithm into two phases. The algorithm is in Phase 1 while there is a pair of adjacent sampled points $x_i$ and $x_{i+1}$ for which $AL(x_i, x_{i+1}) > 3\epsilon$. When all pairs of adjacent samples have $AL$ at most $3\epsilon$, the algorithm is in Phase 2. Note that by Proposition 1 (2), area looseness between adjacent samples never increases as the algorithm executes, so once it enters Phase 2, it never goes back to Phase 1. We now bound the number of samples made in steps 1–4 in the phases.

**Lemma 2** *In Phase 1,* LIPSCHITZ-MC-INTEGRATE *makes* $O(\mathrm{ROPT}(f, \epsilon) \log(1/\epsilon))$ *samples on problem instance* $(f, \epsilon)$.

**Proof:** Let $X$ be the set of $x_i$'s constructed as in Lemma 1. We count the samples made by LIPSCHITZ-MC-INTEGRATE relative to $X$. By Proposition 4, there are at most $O(|X|)$ splits. We now need a lower bound on the size of intervals in Phase 1 to count the number of squeezes. We note that an interval whose length is smaller than $\sqrt{6\epsilon}$ has area looseness at most $3\epsilon$ (by Proposition 1 (1)) and will therefore never be chosen for subdivision in Phase 1. Therefore, in Phase 1, every interval has length at least $\sqrt{6\epsilon}/2$. So by Proposition 6, there are at most $|X|\log((\sqrt{6\epsilon}/2)^{-1}/|X|) = O(|X|\log(1/\epsilon))$ squeezes. There are no fizzles because any interval whose area looseness is greater than $3\epsilon$ must have a point of $X$ (by Proposition 1 (2) and by construction of $X$). By Lemma 1, $|X| = O(\mathrm{ROPT}(f,\epsilon))$, so we have the claimed bound. □

**Lemma 3** *In Phase 2,* LIPSCHITZ-MC-INTEGRATE *uses at most* $O(\mathrm{ROPT}(f,\epsilon)^{4/3} + \mathrm{ROPT}(f,\epsilon)\log(1/\epsilon))$ *samples on problem instance* $(f,\epsilon)$.

**Proof:** After Phase 1 is complete, the area looseness between adjacent sampled points is at most $3\epsilon$. Let $0 = y_1 < y_2 < \cdots < y_m = 1$ be the smallest subset of sampled points (including 0 and 1) such that $AL(y_i, y_{i+1}) \le 3\epsilon$ for all $y$. We claim that $m \le 6 \cdot \mathrm{ROPT}(f,\epsilon)$. Consider the set of $x_i$'s constructed as in Lemma 1. If $y_i$'s are a minimal set of points with area looseness no greater than $3\epsilon$ between adjacent ones, then every interval of the form $[x_i, x_{i+1}]$ has at most two $y_i$'s (if there are three, the middle one is unnecessary). Therefore there are at most twice as many $y_i$'s as $x_i$'s.

Now assume the algorithm makes more samples in Phase 2 than in Phase 1 because otherwise, it makes $O(\mathrm{ROPT}(f,\epsilon)\log(1/\epsilon))$ samples and we are done. We apply Propostion 9 to prove this lemma. Let $Y$ be the set of $y_i$'s, let $Z^{(0)}$ be the set of sampled points at the end of Phase 1 and let $t_0 = 550 \cdot \mathrm{ROPT}^{4/3}$. We have $A = \sum_{i=1}^{m-1} AL(y_i, y_{i+1}) \le 18 \cdot \mathrm{ROPT} \cdot \epsilon$. By Proposition 9, after $t_0$ samples, the total $AL^2$ will be at most $\frac{4608 \cdot (6 \cdot \mathrm{ROPT})^2 \cdot (18 \cdot \mathrm{ROPT})^2 \epsilon^2}{550^3 \mathrm{ROPT}^4} \le \epsilon^2/3$ so the algorithm will stop after $t_0$ steps. □

The following proposition shows that as our algorithm samples, the total squared area looseness declines as the cube of the number of samples. We prove it by associating a number with each interval that is an upper bound on its area looseness. We then show that these numbers are within a factor of four of each other and use this to show that that the sum of their squares decreases as the cube of the number of samples.

**Proposition 9** *Let* $Y = \{y_1, \ldots, y_m\}$ *with* $0 = y_1 < \cdots < y_m = 1$, *and let* $A = \sum_{i=1}^{m-1} AL(y_i, y_{i+1})$. *Consider the sequence* $Z^{(0)}, Z^{(1)}, Z^{(2)}, \ldots$ *of sets of samples where* $Z^{(0)} \supseteq Y$ *is an arbitrary superset of* $Y$ *and, for each* $t \ge 1$, $Z^{(t)} = Z^{(t-1)} \cup \{z^{(t)}\}$ *where* $z^{(t)}$ *is the midpoint* $(x^{(t)} + y^{(t)})/2$ *of the interval* $(x^{(t)}, y^{(t)})$ *of* $Z^{(t-1)}$ *with the largest area looseness* $AL(x^{(t)}, y^{(t)})$. *Then, for any* $t_0 \ge |Z_0|$, $\sum_{(x,y)\in\mathcal{I}(Z^{(t)})} AL^2(x,y) \le (4608 m^2 A)/t_0^3$.

**Proof:** In this proposition if $X = \{x_1, \ldots, x_n\}$ is a set of real numbers with $x_1 < \cdots < x_n$, let $\mathcal{I}(X) = \{(x_i, x_{i+1}) \mid 1 \le i < n\}$.

