# Guaranteed, Adaptive, Automatic Algorithms for Univariate Integration: Methods, Cost and Implementation

Yizhi Zhang

Department of Applied Mathematics, Illinois Institute of Technology

October 29, 2018

# Contents

# Motivation

- $\int_a^b f(x)\mathrm{d}x$
- Simple functions such as $\int_a^b x^2\mathrm{d}x = x^3/3|_a^b$. Use pen and paper.
- Functions with no analytical solutions such as the standard normal p.d.f $\int_a^b e^{-x^2/2}/\sqrt{2\pi}\mathrm{d}x$. Use numerical methods.

# Fixed-cost algorithms

- The traditional trapezoidal rule and Simpson's rule are typical two examples.
-

$$\mathrm{err}(f,n) \leq \overline{\mathrm{err}}(f,n) := C(n)\,\mathrm{Var}(f^{(p)}), \tag{1}$$

- The algorithms provide guarantees when $\mathrm{Var}(f^{(p)}) < \sigma$.
- The algorithms are automatic. Cost $n = C^{-1}(\varepsilon/\sigma)$.
- The algorithms are not adaptive. The upper bound on variation, $\sigma$ is not dependent on the integrand.
- A constant $c \cdot f$ may not work.

# Adaptive algorithms with no guarantees

- MATLAB's `integral` and Chebfun toolbox are typical two examples.
- The algorithms are adaptive. No $\sigma$ required.
- The algorithms provide no guarantees.
- The algorithms can always be fooled by spiky functions.

# What do we think of the problem

- The goal is to construct adaptive, automatic algorithms with guarantees of success.
- Define the space of integrands in the cone, not ball.
- Error bound (1) is approximated using only function values.
- To fixed-cost algorithms: adaptive, succeed in the cone.
- To adaptive algorithms: rigorous proof of how spiky $f$ can be.

# Contents

# work flow

# Trapezoidal rule and Simpson's rule

The composite trapezoidal rule can be defined as:

$$T(f,n) = \frac{b-a}{2n} \sum_{j=0}^{n} (f(u_j) + f(u_{j+1})), \tag{2}$$

where

$$u_j = a + \frac{j(b-a)}{n}, \qquad j = 0, \ldots, n, \qquad n \in \mathbb{N}. \tag{3}$$

The composite Simpson's rule can be defined as:

$$S(f,n) = \frac{b-a}{18n} \sum_{j=0}^{3n-1} (f(v_{2j}) + 4f(v_{2j+1}) + f(v_{2j+2})), \tag{4}$$

where

$$v_j = a + \frac{j(b-a)}{6n}, \qquad j = 0, \ldots, 6n, \qquad n \in \mathbb{N}. \tag{5}$$

# Error of two methods

To better define (1):

$$\mathsf{err}(f,n) \le \overline{\mathsf{err}}(f,n) := C(n)\,\mathrm{Var}(f^{(p)}), \tag{6}$$

$$C(n) = \begin{cases} \frac{(b-a)^2}{8n^2}, p = 1, & \text{trapezoidal rule,} \\ \frac{(b-a)^4}{93312n^4}, p = 3, & \text{Simpson's rule.} \end{cases}$$

The error bounds contains the total variation, which is unknown to the users.

# Variation and function spaces

$$\mathrm{Var}(f) := \sup \left\{ \widehat{V}(f, \{x_i\}_{i=0}^{n+1}), \quad \text{for any } n \in \mathbb{N}, \text{ and } \{x_i\}_{i=0}^{n+1} \right\}, \tag{7}$$

where

$$\widehat{V}(f, \{x_i\}_{i=0}^{n+1}) = \sum_{i=1}^{n-1} |f(x_{i+1}) - f(x_i)|. \tag{8}$$

To ensure a finite error bound, our algorithms are defined for function spaces with finite total variations of the respective derivatives:

$$\mathcal{V}^p := \{f \in C^{p-1}[a, b] : \mathrm{Var}(f^{(p)}) < \infty\}, \tag{9}$$

where for the trapezoidal rule, $p = 1$, and for the Simpson's rule, $p = 3$.

# Spaces and subspaces (cones)

Our algorithms will not work for all $f \in \mathcal{V}^p$ but will work for integrands in cones:

$$\mathcal{C}^p := \Big\{ f \in \mathcal{V}^p, \mathrm{Var}(f^{(p)}) \le \mathfrak{C}(\mathsf{size}(\{x_i\}_{i=0}^{n+1}))\widehat{V}(f^{(p)}, \{x_i\}_{i=0}^{n+1}),$$
$$\text{for all choices of } n \in \mathbb{N}, \text{and } \{x_i\}_{i=0}^{n+1} \text{ with } \mathsf{size}(\{x_i\}_{i=0}^{n+1}) < \mathfrak{h} \Big\}. \quad (10)$$

$$\mathsf{size}(\{x_i\}_{i=0}^{n+1}) := \max_{i=0,\cdots,n} x_{i+1} - x_i. \quad (11)$$

$$\mathfrak{C}(h) := \frac{\mathfrak{C}(0)}{1 - h/\mathfrak{h}}; \quad \mathfrak{C}(0) > 1 \quad (12)$$

# How to find error bounds

- The definition of the cone suggests a way of bounding $\mathrm{Var}(f^{(p)})$ in terms of $\widehat{V}(f^{(p)}, \{x_i\}_{i=0}^{n+1})$.
- Our algorithms are based on function values and $\widehat{V}(f^{(p)}, \{x_i\}_{i=0}^{n+1})$ is defined in terms of derivative values.
- Find a way to express derivative values in terms of function values.

