Constructing Guaranteed Automatic Numerical Algorithms for Univariate Integration

Yizhi Zhang, Joint work with Prof. Fred Hickernell

Department of Applied Mathematics, Illinois Institute of Technology

July 10, 2014
Contents

- Introduction.
- Automatic Numerical Integration, Theory and Practise.
- GAIL
- Future Work
By “automatic”, it is meant that the user provides a function $f$ and an error tolerance, $\varepsilon$, and the algorithm attempts to provide an approximate solution that is within a distance of $\varepsilon$ of the true solution. The algorithm will adaptively decide how many and which pieces of function data are needed.

We want to establish a framework for providing rigorous guarantees for automatic algorithms.
Motivation

- Non-adaptive methods provide guarantees.
- Adaptive methods provide no guarantee.
Non-adaptive, guaranteed

The user provides a function, the error tolerance, and some conditions.

\[ x \mapsto f(x), \varepsilon = \text{tolerance}, \sigma \text{ such that } \|f''\|_1 \leq \sigma. \]

For example, using the trapezoidal rule to compute \( \int_a^b f(x)dx \):

\[ \text{cost} = n + 1 = \left\lceil \sqrt{\frac{\sigma}{8\varepsilon}} \right\rceil + 1. \]

We can have an estimation of \( T_n(f) \) such that

\[ \left| \int_a^b f(x)dx - T_n(f) \right| \leq \varepsilon. \]

Guaranteed!
Adaptive, but not guaranteed

The user provides a function, and the error tolerance.

\[ x \mapsto f(x), \varepsilon = \text{tolerance}. \]

For example, using MATLAB’s integral to compute \( \int_a^b f(x)dx \), the cost depends on how hard the problem is. But, there is no guarantee to achieve an estimation of \( Q_n(f) \) such that

\[ \left| \int_a^b f(x)dx - Q_n(f) \right| \leq \varepsilon, \]
Assumptions

The node set and the linear spline algorithm using $n$ function values are defined for $n \in \mathcal{I} := \{2, 3, \ldots\}$ as follows:

$$x_i = a + \frac{i - 1}{n - 1} (b - a), \quad i = 1, \ldots, n,$$

$$A_n(f)(x) := \frac{n - 1}{b - a} \left[ f(x_i)(x_{i+1} - x) + f(x_{i+1})(x - x_i) \right]$$

for $x_i \leq x \leq x_{i+1}$. 

Yizhi Zhang, Joint work with Prof. Fred Hickernell
The problem to be solved is univariate integration on interval \([a, b]\),

\[ S(f) := \text{INT}(f) := \int_{a}^{b} f(x) \, dx \in \mathcal{G} := \mathbb{R}. \]

The fixed cost building blocks to construct the adaptive integration algorithm are the composite trapezoidal rules based on \(n - 1\) trapezoids:

\[ T_n(f) := \int_{a}^{b} A_n(f) \, dx = \frac{b - a}{2n - 2} [f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(x_n)]. \]
Assumptions, continued

The space of input functions is $\mathcal{V}$, the space of functions whose first derivatives have finite variation:

$$\mathcal{V}^1[a, b] = \{ f \in C^1[a, b] : \text{Var}(f') < \infty \}.$$

The space of outputs is the real space $\mathbb{R}$. The stronger semi-norm is $|f|_\mathcal{F} = \text{Var}(f')$, while the weaker semi-norm is

$$|f|_{\tilde{\mathcal{F}}} := \|f' - A_2(f)'\|_1 = \left\| f' - \frac{f(b) - f(a)}{b - a} \right\|_1 = \text{Var}(f - A_2(f)), $$

The cone of the integrand is defined as

$$\mathcal{C}_{\tau_{a,b}} := \left\{ f \in \mathcal{V}^1 : \text{Var}(f') \leq \frac{\tau_{a,b}}{b - a} \left\| f' - \frac{f(b) - f(a)}{b - a} \right\|_1 \right\}.$$ 