First we define a number $q^{(t)}(x,y)$ associated with each interval $(x,y)$ of $Z_t$. Define $q^{(0)}(x,y) = L(x,y) \cdot (y-x)$. For $t > 0$, let $Q^{(t)} = L(x^{(t)}, y^{(t)}) \cdot (y^{(t)} - x^{(t)})$. Define $q^{(t)}(x^{(t)}, z^{(t)}) = q^{(t)}(z^{(t)}, y^{(t)}) = Q^{(t)}/2$, and define $q^{(t)}(x,y) = \min\{q^{(t-1)}(x,y), 2Q^{(t)}\}$ for all other intervals $(x,y)$ (those without $z^{(t)}$ as an endpoint).

We claim that $q^{(t)}(x,y) \ge L(x,y) \cdot (y-x)$. For $t = 0$, this property holds with equality. For $t > 0$, $q^{(t)}(x^{(t)}, z^{(t)}) = Q^{(t)}/2 = L(x^{(t)}, y^{(t)}) \cdot (y^{(t)} - x^{(t)})/2$. The claim follows for $(x^{(t)}, z^{(t)})$ because $L(x^{(t)}, y^{(t)}) \ge L(x^{(t)}, z^{(t)})$, by Proposition 3 (2), and because $(y^{(t)} - x^{(t)})/2 = z^{(t)} - x^{(t)}$. The claim follows symmetrically for $(z^{(t)}, y^{(t)})$. For all other intervals $(x,y)$ of $Z^{(t)}$, $t > 0$, we know by induction on $t$ that $q^{(t-1)}(x,y) \ge L(x,y) \cdot (y-x)$, so it remains only to show that $2Q^{(t)} \ge L(x,y) \cdot (y-x)$. By Proposition 2 (2), $2Q^{(t)} \ge 2AL(x^{(t)}, y^{(t)})$. Because $(x^{(t)}, y^{(t)})$ has the maximum area looseness among intervals of $Z^{(t-1)}$, $2AL(x^{(t)}, y^{(t)}) \ge 2AL(x,y)$. By Proposition 2 (2), $2AL(x,y) \ge L(x,y) \cdot (y-x)$. Thus $2Q^{(t)} \ge L(x,y) \cdot (y-x)$ and the claim follows.

Let $I^{(t)} = \mathcal{I}(Z^{(t)}) \setminus \mathcal{I}(Z^{(0)})$ be the set of intervals of $Z^{(t)}$ that are the result of subdivision. Notice that for $t > 0$, $t < |I^{(t)}| \le 2t$. We claim that $\max_{(x,y)\in\mathcal{I}(Z^{(t)})} q^{(t)}(x,y) \le 4 \min_{(x,y)\in I^{(t)}} q^{(t)}(x,y)$, for any $t > 0$. (For $t = 0$, $I^{(t)}$ is empty, so the claim is meaningless.) We use the argument above that $2Q^{(t)} \ge L(x,y) \cdot (y-x)$ for any interval $(x,y)$ of $Z^{(t)}$, and thus $2Q^{(t)} \ge \max_{(x,y)\in\mathcal{I}(Z^{(t)})} q^{(t)}(x,y)$. For $t = 1$, $\max_{(x,y)\in\mathcal{I}(Z^{(1)})} q^{(1)}(x,y) \le 2Q^{(1)}$, and $\min_{(x,y)\in I^{(1)}} q^{(1)}(x,y)$ is the common value $Q^{(1)}/2$ assigned to the two intervals $(x^{(1)}, z^{(1)})$ and $(z^{(1)}, y^{(1)})$ resulting from the first subdivision. Thus the base case of $t = 1$

follows. For $t > 1$, let

$$M^+ = \max_{(x,y)\in\mathcal{I}(Z^{(t-1)})\setminus\{(x^{(t)},y^{(t)})\}} q^{(t-1)}(x,y)$$

and let

$$M^- = \min_{(x,y)\in I^{(t-1)}\setminus\{(x^{(t)},y^{(t)})\}} q^{(t-1)}(x,y).$$

By the inductive hypothesis on $t$, $\max_{(x,y)\in\mathcal{I}(Z^{(t-1)})} q^{(t-1)}(x,y) \le 4\min_{(x,y)\in I^{(t-1)}} q^{(t-1)}(x,y)$, so by dropping a term each from each side, $M^+ \le 4M^-$. By construction of the $q^{(t)}$'s, we have $\min_{(x,y)\in I^{(t)}} q^{(t)}(x,y) = \min\{M^-, Q^{(t)}/2\}$ and $\max_{(x,y)\in\mathcal{I}(Z^{(t)})} q^{(t)}(x,y) = \max\{\min\{M^+, 2Q^{(t)}\}, Q^{(t)}/2\}$. As argued above, $Q^{(t)} \le q^{(t-1)}(x^{(t)},y^{(t)})$, and by dropping terms from the induction hypothesis, $q^{(t-1)}(x^{(t)},y^{(t)}) \le 4M^-$. From this, we conclude that $\min\{4M^-, 2Q^{(t)}\} = \max\{\min\{4M^-, 2Q^{(t)}\}, Q^{(t)}\} \ge \max\{\min\{M^+, 2Q^{(t)}\}Q^{(t)}/2\}$ and the claim follows.