# Trapezoidal rule

The backward finite difference is used to approximate $f'$. Let the width of the interval $h = u_{j+1} - u_j = (b-a)/n$ and

$$f[u_j] = f(u_j), \text{ for } j = 0, \cdots, n,$$
$$f[u_j, u_{j-1}] = \frac{f(u_j) - f(u_{j-1})}{h}, \text{ for } j = 1, \cdots, n.$$

According to Mean Value Theorem for finite differences,

$$f'(x_j) = \frac{f(u_j) - f(u_{j-1})}{h} = \frac{n}{b-a}[f(u_j) - f(u_{j-1})], \tag{13}$$

for some $x_j \in (u_{j-1}, u_j)$.

$$\text{size}(\{x_j\}_{j=0}^{n+1}) \leq 2h = 2(b-a)/n < \mathfrak{h}. \tag{14}$$

# Trapezoidal rule, continued

The approximation $\widetilde{V}_1(f,n)$ to $\widehat{V}(f',\{x_j\}_{i=0}^{n+1})$ using only function values is defined as:

$$\begin{aligned}\widehat{V}(f',\{x_j\}_{j=0}^{n+1}) &= \sum_{j=1}^{n-1} |f'(x_{j+1}) - f'(x_j)|\,, \\ &= \sum_{j=1}^{n-1} \left| \frac{n}{b-a} [f(u_{j+1}) - f(u_j) - f(u_j) + f(u_{j-1})] \right| \\ &= \frac{n}{b-a} \sum_{j=1}^{n-1} |f(u_{j+1}) - 2f(u_j) + f(u_{j-1})| =: \widetilde{V}_1(f,n). \end{aligned} \tag{15}$$

# Trapezoidal rule, error bound

$$
\begin{aligned}
\overline{\mathsf{err}}_{\mathsf{t}}(f,n) :=& C(n)\,\mathrm{Var}(f'), && \text{(by (6))}\\
\le& C(n)\mathfrak{C}(\mathsf{size}(\{x_i\}_{i=0}^{n+1}))\widehat{V}(f',\{x_i\}_{i=0}^{n+1}), && \text{(by (10))}\\
=& C(n)\mathfrak{C}(\mathsf{size}(\{x_j\}_{i=0}^{n+1}))\widetilde{V}_1(f,n), && \text{(by (15))}\\
\le& \frac{(b-a)^2\mathfrak{C}(2(b-a)/n)\widetilde{V}_1(f,n)}{8n^2}. && \text{(by (14))}
\end{aligned}
$$

# Simpson's rule

The third order backward finite difference is used to approximate $f'''$. Let $h = v_{j+1} - v_j = (b-a)/6n$ be the width of the interval and

$$\begin{aligned}
f[v_j] &= f(v_j), && \text{for } j = 0, \cdots, 6n,\\
f[v_j, v_{j-1}] &= \frac{f(v_j) - f(v_{j-1})}{h}, && \text{for } j = 1, \cdots, 6n,\\
f[v_j, v_{j-1}, v_{j-2}] &= \frac{f(v_j) - 2f(v_{j-1}) + f(v_{j-2})}{2h^2}, && \text{for } j = 2, \cdots, 6n,\\
f[v_j, v_{j-1}, v_{j-2}, v_{j-3}] &= \frac{f(v_j) - 3f(v_{j-1}) + 3f(v_{j-2}) - f(v_{j-3})}{6h^3}, && \text{for } j = 3, \cdots, 6n.
\end{aligned}$$

According to Mean Value Theorem for finite differences,

$$f'''(x_j) == \frac{216n^3}{(b-a)^3}[f(v_{3j}) - 3f(v_{3j-1}) + 3f(v_{3j-2}) - f(v_{3j-3})]. \tag{16}$$

for some $x_j \in (v_{3j-3}, v_{3j})$.

$$\text{size}(\{x_j\}_{i=0}^{2n+1}) \le 6h = (b-a)/n < \mathfrak{h}. \tag{17}$$

# Simpson's rule, continued

The approximation $\widetilde{V}_3(f,n)$ to $\widehat{V}(f''', \{x_i\}_{i=0}^{n+1})$ using only function values can be written as follow:

$$\begin{aligned}
\widehat{V}(f''', \{x_j\}_{j=0}^{n+1}) &= \sum_{j=1}^{n-1} |f'''(x_{j+1}) - f'''(x_j)|, \\
&= \sum_{j=1}^{2n-1} \left| \frac{216n}{(b-a)^3} [(f(v_{3j+3}) - 3f(v_{3j+2}) + 3f(v_{3j+1}) - f(v_{3j})) \right. \\
&\qquad \left. -(f(v_{3j}) - 3f(v_{3j-1}) + 3f(v_{3j-2}) - f(v_{3j-3}))] \right| \\
&= \frac{216n^3}{(b-a)^3} \sum_{j=1}^{2n-1} |f(v_{3j+3}) - 3f(v_{3j+2}) + 3f(v_{3j+1}) \\
&\qquad -2f(v_{3j}) + 3f(v_{3j-1}) - 3f(v_{3j-2}) + f(v_{3j-3})| =: \widetilde{V}_3(f,n).
\end{aligned} \tag{18}$$