For simplicity, I will denote $\tau_{a,b}$ as $\tau$ for the rest of the context.
In practice, we estimate the stronger norm and the weaker norm in the following way:

\[
F_n(f) := \text{Var}(A_n(f)') = \frac{n - 1}{b - a} \sum_{i=1}^{n-2} \left| f(x_i) - 2f(x_{i+1}) + f(x_{i+2}) \right|,
\]

and

\[
\tilde{F}_n(f) := \left| A_n(f) \right|_{\tilde{\mathcal{F}}} = \left\| A_n(f)' - A_2(f)' \right\|_1
= \sum_{i=1}^{n-1} \left| f(x_{i+1}) - f(x_i) - \frac{f(b) - f(a)}{n - 1} \right|.
\]

(2)
Multi-step Automatic Algorithms

Algorithm (Adaptive Univariate Integration)

Let the sequence of algorithms \( \{T_n\}_{n \in \mathcal{I}} \), \( \{\tilde{F}_n\}_{n \in \mathcal{I}} \), and \( \{F_n\}_{n \in \mathcal{I}} \) be as described above. Choose integer \( n_{lo}, n_{hi} \), such that \( n_{lo} \leq n_{hi} \). Set \( i = 1 \). Let \( n_1 = \max \left\{ \left\lceil n_{hi} \left( \frac{n_{lo}}{n_{hi}} \right)^{\frac{1}{1+b-a}} \right\rceil, 3 \right\} \). Let \( \tau_{a,b} = 2n_1 - 3 \). For any error tolerance \( \varepsilon \) and input function \( f \), do the following:

Stage 1. Estimate \( \left\| f' - \frac{f(b) - f(a)}{b-a} \right\|_1 \) and bound \( \text{Var}(f') \). Compute \( \tilde{F}_{n_i}(f) \) and \( F_{n_i}(f) \).
Multi-step Automatic Algorithms, continued

Algorithm

Stage 2. Check the necessary condition for \( f \in C_{\tau_{a,b}} \). Compute

\[
\tau_{\text{min},n_i} = \frac{F_n(f)}{\tilde{F}_n(f) + (b-a)F_n(f)/(2n_i - 2)}.
\]

If \( \tau_{a,b} \geq \tau_{\text{min},n_i} \), then go to stage 3. Otherwise, set \( \tau_{a,b} = 2\tau_{\text{min},n_i} \). If \( n_i \geq (\tau + 1)/2 \), then go to stage 3. Otherwise, choose

\[
n_{i+1} = 1 + (n_i - 1) \left\lceil \frac{\tau_{a,b} + 1}{2n_i - 2} \right\rceil.
\]

Go to Stage 1.
Multi-step Automatic Algorithms, continued

Algorithm

Stage 3. Check for convergence. Check whether $n_i$ is large enough to satisfy the error tolerance, i.e.

$$\tilde{F}_{n_i}(f) \leq \frac{4\varepsilon(n_i - 1)(2n_i - 2 - \tau_{a,b}(b - a))}{\tau_{a,b}(b - a)^2}.$$ 

If this is true, then return $T_{n_i}(f)$ and terminate the algorithm. If this is not true, choose

$$n_{i+1} = 1 + (n_i - 1) \max \left\{ 2, \left\lfloor \frac{1}{(n_i - 1)} \sqrt{\frac{\tau_{a,b}(b - a)\tilde{F}_{n_i}(f)}{8\varepsilon}} \right\rfloor \right\}.$$ 

Go to Stage 1.
Upper Bound of Computational Cost

Our algorithm provides that

\[
\text{cost}(T, f; \varepsilon) \leq (b - a) \sqrt{\frac{\tau(b - a) \text{Var}(f')}{4\varepsilon}} + \tau + 4.
\]

Assume that \( f \) lies in the cone defined before:

\[
\frac{2}{b - a} \| f' - \frac{f(b) - f(a)}{b - a} \|_1 \leq \| f'' \|_1 \leq \frac{\tau}{b - a} \| f' - \frac{f(b) - f(a)}{b - a} \|_1.
\]