The intervals in $\mathcal{I}(Z^{(0)}) \cup \mathcal{I}(Z^{(1)}) \cup \cdots \cup \mathcal{I}(Z^{(t_0)})$ (removing duplicate occurrences of intervals) form a natural structure of rooted binary trees. For each $1 \le t \le t_0$, define the interval $(x^{(t)}, y^{(t)})$ to be the *parent* of intervals $(x^{(t)}, z^{(t)})$ and $(z^{(t)}, y(t))$. This definitions yields a unique parent interval for every interval in $I^{(1)} \cup \cdots \cup I^{(t_0)}$. The parent of such an interval may not be in $I^{(1)} \cup \cdots \cup I^{(t_0)}$, but it is in $\mathcal{I}(Z^{(0)}) \cup \mathcal{I}(Z^{(1)}) \cup \cdots \cup \mathcal{I}(Z^{(t_0)})$. The intervals without parents are thus precisely the intervals in $Z^{(0)}$, which we define as *roots*. This parent and root structure defines a forest of rooted binary trees on intervals in $\mathcal{I}(Z^{(0)}) \cup \mathcal{I}(Z^{(1)}) \cup \cdots \cup \mathcal{I}(Z^{(t_0)})$. Every interval is a subinterval of its parent and has half the length. The leaves of the trees correspond precisely to intervals of $Z^{(t_0)}$.

We now obtain an upper bound on the $q^{(t)}$'s. Let $I_j^{(t_0)} = \{(x,y) \in I^{(t_0)} \mid y_j \le x < y_{j+1}\}$. We claim that, for all $1 \le j < m$,

$$\sum_{(x,y)\in I_j^{(t_0)}} q^{(t_0)}(x,y)/(y-x) \le 6L(y_j, y_{j+1}).$$

The intervals in $I_j^{(t_0)}$ are precisely the leaves of the trees of nonzero height rooted at intervals of $Z^{(0)}$ that are subintervals of $(y_j, y_{j+1})$. By Proposition 3 (3), it suffices to prove the claim separately for each such tree: for each tree $T$ in the forest, with root interval $(r(T), s(T))$ of $Z^{(0)}$ and with leaves $\Lambda(T)$,

$$\sum_{(x,y)\in\Lambda(T)} q^{(t_0)}(x,y)/(y-x) \le 6L(r(T), s(T)).$$

For the proof, we make the following stronger claim about the subtree $T(r,s)$ rooted at any nonleaf node $(r,s)$:

$$\sum_{(x,y)\in\Lambda(T(r,s))} q^{(t_0)}(x,y)/(y-x) \le (6 - 2^{3-h(T(r,s))})L(r,s).$$

Let $(a,b)$ and $(c,d)$ be the left and right children of $(r,s)$, respectively. If both $(a,b)$ and $(c,d)$ are leaves, then $h(T) = 1$ and $q^{(t_0)}(a,b) = q^{(t_0)}(c,d) = L(r,s)\cdot(s-r)/2$, so $q^{(t_0)}(a,b)/(b-a) + q^{(t_0)}(c,d)/(d-c) = 2L(r,s)$ as desired. If neither $(a,b)$ nor $(c,d)$ are leaves, then we can break the sum into two pieces and apply induction on height to each piece:

$$\begin{aligned}
\sum_{(x,y)\in\Lambda(T(r,s))} \frac{q^{(t_0)}(x,y)}{y-x} &= \sum_{(x,y)\in\Lambda(T(a,b))} \frac{q^{(t_0)}(x,y)}{y-x} + \sum_{(x,y)\in\Lambda(T(c,d))} \frac{q^{(t_0)}(x,y)}{y-x} \\
&\le (6 - 2^{3-h(T(a,b))})L(a,b) + (6 - 2^{3-h(T(c,d))})L(c,d) \\
&\le (6 - 2^{3-h(T(r,s))})(L(a,b) + L(c,d)),
\end{aligned}$$

which by Proposition 3 (3) is at most $(6 - 2^{3-h(T(r,s))})L(r,s)$ as desired. If exactly one of $(a,b)$ and $(c,d)$ is a leaf, then we relabel so that $(a,b)$ is the leaf. Let $(e,f)$ be a leaf of $T(c,d)$ whose distance in the tree from $(c,d)$ is $h(T(c,d)) = h(T(r,s)) - 1$. so that $f - e = (d-c)/2^{h(T(r,s))-1} = (b-a)/2^{h(T(r,s))-1}$. Because $\max_{(x,y)\in\mathcal{I}(Z^{(t_0)})} q^{(t_0)}(x,y) \le 4\min_{(x,y)\in I^{(t_0)}} q^{(t_0)}(x,y)$, we have $q^{(t_0)}(a,b) \le 2^2 q^{(t_0)}(e,f)$. Thus $q^{(t_0)}(a,b)/(b-a) \le q^{(t_0)}(e,f)\cdot 2^{3-h(T(r,s))}/(f-e)$. At the time $t$ that interval $(e,f)$ was created from its

parent $(g,h)$, $q^{(t)}(e,f) = L(g,h)\cdot(f-e)$, which by Proposition 3 (2) is at most $L(c,d)\cdot(f-e)$. As time $t$ progresses, $q^{(t)}(e,f)$ only decreases. Thus $q^{(t_0)}(e,f)2^{3-h(T(r,s))}/(f-e) \le L(c,d)2^{3-h(T(r,s))}$. By induction on height,

$$\begin{aligned}
\sum_{(x,y)\in\Lambda(T(r,s))} \frac{q^{(t_0)}(x,y)}{y-x} &= \frac{q^{(t_0)}(a,b)}{b-a} + \sum_{(x,y)\in\Lambda(T(c,d))} \frac{q^{(t_0)}(x,y)}{y-x} \\
&\le L(c,d)2^{3-h(T(r,s))} + (6-2^{3-h(T(c,d))})L(c,d) \\
&= (6+2^{3-h(T(r,s))} - 2^{4-h(T(r,s))})L(c,d) \\
&= (6-2^{3-h(T(r,s))})L(c,d),
\end{aligned}$$

which by Proposition 3 (2) is at most $(6-2^{3-h(T(r,s))})L(r,s)$ as desired. This completes the proof of the claim.