# Simpson's rule, error bound

Therefore the error bound on error estimate for our Simpson's rule algorithm using only function values is obtained as:

$$\begin{aligned}
\overline{\mathsf{err}}_{\mathsf{s}}(f,n) :=& C(n)\,\mathrm{Var}(f'''), && \text{(by (6))}\\
\leq& C(n)\mathfrak{C}(\mathsf{size}(\{x_i\}_{i=0}^{2n+1}))\widehat{V}(f''',\{x_i\}_{i=0}^{2n+1}), && \text{(by (10))}\\
=& C(n)\mathfrak{C}(\mathsf{size}(\{x_j\}_{i=0}^{2n+1}))\widetilde{V}_3(f,n), && \text{(by (18))}\\
\leq& \frac{(b-a)^4\mathfrak{C}((b-a)/n)\widetilde{V}_3(f,n)}{93312n^4}. && \text{(by (17))}
\end{aligned}$$

# Multi-step trapezoidal rule algorithm

## Algorithm (Trapezoidal Rule Adaptive Algorithm `integral_t`)

Let the sequence of algorithms $\{T_n\}_{n\in\mathbb{N}}$, and $\widetilde{V}_1(\cdot,\cdot)$ be as described above. Let $\mathfrak{h} \le (b-a)$. Set $k=0$, $n_k=1$. For any integrand $f$ and error tolerance $\varepsilon$, do the following:

Step 1. (Re)set the upper bound on $\mathrm{Var}(f')$; increase number of nodes. Let
$$\eta_k = \infty \text{ and } n_{k+1} = n_k \times \lceil 2(b-a)/(\mathfrak{h} n_k) \rceil.$$

Step 2. Compute the largest lower bound on $\mathrm{Var}(f')$. Let $k = k+1$. Compute $\widetilde{V}_1(f, n_k)$ in (15).

Step 3. Compute the smallest upper bound on $\mathrm{Var}(f')$. Compute
$$\eta_k = \min\left(\eta_{k-1}, \mathfrak{C}(2(b-a)/n_k)\widetilde{V}_1(f,n_k)\right).$$

# Multi-step trapezoidal rule algorithm, continued

## Algorithm

Step 4. Check the necessary condition for $f \in \mathcal{C}^1$. If $\widetilde{V}_1(f, n_k) \le \eta_k$, then go to Step 5. Otherwise, set $\mathfrak{h} = \mathfrak{h}/2$.

1. Let $\mathfrak{J} = \{j \in \{1, \cdots, k\} : n_j \ge 2(b-a)/\mathfrak{h}\}$. If $\mathfrak{J}$ is empty, go to Step 1.
2. Otherwise, recompute the upper bound on $\mathrm{Var}(f')$, for all $n_j$, with $j \in \mathfrak{J}$ as follows:

   For $j' = \min \mathfrak{J}$, let $\eta_{j'} = \mathfrak{C}(2(b-a)/n_{j'})\widetilde{V}_1(f, n_{j'})$,

   Compute

   $\eta_j = \min\{\eta_{j-1}, \mathfrak{C}(2(b-a)/n_j)\widetilde{V}_1(f, n_j)\}$, for $j = j' + 1, \cdots, k$.

   Go to the beginning of Step 4.

# Multi-step trapezoidal rule algorithm, continued

## Algorithm

Step 5. Check for convergence. Check whether $n_k$ is large enough to satisfy the error tolerance, i.e.

$$n_k^2 \geq \frac{\eta_k (b-a)^2}{8\varepsilon}.$$

1. If this is true, return $\texttt{integral\_t}(f, \varepsilon) = T_{n_k}(f)$ and terminate the algorithm.
2. Otherwise, go the Step 6.

Step 6. Increase the number of nodes and loop again. Choose

$$n_{k+1} = n_k \times \max\left( \left\lceil \frac{(b-a)}{n_k} \sqrt{\frac{\widetilde{V}_1(f, n_k)}{8\varepsilon}} \right\rceil, 2 \right).$$

Go to Step 2.

# Multi-step Simpson's rule algorithm

## Algorithm (Simpson's Rule Adaptive Algorithm `integral_s`)

Let the sequence of algorithms $\{S_n\}_{n\in\mathbb{N}}$, and $\widetilde{V}_3(\cdot,\cdot)$ be as described above. Let $\mathfrak{h} \le (b-a)/6$. Set $k=0$, $n_k=0$. For any integrand $f$ and error tolerance $\varepsilon$, do the following:

Step 1. (Re)set the upper bound on $\mathrm{Var}(f''')$; increase the number of nodes.
Let $\eta_k = \infty$ and $n_{k+1} = n_k \times \lceil (b-a)/\mathfrak{h} n_k \rceil$.

Step 2. Compute the largest lower bound on $\mathrm{Var}(f''')$. Let $k = k+1$. Compute $\widetilde{V}_3(f, n_k)$ in (18).

Step 3. Compute the smallest upper bound on $\mathrm{Var}(f''')$. Compute

$$\eta_k = \min\left(\eta_{k-1}, \mathfrak{C}((b-a)/n_k)\widetilde{V}_3(f,n_k)\right).$$

# Multi-step Simpson's rule algorithm, continued

## Algorithm

Step 4. Check the necessary condition for $f \in \mathcal{C}^3$. If $\widetilde{V}_3(f, n_k) \leq \eta_k$, go to Step 5. Otherwise, set $\mathfrak{h} = \mathfrak{h}/2$.

1. Let $\mathfrak{J} = \{j \in \{1, \cdots, k : n_j\} \geq (b-a)/\mathfrak{h}\}$. If $\mathfrak{J}$ is empty, go to Step 1.
2. Otherwise, recompute the upper bound on $\mathrm{Var}(f''')$ for all $n_j$, with $j \in \mathfrak{J}$ as follows:

   For $j' = \min \mathfrak{J}$, let $\eta_{j'} = \mathfrak{C}((b-a)/n_{j'})\widetilde{V}_3(f, n_{j'})$,

   Compute

   $$\eta_j = \min\{\eta_{j-1}, \mathfrak{C}((b-a)/n_j)\widetilde{V}_3(f, n_j)\}, \text{ for } j = j'+1, \cdots, k.$$

   Go to the beginning of Step 4.

# Multi-step Simpson's rule algorithm, continued

## Algorithm

Step 5. Check for convergence. Check whether $n_k$ is large enough to satisfy the error tolerance, i.e.

$$n_k^4 \geq \frac{\eta_k (b-a)^4}{93312\varepsilon}.$$

1. If true, return $\texttt{integral\_s}(f, \varepsilon) = S_{n_k}(f)$ and terminate the algorithm.
2. Otherwise, go the Step 6.

Step 6. Increase the number of nodes and loop again. Choose

$$n_{k+1} = n_k \times \max\left( \left\lceil \frac{(b-a)}{n_k} \left( \frac{\widetilde{V}_3(f, n_k)}{93312\varepsilon} \right)^{1/4} \right\rceil, 2 \right).$$

Go to Step 2.

# Computational cost of the trapezoidal rule

## Theorem

*Let $N(f, \varepsilon)$ denote the final number $n_k$ in Step 5 when* `integral_t` *terminates. Then this number is bounded below and above in terms of the true, yet unknown,* $\operatorname{Var}(f')$.

$$\max\left(\left\lfloor \frac{2(b-a)}{\mathfrak{h}} \right\rfloor, \left\lceil 2(b-a)\sqrt{\frac{\operatorname{Var}(f')}{8\varepsilon}} \right\rceil\right) \le N(f,\varepsilon)$$

$$\le 2\min\left\{ n \in \mathbb{N} : n \ge \left(\left\lfloor \frac{2(b-a)}{\mathfrak{h}} \right\rfloor\right), \zeta(n)\operatorname{Var}(f') \le \varepsilon \right\}$$

$$\le 2 \min_{0<\alpha\le 1} \max\left(\left(\left\lfloor \frac{2(b-a)}{\alpha\mathfrak{h}} \right\rfloor\right), (b-a)\sqrt{\frac{\mathfrak{C}(\alpha\mathfrak{h})\operatorname{Var}(f')}{8\varepsilon}}\right), \quad (19)$$

*where $\zeta(n) = (b-a)^2\mathfrak{C}(2(b-a)/n)/(8\varepsilon n^2)$. The number of function values, namely, the computational cost required by the algorithm, is $N(f,\varepsilon)+1$.*

# Computational cost of the Simpson's rule

## Theorem

*Let $N(f, \varepsilon)$ denote the final number of $n_k$ in Step 5 when* `integral_s` *terminates. Then this number is bounded below and above in terms of the true, yet unknown,* $\mathrm{Var}(f''')$.

$$\max\left(\left\lfloor\frac{(b-a)}{\mathfrak{h}}\right\rfloor, \left\lceil(b-a)\left(\frac{\mathrm{Var}(f''')}{93312\varepsilon}\right)^{1/4}\right\rceil\right) \le N(f,\varepsilon)$$

$$\le 2\min\left\{n\in\mathbb{N}: n \ge \left(\left\lfloor\frac{(b-a)}{\mathfrak{h}}\right\rfloor\right), \zeta(n)\,\mathrm{Var}(f''') \le \varepsilon\right\}$$

$$\le 2\min_{0<\alpha\le 1}\max\left(\left(\left\lfloor\frac{(b-a)}{\alpha\mathfrak{h}}\right\rfloor\right), (b-a)\left(\frac{\mathfrak{C}(\alpha\mathfrak{h})\,\mathrm{Var}(f''')}{93312\varepsilon}\right)^{1/4}\right), \quad (20)$$

*where* $\zeta(n) = (b-a)^4\mathfrak{C}((b-a)/n)/(93312n^4)$. *The number of function values, namely the computational cost, required by the algorithm is* $6N(f,\varepsilon)+1$.

# Computational cost of the Simpson's rule, continued

## Proof.

No matter what inputs $f$ and $\varepsilon$ are provided, $N(f,\varepsilon) \geq n_1 = \lfloor (b-a)/\hbar \rfloor$. Then the number of intervals increases until $\overline{\text{err}}(f,n) \leq \varepsilon$. From the error bound defined in (6), $C(N(f,\varepsilon))\,\text{Var}(f''') \leq \varepsilon$. Thus $N(f,\varepsilon) \geq \left\lceil (b-a)\left(\frac{\text{Var}(f''')}{93312\varepsilon}\right)^{1/4} \right\rceil$. This implies the lower bound on $N(f,\varepsilon)$.