Finally consider $Z^{(t_0)}$. Let $M^+ = \max_{(x,y)\in\mathcal{I}(Z^{(t_0)})} q^{(t_0)}(x,y)$. For any interval $(x,y)\in I^{(t_0)}$, $M^+ \le 4q^{(t_0)}(x,y)$, so for any $j$, $\sum_{(x,y)\in I_j^{(t_0)}} M^+/(y-x) \le 4\sum_{(x,y)\in I_j^{(t_0)}} q^{(t_0)}(x,y)/(y-x)$, which by the previous claim is at most $24L(y_j,y_{j+1})$. Because leaf intervals in a tree partition the root interval, $\sum_{(x,y)\in I_j^{(t_0)}}(y-x) \le y_{j+1}-y_j$. Multiplying these two inequalities,

$$\begin{aligned}
24L(y_j,y_{j+1})\cdot(y_{j+1}-y_j) &\ge \sum_{(x,y)\in I_j^{(t_0)}} \sum_{(x',y')\in I_j^{(t_0)}} \frac{M^+}{y-x}(y'-x') \\
&= \sum_{(x,y)\in I_j^{(t_0)}} M^+ + \sum_{(x,y)\in I_j^{(t_0)}} \sum_{\substack{(x',y')\in I_j^{(t_0)} \\ x<x'}} \left( M^+\frac{y-x}{y'-x'} + M^+\frac{y'-x'}{y-x} \right) \\
&= |I_j^{(t_0)}|M^+ + \sum_{(x,y)\in I_j^{(t_0)}} \sum_{\substack{(x',y')\in I_j^{(t_0)} \\ x<x'}} M^+ \underbrace{\left( \frac{y-x}{y'-x'} + \frac{y'-x'}{y-x} \right)}_{=\ \alpha+1/\alpha\ \ge\ 2} \\
&\ge |I_j^{(t_0)}|M^+ + 2\binom{|I_j^{(t_0)}|}{2} M^+ \\
&= |I_j^{(t_0)}|^2 M^+.
\end{aligned}$$

Summing over $j$,

$$\sum_{j=1}^{m-1} 24L(y_j,y_{j+1})\cdot(y_{j+1}-y_j) \ge M^+ \sum_{j=1}^{m-1} |I_j^{(t_0)}|^2 \ge M^+ \frac{|I^{(t_0)}|^2}{m} \ge M^+ \frac{t_0|I^{(t_0)}|}{m}.$$

Now $|\mathcal{I}(Z^{(t_0)})| = |\mathcal{I}(Z^{(0)})| + t_0 = |Z^{(0)}| + t_0 - 1$. Because $t_0 \ge |Z^{(0)}| - 2$, $2|I^{(t_0)}| \ge |Z^{(0)}| + t_0 - 1$. Thus

$$\begin{aligned}
\sum_{(x,y)\in\mathcal{I}(Z^{(t_0)})} \left(q^{(t_0)}(x,y)\right)^2 &\le (|Z^{(0)}| + t_0 - 1)(M^+)^2 \\
&\le 2|I^{(t_0)}|(M^+)^2 \\
&\le 2|I^{(t_0)}| \left( \frac{m}{t_0|I^{(t_0)}|} \sum_{j=1}^{m-1} 24L(y_j,y_{j+1})\cdot(y_{j+1}-y_j) \right)^2 \\
&= \frac{1152m^2}{t_0^2|I^{(t_0)}|} \left( \sum_{j=1}^{m-1} L(y_j,y_{j+1})\cdot(y_{j+1}-y_j) \right)^2,
\end{aligned}$$

which by Proposition 2 (2) is at most $(4608m^2A)/(t_0^2|I^{(t_0)}|)$. Recall that $q^{(t_0)}(x,y) \ge L(x,y)\cdot(y-x)$, which

is at least $AL(x,y)$ by Proposition 2 (2). Therefore

$$\sum_{(x,y)\in\mathcal{I}(Z^{(t_0)})} AL^2(x,y) \le \sum_{(x,y)\in\mathcal{I}(Z^{(t_0)})} \left(q^{(t_0)}(x,y)\right)^2 \le \frac{4608m^2A}{t_0^2|I^{(t_0)}|}.$$

□

The upper bound follows immediately from the two lemmas we have shown.

# 7 Randomized Lower Bounds

We begin by using a proof very similar to that of Theorem 2 to show that the logarithmic factor over ROPT is necessary.

**Lemma 4** *For any algorithm and for any $\epsilon > 0$ and any integer $k$ such that $0 < k < \epsilon^{-1}/2$, there exists a problem instance $(f,\epsilon)$ of* LIPSCHITZ-INTEGRATION *with $\mathrm{ROPT}(f,\epsilon) = O(k)$ on which that algorithm performs $\Omega(k\log(\epsilon^{-1}/k^2))$ samples.*

**Proof:** Let $n = k\lfloor 1/(k\sqrt{\epsilon})\rfloor$. Divide the parameter space $[0,1]$ into $n$ equal regions and group them into $k$ groups of $n/k$ regions each. In each group, let $n/k-1$ regions have slope 1 and let 1 region have slope $-1$ (refer back to Figure 4 for an illustration). The parameters 0, 1, and at every point where the slope of $f$ changes are a proof set because the area looseness of all unsampled intervals is 0. This implies that for any such function, $\mathrm{ROPT} \le \mathrm{DOPT} \le 2k+2 = O(k)$.

Now we show that any algorithm will need additional samples on some function of this type. Consider a group, and consider an interval $I$ consisting of the region on which $f$ has negative slope in that group and an adjacent region in that group. Because the value of $f$ is the same on the endpoints of $I$, $AL(I) = 2/n^2 \ge 2\epsilon$. If there is a problem instance on which an algorithm samples in some $I$ with probability less than $1/3$, the algorithm could be fed that problem instance and the problem instance that has the negative-slope region exchanged with the positive-slope region in $I$, with probability $1/2$ each. The algorithm would be able to distinguish between them with probability at most $1/3$ and the integrals of the functions differ by $2\epsilon$, so the algorithm will be wrong at least $1/3$ of the time. Therefore, any correct algorithm finds each $I$ with probability at least $1/3$. This is equivalent to a continuous binary search that requires $\Omega(\log(n/k)) = \Omega(\log(\epsilon^{-1}/k^2))$ samples. This search must be performed independently in $k/3$ groups in expectation, so the expected number of samples is at least $\Omega(k\log(\epsilon^{-1}/k^2))$. □

To prove that LIPSCHITZ-MC-INTEGRATE cannot be improved by more than a constant factor, we first show that Lemma 1 is actually a tight (to within a constant factor) lower bound on ROPT by proving the following upper bound.

**Lemma 5** *Given a Lipschitz function $f$, there is a set of points $0 = x_1 < x_2 < \cdots < x_k = 1$ such that for $1 \le i \le k-2$, $AL(x_i, x_{i+1}) = \epsilon/4$, and $AL(x_{k-1}, x_k) \le \epsilon/4$. Furthermore, $\mathrm{ROPT}(f,\epsilon) \le 2k-1$.*

**Proof:** The construction of the $x_i$'s is completely analogous to that performed in the proof of Lemma 1. To prove that $\mathrm{ROPT}(f,\epsilon) \le 2k-1$, we give an algorithm that solves LIPSCHITZ-INTEGRATION and uses $2k-1$ samples on $(f,\epsilon)$.

When the algorithm is given a function $g$, it samples at all $x_i$ and then at $k-1$ points picked uniformly at random, one from each interval between $x_i$ and $x_{i+1}$. If at every sample, $g(x) = f(x)$, the algorithm outputs $\int_0^1 f(x)dx$. Otherwise, it executes a slow deterministic algorithm that is always correct.

We first note that this algorithm uses $2k-1$ samples on $(f,\epsilon)$ because in that case, the deterministic algorithm is never executed. It is also clear that the algorithm is always correct in that case. The algorithm is also correct whenever the deterministic algorithm is executed. Thus, we need to prove that if $\left|\int_0^1 g(x)dx - \int_0^1 f(x)dx\right| > \epsilon$, then the probability of the deterministic algorithm being executed is at least $2/3$. We can immediately disregard $g$ for which $g(x_i) \ne f(x_i)$ for some $i$, as the deterministic algorithm is always executed in this case.

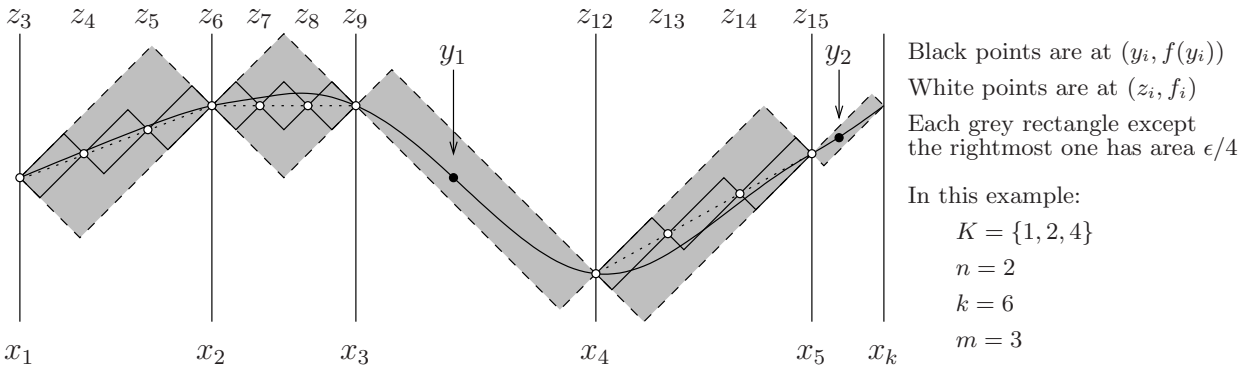


Figure 5: Figure for proof of Lemma 6. To illustrate the concepts more clearly, $m > n$ in the figure, although that is never really the case.