# Computational cost of the Simpson's rule, continued

## Proof.

Let $K$ be the value of $k$ for which `integral_s` terminates. Since $n_1$ satisfies the upper bound, one may assume that $K \geq 2$. Let $m$ be the multiplication integer found in Step 6. Note that $\zeta((m-1)n_{K-1})\,\mathrm{Var}(f''') > \varepsilon$. For $m = 2$, this is true because $\zeta(n_{K-1})\,\mathrm{Var}(f''') \geq \zeta(n_{K-1})\widetilde{V}_3(f, n_{K-1}) > \varepsilon$. For $m > 2$ it is true because of the definition of $m$. Since $\zeta$ is a decreasing function, it follows that

$$(m-1)n_{K-1} < n^* := \min\left\{n \in \mathbb{N} : n \geq \left\lfloor \frac{2(b-a)}{n} \right\rfloor, \zeta(n)\,\mathrm{Var}(f''') \leq \varepsilon\right\}.$$

Therefore $n_K = m^* n_{K-1} < m^* \frac{n^*}{m-1} = \frac{m}{m-1} n^* \leq 2n^*$.

# Computational cost of the Simpson's rule, continued

## Proof.

For fixed $\alpha \in (0, 1]$, we only need to consider that case where $n^* > \lfloor 2(b-a)/(\alpha\mathfrak{h}) \rfloor$. This implies that $n^* > \lfloor 2(b-a)/(\alpha\mathfrak{h}) \rfloor \geq 2(b-a)/(\alpha\mathfrak{h})$ thus $\alpha\mathfrak{h} \geq 2(b-a)/(n^*)$. Also by the definition of $n^*$, $\zeta$, and $\mathfrak{C}$ is non-decreasing:

$$
\begin{aligned}
& \zeta(n^*)\,\mathrm{Var}(f''') > \varepsilon, \\
\Rightarrow\ & 1 < \left(\frac{\zeta(n^*)\,\mathrm{Var}(f''')}{\varepsilon}\right)^{1/4}, \\
\Rightarrow\ & n^* < n^* \left(\frac{\zeta(n^*)\,\mathrm{Var}(f''')}{\varepsilon}\right)^{1/4}, \\
& = n^* \left(\frac{(b-a)^4 \mathfrak{C}(2(b-a)/(n^*))\,\mathrm{Var}(f''')}{93312(n^*)^4\varepsilon}\right)^{1/4}, \\
& \leq (b-a) \left(\frac{\mathfrak{C}(\alpha\mathfrak{h})\,\mathrm{Var}(f''')}{93312\varepsilon}\right)^{1/4}.
\end{aligned}
$$

□

# Lower bound on complexity for the Simpson's rule

Use

$$\mathsf{bump}(x;t,h) := \begin{cases} (x-t)^3/6, & t \le x < t+\delta, \\ [-3(x-t)^3 + 12\delta(x-t)^2 & \\ \qquad -12\delta^2(x-t) + 4\delta^3]/6, & t+\delta \le x < t+2\delta, \\ [3(x-t)^3 - 24\delta(x-t)^2 & \\ \qquad +60\delta^2(x-t) - 44\delta^3]/6, & t+2\delta \le x < t+3\delta, \\ (t+4\delta-x)^3/6, & t+3\delta \le x < t+4\delta, \\ 0, & \text{otherwise}, \end{cases} \tag{21}$$

to build the fooling function

$$\text{twobp}(x; t, \delta, \pm) := \text{bump}(x; a, \mathfrak{h}) \pm \frac{15[\mathfrak{C}(\delta) - 1]}{16} \text{bump}(x; t, \delta)$$

Although $\text{twobp}'''(x; t, \delta, \pm)$ may have a bump with arbitrarily small width $4\delta$, the height is small enough for $\text{twobp}'''(x; t, \delta, \pm)$ to lie in the cone.