Consider an interval $(x_i, x_{i+1})$ and let $S_i = \{x \in (x_i, x_{i+1}) | g(x) \neq f(x)\}$. Let $p_i$ be $\frac{\mu(S)}{x_{i+1}-x_i}$, the Lebesgue measure of $S_i$ divided by the length of the associated interval. By Proposition 2 (1,2), for $x \in S_i$, $|g(x) - f(x)| \leq 2AL(x_i, x_{i+1})/(x_{i+1} - x_i) \leq \epsilon/2(x_{i+1} - x_i)$. Therefore, $\int_{x_i}^{x_{i+1}} g(x) - f(x)\, dx \leq \epsilon \cdot p_i/2$, so if $\left|\int_0^1 g(x)dx - \int_0^1 f(x)dx\right| > \epsilon$, then $\sum_{i=1}^{k-1} p_i > 2$. Now notice that $p_i$ is the probability that the sample in $(x_i, x_{i+1})$ is in $S$ (and if such an event occurs, the deterministic algorithm is executed). Thus, we need to show that $\prod_{i=1}^{k-1}(1 - p_i) \leq 1/3$. Because the sum $\sum_{i=1}^{k-1}(1 - p_i)$ is smaller than $k - 3$, the product is maximized when all of the $p_i$ are equal to $2/(k-1)$. In that case, the product is equal to $(1-2/(k-1))^{k-1}$. Because $1 + x \leq e^x$, $1 - 2/(k-1) \leq e^{-2/(k-1)}$, so $(1 - 2/(k-1))^{k-1} \leq e^{-2} \leq 1/3$. □

The above lemma implies that deterministic algorithms are not very powerful relative to ROPT. For instance, if $f(x) = 0$ for all $x$, $\text{ROPT}(f, \epsilon) = O(\epsilon^{-1/2})$ by Lemma 5, but DOPT is $\Theta(\epsilon^{-1})$. Therefore every deterministic algorithm requires $\Omega(\text{ROPT}^2)$ samples on some instances. Of course, as Lemma 4 (for small $k$) shows, it is easy to construct problem instance families on which deterministic algorithms are just as powerful as randomized ones.

We are now ready to prove the other part of the lower bound on the performance of randomized algorithms relative to ROPT. The lower bound we prove actually shows something stronger than simply that for any algorithm, there is a problem instance on which that algorithm needs $\Omega(\text{ROPT}^{4/3})$ samples. We show that given a function $f$ and a bunch of points $y_i$'s, we can construct a family of functions that are all similar to $f$ (actually equal to it on $y_i$'s) and have no larger ROPT's than $f$, but any algorithm uses $\Omega(\text{ROPT}^{4/3})$ samples on some member of that family. This shows that the 4/3 exponent is necessary for problem instances of all difficulties and global shapes.

The proof idea is that we build a function family with $[0, 1]$ divided into $\Theta(\text{ROPT}^{4/3})$ regions with two possible integral values differing by $\Theta(\epsilon/\text{ROPT}^{2/3})$ in each region. If an algorithm samples in fewer than a constant fraction of the regions, the other regions cause the integral to likely be off by $\Theta(\sqrt{\text{ROPT}^{4/3}}) \cdot \Theta(\epsilon/\text{ROPT}^{2/3}) = \Theta(\epsilon)$. To build these regions, we start with $\Theta(\text{ROPT})$ regions with area looseness of each equal to $\Theta(\epsilon)$, discard those that have $y_i$'s in them (because they must be fixed) and divide the remaining regions into $\Theta(\text{ROPT}^{1/3})$ parts each.

**Lemma 6** *Consider a problem instance $(f, \epsilon)$. Let $n = \lfloor \text{ROPT}(f, \epsilon)/4 \rfloor - 1$. For any set of $n$ parameter values $y_i$, there exists a function family $G$ such that:*

*1. For all $g \in G$, and for all $i$, $g(y_i) = f(y_i)$.*

*2. For all $g \in G$, $\text{ROPT}(g, \epsilon) = O(n)$.*

*3. For any correct algorithm, there is a $g \in G$ such that the algorithm performs $\Omega(n^{4/3})$ samples in expectation on $(g, \epsilon)$.*

**Proof:** See Figure 5. Consider the set of $k$ points $x_i$, defined as in Lemma 5. From the lemma, we have $k \geq 2n+2$. Let $K = \{i \in \{1,\ldots,k-2\} | \forall y_j : y_j \notin (x_i, x_{i+1})\}$ denote the set of indices of intervals between $x_i$'s that do not contain $y_j$'s. Because the number of $y_j$'s is at most $n$, $|K| \geq n$.

We now divide each interval whose index is in $K$ into $\Theta(n^{1/3})$ equal parts. The big-$O$ notation allows us to assume $n$ is at least 343. Let $m = \lfloor n^{1/3}/7 \rfloor$. Consider an interval $[x_i, x_{i+1}]$ for $i \in K$ and let $z_{im+j} = x_i + (x_{i+1} - x_i)j/m$ for $0 \leq j \leq m$. Also let $f_{im+j} = f(x_i) + (f(x_{i+1}) - f(x_i))j/m$. Note that if both $i$ and $i+1$ are in $K$, we have defined $z_{(i+1)m}$ and $f_{(i+1)m}$ twice, but the definitions are consistent.

Let $B = b_1, \ldots, b_{mk}$ be a sequence of variables where $b_i \in \{1, -1\}$. For each possible $B$ define a function:

$$g_B(x) = \begin{cases} HI_{z_i}^{z_{i+1}}(x), & \text{on interval } (z_i, z_{i+1}) \text{ if } b_i = 1 \\ LO_{z_i}^{z_{i+1}}(x), & \text{on interval } (z_i, z_{i+1}) \text{ if } b_i = -1 \\ f(x), & \text{if it is not between a consecutive pair of defined } z\text{'s} \end{cases}$$

where, for the purposes of $HI$ and $LO$, the value at $z_i$ is $f_i$, not $f(z_i)$. By Lemma 5, $\text{ROPT}(g_B, \epsilon) \leq 2k-1$. But by Lemma 1, we have $\text{ROPT}(f, \epsilon) \geq (p-2)/3$ where $p$ is the number of points necessary for area-looseness between adjacent ones to be smaller than $3\epsilon$. But we may apply Proposition 1(4) four times to obtain $16p$ intervals whose area-looseness is smaller than $\epsilon/4$. So $k < 16p$ and therefore, $\text{ROPT}(g_B, \epsilon) \leq 2k-1 \leq 32p = O(n)$.

If $i = am + j$, with $a \in K$ and $0 \leq j \leq m-1$, then $AL_{g_B}(z_i, z_{i+1}) = AL_{g_B}(z_{am}, z_{(a+1)m})/m^2 = \epsilon/4m^2$. So by Proposition 2 (3),

$$\int_0^1 g_B(x)\,dx = c + \frac{\epsilon}{8m^2} \sum_{i \in K, 0 \leq j \leq m-1} b_{im+j},$$

where $c$ does not depend on $B$.

Fix an algorithm $ALG$ that solves LIPSCHITZ-INTEGRATION, but use probability amplification to make it correct with probability at least 5/6. Let $t_B$ be the expected number of samples $ALG$ performs on $g_B$. Let $t = \max_B t_B$. By Markov's inequality, the probability that the algorithm performs more than $6t$ samples is no more than 1/6. So without loss of generality, we can assume that $ALG$ always uses no more than $6t$ samples and is correct with probability 2/3.

Let $S$ be the set of all possible executions of $ALG$ on all possible $g_B$'s. An "execution" consists of the parameters where $ALG$ samples in order, the values it receives, and its output. Every particular $g_B$ defines a probability distribution on $S$. Note that if $s \in S$ is an execution and $g_{B_1}$ and $g_{B_2}$ are two functions that are equal everywhere $s$ samples, then $\Pr[s|g_{B_1}] = \Pr[s|g_{B_2}]$.

Suppose that $B$ is picked uniformly at random. Consider any execution $s \in S$ and consider all $B$ such that $g_B$ is consistent with $s$. Bayes' Theorem states that $\Pr[g_B|s] = \Pr[g_B]\Pr[s|g_B]/\Pr[s]$. Note that the right hand side is the same for all $g_B$'s that are consistent with $s$.

Suppose for contradiction that $6t \leq nm/3$. Then there are at least $2nm/3$ intervals of the form $(z_i, z_{i+1})$ in which $s$ does not sample. Therefore, there are $q$ variables $b_i$ whose value does not influence whether $g_B$ is consistent with $s$, and $q \geq 2nm/3$. So the true integral value is $c + \frac{\epsilon}{8m^2} \cdot (2V - q)$ where $V$ is distributed as the sum of $q$ Bernoulli variables. Assume without loss of generality that $q$ is even (otherwise, look at $q-1$). The probability that $2V - q$ is exactly zero is $\binom{q}{q/2}/2^q$, which, by Stirling's approximation (see appendix), is smaller than $\sqrt{2/(\pi q)}$.

Because for all $a$, $\Pr[2V - q = a] \leq \Pr[2V - q = 0]$, we have $\Pr[|2V - q| < \sqrt{q}/3] < \sqrt{2/(\pi q)}\sqrt{q}/3 < 1/3$, so $\Pr[|2V - q| > \sqrt{q}/3] > 2/3$. Therefore, with probability at least 1/3, the value of the integral is at least $c + \frac{\epsilon}{8m^2}\sqrt{q}/3$ and with probability 1/3, it is at most $c - \frac{\epsilon}{8m^2}\sqrt{q}/3$. Because $q = 2n\lfloor n^{1/3}/7 \rfloor \geq 686\lfloor n^{1/3}/3 \rfloor^4 = 686m^4$, $\frac{\epsilon}{8m^2}\sqrt{q}/3 \geq \epsilon$ and with probability 2/3, the correct value is outside the range $(c-\epsilon, c+\epsilon)$. An output cannot be correct for values on both sides of the range, so the probability of failure given this execution is at least 1/3.

Therefore, if $B$ is chosen randomly, the probability of failure of any execution is at least 1/3 and therefore, there exists a $B$ for which the probability of failure is at least 1/3, contradicting the assumption that $ALG$ is a correct algorithm. Therefore, the expected number of samples $ALG$ makes is at least $nm/18 = \Omega(n^{4/3})$. □

Together, the two lemmas imply that the algorithm presented is optimally adaptive.

**Theorem 4** *Given an $\epsilon > 0$ and an integer $k$ such that $0 < k < \epsilon^{-1}/2$, there is a family of problem instances such that* $\mathrm{ROPT} = O(k)$ *on every member on the family, but any algorithm requires $\Omega(k^{4/3} + k\log(1/\epsilon))$ samples in expectation on some member of that family.*

**Proof:** If $k^{1/3} \geq \log(1/\epsilon)$, use the family constructed in Lemma 6. Otherwise, use the family constructed in Lemma 4, noting that $\log(\epsilon^{-1}/k^2) \leq \log(\epsilon^{-1}/\log^6(1/\epsilon)) = \Theta(\log(1/\epsilon))$. □

The nonadaptive method in [1] is to divide $[0,1]$ into $n$ equal subintervals and randomly sample in each. They prove that if the input function is Lipschitz with constant 1, this results in an error of at most $O(n^{-3/2})$. We now prove a simple corollary to Lemma 6 to show that this method is optimal in the worst case.