# Lower bound on complexity for the trapezoidal rule

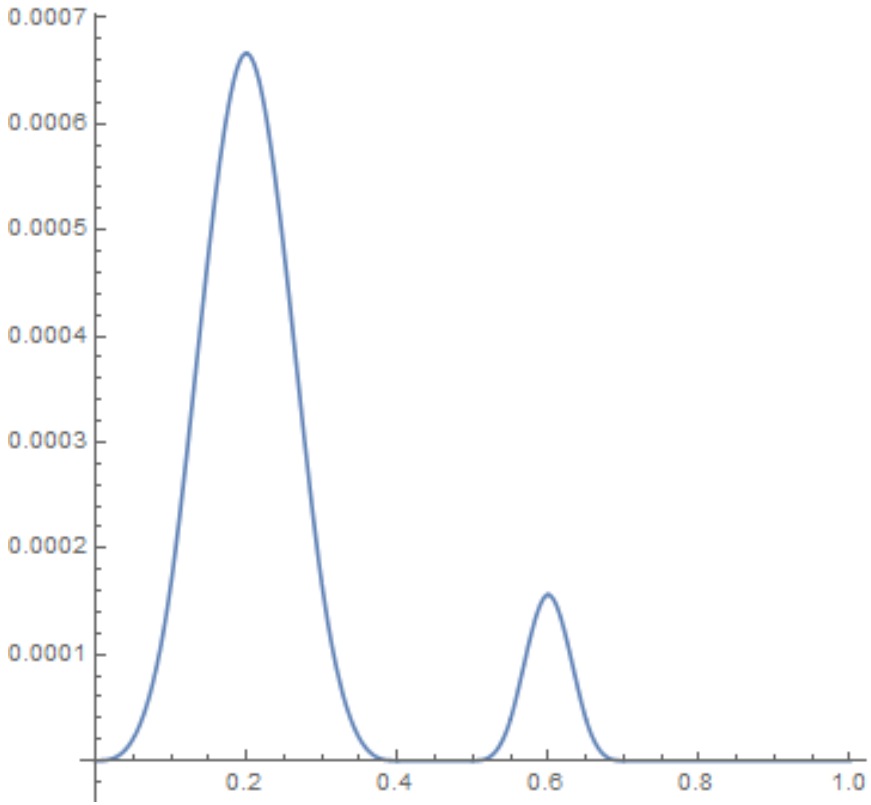


Figure: A fooling function that can fool the Simpson's rule algorithms, where $a = 0$, $b = 1$, $t = 0.5$, $\delta = 0.05$ and $\mathfrak{h} = 0.1$.

# Lower bound on the complexity for the Simpson's rule

## Theorem

*Let* $\mathtt{int}$ *be any (possibly adaptive) algorithm that succeeds for all integrands in* $\mathcal{C}^3$*, and only uses integrand values. For any error tolerance* $\varepsilon > 0$ *and any arbitrary value of* $\mathrm{Var}(f''')$*, there will be some* $f \in \mathcal{C}^3$ *for which* $\mathtt{int}$ *must use at least*

$$-\frac{5}{4} + \frac{b-a-5\mathfrak{h}}{8}\left[\frac{[\mathfrak{C}(0)-1]\,\mathrm{Var}(f''')}{\varepsilon}\right]^{1/4} \tag{22}$$

*function values. As* $\mathrm{Var}(f''')/\varepsilon \to \infty$ *the asymptotic rate of increase is the same as the computational cost of* `integral_s`*. Thus* `integral_s` *has optimal order for integration of functions in* $\mathcal{C}^3$*.*

# Lower bound on the complexity for the Simpson's rule, continued

## Proof.

For any positive $\alpha$, suppose that $\texttt{int}(\cdot, \varepsilon)$ evaluates integrand $\alpha\text{bump}'''(\cdot; t, \delta)$ at $n$ nodes before returning to an answer. Let $\{x_j\}_{j=1}^m)$ be the $m < n$ ordered nodes used by $\texttt{int}(\cdot, \varepsilon)$ that fall in the interval $(x_0, x_{m+1})$ where $x_0 := a + 3\mathfrak{h}$, $x_{m+1} := b - \delta$ and $\delta := (b - a - 5\mathfrak{h})/(4n + 5)$. There must exits at least one of these $x_j$ with $i = 0, \cdots, m$ for which

$$\frac{x_{j+1} - x_j}{4} \geq \frac{x_{m+1} - x_0}{4(m+1)} \geq \frac{x_{m+1} - x_0}{4(n+1)} = \frac{b - a - 5\mathfrak{h} - \delta}{4n + 4} = \delta.$$

# Lower bound on the complexity for the Simpson's rule, continued

## Proof.

Choose one such $x_j$ and call it $t$. The choice of $t$ and $\delta$ ensures that $\texttt{int}(\cdot, \varepsilon)$ cannot distinguish between $\alpha\text{bump}(\cdot; t, \delta)$ and $\alpha\text{twobp}(\cdot; t, \delta, \pm)$. Thus

$$\texttt{int}(\alpha\text{twobp}(\cdot; t, \delta, \pm), \varepsilon) = \texttt{int}(\alpha\text{bump}(\cdot; t, \delta), \varepsilon)$$

Moreover, $\alpha\text{bump}(\cdot; t, \delta)$ and $\alpha\text{twobp}(\cdot; t, \delta, \pm)$ are all in the cone $\mathcal{C}^3$. This means that $\texttt{int}(\cdot, \varepsilon)$ is successful for all of the functions.

# Lower bound on the complexity for the Simpson's rule, continued

## Proof.

$$\varepsilon \geq \frac{1}{2}\Big[\left|\int_a^b \alpha\mathsf{twobp}(x;t,\delta,-)dx - \mathtt{int}(\alpha\mathsf{twobp}(\cdot;t,\delta,-),\varepsilon)\right| + \left|\int_a^b \alpha\mathsf{twobp}(x;t,\delta,+)dx - \mathtt{int}(\alpha\mathsf{twobp}(\cdot;t,\delta,+),\varepsilon)\right|\Big]$$

$$\begin{aligned}
&\geq \frac{1}{2}\left|\int_a^b \alpha\mathsf{twobp}(x;t,\delta,+)dx - \int_a^b \alpha\mathsf{twobp}(x;t,\delta,-)dx\right| \\
&= \int_a^b \alpha\mathsf{bump}(x;t,\delta)dx \\
&= \frac{15\alpha[\mathfrak{C}(\delta)-1]\delta^4}{16} \\
&= \frac{[\mathfrak{C}(\delta)-1]\delta^4\,\mathrm{Var}(\alpha\mathsf{bump}'''(\cdot;a,\mathfrak{h}))}{16}
\end{aligned}$$