**Corollary 2** *Any algorithm requires $\Omega(\epsilon^{-2/3})$ samples on some problem instance.*

**Proof:** Consider $f(x) = 0$. Then $AL(x_1, x_2) = (x_2 - x_1)^2/2$. So if $AL(x_1, x_2) = 3\epsilon$, we must have $x_2 - x_1 = \sqrt{6\epsilon}$. Therefore, the set constructed in Lemma 1 has $\Theta(\epsilon^{-1/2})$ points and by Lemma 1, $\mathrm{ROPT}(f, \epsilon) = \Theta(\epsilon^{-1/2})$. By Lemma 6, there is a family of problem instances such that any algorithm requires $\Omega((\epsilon^{-1/2})^{4/3}) = \Omega(\epsilon^{-2/3})$ samples on some problem instance. □

# 8 Conclusion

We gave optimally adaptive deterministic and randomized algorithms for LIPSCHITZ-INTEGRATION. To simplify the analysis, we have been lax with constant factors in the randomized algorithm and the related proofs. Thus, it is possible to improve both the algorithm's performance and its analysis by constant factors.

A more interesting open problem is to design adaptive algorithms for definite integration over two or higher-dimensional domains or to prove that good adaptive algorithms do not exist. Although simple Monte Carlo methods readily extend to higher dimensions, designing and analyzing adaptive algorithms seems difficult.

# References


[1] Lucio Barabesi and Marzia Marcheselli. A modified monte carlo integration. *International Mathematical Journal*, 3(5):555–565, 2003.

[2] Ilya Baran. Adaptive algorithms for problems involving black-box lipschitz functions. Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 2004.

At `http://www.mit.edu/~ibaran/papers/mthesis.{pdf,ps}`.

[3] Ilya Baran and Erik D. Demaine. Optimal adaptive algorithms for finding the nearest and farthest point on a parametric black-box curve. In *Proceedings of the 20th Annual ACM Symposium on Computational Geometry*, Brooklyn, NY, June 2004. To appear.

[4] Therese Biedl, Broňa Brejova, Erik D. Demaine, Angèle M. Hamel, Alejandro López-Ortiz, and Tomáš Vinař. Finding hidden independent sets in interval graphs. *Theoretical Computer Science*, 310(1–3):287–307, January 2004.

[5] Philip J. Davis and Philip Rabinowitz. *Methods of Numerical Integration*. Academic Press, San Diego, second edition, 1984.

[6] Erik D. Demaine, Alejandro López-Ortiz, and J. Ian Munro. Adaptive set intersections, unions, and differences. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 743–752, San Francisco, California, January 2000.

[7] Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences*, 66(4):614–656, 2003.

[8] Pierre Hansen, Brigitte Jaumard, and Shi-Hui Lu. On the number of iterations of piyavskii's global optimization algorithm. *Mathematics of Operations Research*, 16(2):334–350, May 1991.

[9] John C. Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10):527–545, December 1996.

[10] S.A. Piyavskii. An algorithm for finding the absolute extremum of a function. *USSR Computational Mathematics and Mathematical Physics*, 12:57–67, 1972.

[11] Herbert Robbins. A remark on stirling's formula. *The American Mathematical Monthly*, 62(1):26–29, 1955.

[12] J.F. Traub, G.W. Wasilkowski, and H. Woźniakowski. *Information-Based Complexity*. Academic Press, New York, 1988.

[13] Arthur G. Werschulz. An overview of information-based complexity. Technical Report CUCS-022-02, Computer Science Department, Columbia University, October 2002.

[14] Z. Zabinsky, B.P. Kristinsdottir, and R.L. Smith. Optimal estimation of univariate black-box lipschitz functions with upper and lower error bounds. *Computers and Operations Research*, 30(10):1539–1553, 2003.

# Appendix

**Derivation Using Stirling Approximation in Proof of Lemma 6:** Stirling's approximation can be written as a double inequality, due to Robbins [11]:

$$\sqrt{2\pi}n^{n+1/2}e^{-n+1/(12n+1)} < n! < \sqrt{2\pi}n^{n+1/2}e^{-n+1/(12n)}.$$

Therefore,

$$\binom{q}{q/2}\Big/2^q = \frac{q!}{(q/2)!^2 2^q} < \frac{\sqrt{2\pi}q^{q+1/2}e^{-q+1/(12q)}}{\left(\sqrt{2\pi}(q/2)^{q/2+1/2}e^{-q/2+1/(6q+1)}\right)^2 2^q} =$$

$$= \frac{q^{q+1/2}e^{-q+1/(12q)}}{\sqrt{2\pi}(q/2)^{q+1}e^{-q+2/(6q+1)}2^q} =$$

$$= \frac{q^{q+1/2}e^{1/(12q)}}{\sqrt{\pi/2}q^{q+1}e^{2/(6q+1)}} = \sqrt{\frac{2}{\pi q}}e^{1/(12q)-2/(6q+1)} < \sqrt{\frac{2}{\pi q}}.$$

□