# Lower bound on the complexity for the Simpson's rule, continued

## Proof.

Substituting $\delta$ in terms of $n$:

$$4n+5 = \frac{b-a-5\mathfrak{h}}{\delta} \geq (b-a-5\mathfrak{h}) \left[\frac{[\mathfrak{C}(\delta)-1]\,\mathrm{Var}(\alpha \mathsf{bump}'''(\cdot;a,\mathfrak{h}))}{16\varepsilon}\right]^{1/4},$$

$$\geq -\frac{5}{4} + \frac{b-a-5\mathfrak{h}}{8} \left[\frac{[\mathfrak{C}(0)-1]\,\mathrm{Var}(\alpha \mathsf{bump}'''(\cdot;a,\mathfrak{h}))}{\varepsilon}\right]^{1/4}.$$

Since $\alpha$ is an arbitrary positive number, the value of $\mathrm{Var}(\alpha \mathsf{bump}'''(\cdot;a,\mathfrak{h}))$ is arbitrary.

Finally, `integral_s` is optimal. □

# Numerical Example

The bump function bump$(x; t, \delta)$ defined in (21) is used as our test function to check the performance of our algorithms. Consider the family of bump test functions on interval $(0, 1)$ defined by

$$f(x, t, \delta) = \text{bump}(x; t, \delta)/\delta^4 \tag{24}$$

with $t \sim \mathcal{U}[0, 1 - 4\delta]$, $\log_{10}(\delta) \sim \mathcal{U}[-4, -1]$. The integration $\int_0^1 f(x, t, \delta)\mathrm{d}x = 1$.

# Experiment Setup

As an experiment, we chose $10000$ random test functions and applied `integral_t` and `integral_s` with an error tolerance of $\varepsilon = 10^{-8}$. The initial values of $\mathfrak{h}$ are set as $0.1$, $0.01$, and $0.001$. The algorithm is considered successful for a particular $f$ if the exact and approximate integrals have a difference no greater than $\varepsilon$. Our algorithms give warnings when the integrand is not in the original cone.

Some commonly available numerical algorithms in MATLAB are `integral` and the Chebfun toolbox. We applied these two routines to the random family of test functions as well. Their success and failure rates are also recorded in Table 1. They do not give warnings of possible failure.

# Results

Table: The success rates of `integral_t` and `integral_s` plus the success rates of other common quadrature algorithms.

| | $\mathfrak{h}$ | Success | Success Warning |
|---|---|---|---|
| `integral_t` | 0.1 | 24.76% | 9.43% |
| | 0.01 | 57.36% | 3.70% |
| | 0.001 | 87.38% | 0.81% |
| `integral_s` | 0.1 | 25.94% | 14.28% |
| | 0.01 | 57.63% | 4.45% |
| | 0.001 | 94.09% | 0.87% |
| `integral` | | 29.68 % | |
| `chebfun` | | 18.91% | |

# Convergence rates

The test function is $f(x,t,\delta) = \text{bump}(x;t,\delta)/\delta^4$ where $t = 0.2$; $\delta = 0.1$. Both `integral_t` and `integral_s` use $\mathfrak{h} = 0.1$.

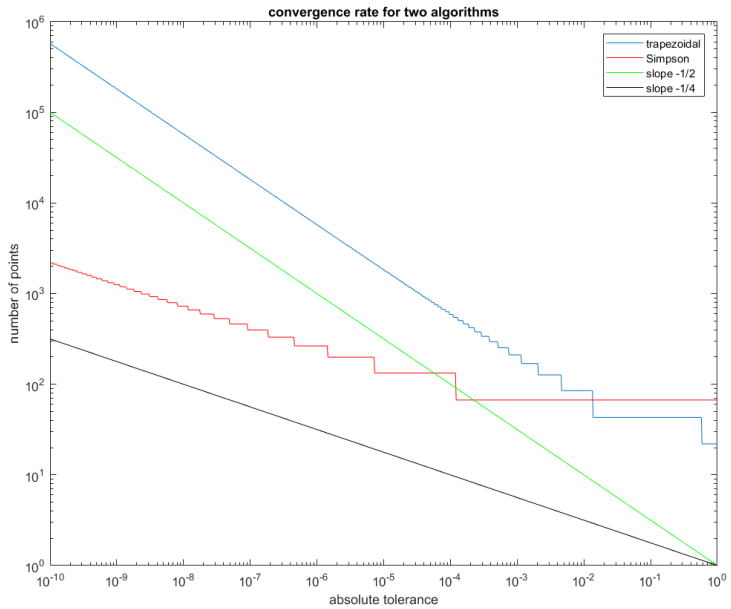


Figure: The convergence curves for `integral_t` and `integral_s`.

# Future Work

- Guaranteed automatic algorithms with higher order convergence rate.
- Locally adaptive algorithms.
- Relative Error.

# Thank you!

# how to find the inflation number

We hope the number of points for the next loop to be large enough such that

$$n_{k+1}^4 \geq \frac{(b-a)^4 \operatorname{Var}(f''')}{93312\varepsilon}.$$

Underestimate $\operatorname{Var}(f''')$ with $\widetilde{V}_1(f, n_k)$.

# how to find the quarter length

The average width of intervals is $(x_{m+1} - x_0)/(m+1)$ and $x_{m+1} - x_0 = b - a - 5\mathfrak{h} - \delta$. Choose one interval such that width less bigger than $4\delta$ . We can choose interval wider than average and $4\delta$ narrower than average:

$$\frac{x_{j+1} - x_j}{4} \geq \frac{x_{m+1} - x_0}{4(m+1)} \geq \frac{x_{m+1} - x_0}{4(n+1)} = \frac{b - a - 5\mathfrak{h} - \delta}{4n+4} = \delta,$$

$$\Rightarrow \delta = \frac{b - a - 5\mathfrak{h}}{4n+5}$$

# Computational cost of the trapezoidal rule

## Theorem

*Let $N(f, \varepsilon)$ denote the final number $n_k$ in Step 5 when* `integral_t` *terminates. Then this number is bounded below and above in terms of the true, yet unknown,* $\mathrm{Var}(f')$.

$$\max\left(\left\lfloor \frac{2(b-a)}{\mathfrak{h}} \right\rfloor, \left\lceil 2(b-a)\sqrt{\frac{\mathrm{Var}(f')}{8\varepsilon}} \right\rceil\right) \le N(f,\varepsilon)$$

$$\le 2\min\left\{ n \in \mathbb{N} : n \ge \left(\left\lfloor \frac{2(b-a)}{\mathfrak{h}} \right\rfloor\right), \zeta(n)\,\mathrm{Var}(f') \le \varepsilon \right\}$$

$$\le 2 \min_{0<\alpha\le 1} \max\left(\left(\left\lfloor \frac{2(b-a)}{\alpha\mathfrak{h}} \right\rfloor\right), (b-a)\sqrt{\frac{\mathfrak{C}(\alpha\mathfrak{h})\,\mathrm{Var}(f')}{8\varepsilon}}\right), \quad (25)$$

*where $\zeta(n) = (b-a)^2\mathfrak{C}(2(b-a)/n)/(8\varepsilon n^2)$. The number of function values, namely, the computational cost required by the algorithm, is $N(f,\varepsilon)+1$.*

# Computational cost of the trapezoidal rule, continued

## Proof.

No matter what inputs $f$ and $\varepsilon$ are provided, $N(f,\varepsilon) \geq n_1 = \lfloor 2(b-a)/\mathfrak{h} \rfloor$. Then the number of intervals increases until $\overline{\mathrm{err}}(f,n) \leq \varepsilon$. From the error bound defined in (6), $C(N(f,\varepsilon))\,\mathrm{Var}(f') \leq \varepsilon$. Thus $N(f,\varepsilon) \geq \left\lceil 2(b-a)\sqrt{\mathrm{Var}(f')/(8\varepsilon)} \right\rceil$. This implies the lower bound on $N(f,\varepsilon)$.

# Computational cost of the trapezoidal rule, continued

## Proof.

Let $K$ be the value of $k$ for which `integral_t` terminates. Since $n_1 = \lfloor 2(b-a)/\mathfrak{h} \rfloor$ satisfies the upper bound, one may assume that $K \geq 2$. Let $m$ be the multiplication integer found in Step 6. Note that $\zeta((m-1)n_{K-1}) \operatorname{Var}(f') > \varepsilon$. For $m = 2$, this is true because $\zeta(n_{K-1}) \operatorname{Var}(f') \geq \zeta(n_{K-1}) \widetilde{V}_1(f, n_{K-1}) > \varepsilon$. For $m > 2$ it is true because of the definition of $m$. Since $\zeta$ is a decreasing function, it follows that

$$(m-1)n_{K-1} < n^* := \min \left\{ n \in \mathbb{N} : n \geq \left\lfloor \frac{2(b-a)}{n} \right\rfloor, \zeta(n) \operatorname{Var}(f') \leq \varepsilon \right\}.$$

Therefore $n_L = m n_{L-1} < m \frac{n^*}{m-1} = \frac{m}{m-1} n^* \leq 2n^*$.

# Computational cost of the trapezoidal rule, continued

## Proof.

For fixed $\alpha \in (0,1]$, we only need to consider that case where $n^* > \lfloor 2(b-a)/(\alpha\mathfrak{h}) \rfloor$. This implies that $n^* > \lfloor 2(b-a)/(\alpha\mathfrak{h}) \rfloor \geq 2(b-a)/(\alpha\mathfrak{h})$ thus $\alpha\mathfrak{h} \geq 2(b-a)/n^*$. Also by the definition of $n^*$, $\zeta$, and $\mathfrak{C}$ is non-decreasing:

$$
\begin{aligned}
&\zeta(n^*)\,\mathrm{Var}(f') > \varepsilon,\\
\Rightarrow\ & 1 < \left(\frac{\zeta(n^*)\,\mathrm{Var}(f')}{\varepsilon}\right)^{1/2},\\
\Rightarrow\ & n^* < n^* \left(\frac{\zeta(n^*-1)\,\mathrm{Var}(f')}{\varepsilon}\right)^{1/2},\\
&= n^* \left(\frac{(b-a)^2\mathfrak{C}(2(b-a)/n^*)\,\mathrm{Var}(f')}{8(n^*)^2\varepsilon}\right)^{1/2},\\
&\leq (b-a)\sqrt{\frac{\mathfrak{C}(\alpha\mathfrak{h})\,\mathrm{Var}(f')}{8\varepsilon}}.
\end{aligned}
$$

□