So we can use data-driven \( F_n(f) = \|(\text{linear spline of } f)' - \frac{f(b) - f(a)}{b - a}\|_1 \) to reliably bound \( \| f' - \frac{f(b) - f(a)}{b - a} \|_1 \), and then \( \| f'' \|_1 \):

\[
0 \leq \| f' - \frac{f(b) - f(a)}{b - a} \|_1 - F_n(f) \leq \frac{\| f'' \|_1}{2n - 2} \leq \frac{\tau \| f' - (f(b) - f(a))/(b - a) \|_1}{2n - 2},
\]

\[
\| f'' \|_1 \leq \tau \| f' - \frac{f(b) - f(a)}{b - a} \|_1 \leq \frac{\tau(b - a)F_n(f)}{2 - \tau/(n - 1)}.
\]
Then the $n$ needed to meet the desired tolerance is any one satisfying this data-driven criterion

$$\frac{\tau F_n(f)(b - a)}{4(n - 1)(2n - 2 - \tau)}.$$
Consider the family of bump test functions defined by

\[
\begin{align*}
f(x) &= \\
&= \begin{cases} \\
\beta [4\alpha^2 + (x - z)^2 - (x - z - \alpha)|x - z - \alpha|] \\
- (x - z + \alpha)|x - z + \alpha|], & \text{if } z - 2\alpha \leq x \leq z + 2\alpha, \\
0, & \text{otherwise.}
\end{cases}
\end{align*}
\]

(3)

with \(\log_{10}(\alpha) \sim \mathcal{U}[-4, -1]\), \(z \sim \mathcal{U}[2\alpha, 1 - 2\alpha]\), and \(\beta = 1/(4\alpha^3)\) chosen to make \(\int_0^1 f(x) \, dx = 1\). It follows that \(|f| \bar{F} = 1/\alpha\) and \(\operatorname{Var}(f') = 2/\alpha^2\). The probability that \(f \in \mathcal{C}_\tau\) is \(\min(1, \max(0, (\log_{10}(\tau/2) - 1) / 3))\).
Experiment Setup

As an experiment, we chose 10000 random test functions and applied Algorithm 1 with an error tolerance of $\varepsilon = 10^{-8}$ and initial $\tau$ values of 10, 100, 1000. The algorithm is considered successful for a particular $f$ if the exact and approximate integrals agree to within $\varepsilon$. The success and failure rates are given in Table 1. Our algorithm imposes a cost budget of $N_{\text{max}} = 10^7$. The probability that $f$ initially lies in $C_{\tau}$ is the smaller number in the third column of Table 1, while the larger number is the empirical probability that $f$ eventually lies in $C_{\tau}$ after possible increases in $\tau$ made by Stage 2 of Algorithm 1. For this experiment Algorithm 1 was successful for all $f$ that finally lie inside $C_{\tau}$, for which there was no warning. It was also successful for a small percentage of functions lying outside the cone.
## Results

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>$\text{Prob}(f \in C_\tau)$</th>
<th>Success</th>
<th>Failure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No Warning</td>
<td>Warning</td>
</tr>
<tr>
<td>10</td>
<td>0% $\rightarrow$ 25%</td>
<td>25%</td>
<td>&lt; 1%</td>
</tr>
<tr>
<td>Algorithm 1</td>
<td>23% $\rightarrow$ 58%</td>
<td>56%</td>
<td>2%</td>
</tr>
<tr>
<td>1000</td>
<td>57% $\rightarrow$ 88%</td>
<td>68%</td>
<td>20%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Success</th>
<th>Failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>quad</td>
<td>8%</td>
<td>92%</td>
</tr>
<tr>
<td>integral</td>
<td>19%</td>
<td>81%</td>
</tr>
<tr>
<td>chebfun</td>
<td>29%</td>
<td>71%</td>
</tr>
</tbody>
</table>

**Table:** The probability of the test function lying in the cone for the original and eventual values of $\tau$ and the empirical success rate of Algorithm 1 plus the success rates of other common quadrature algorithms.
Guaranteed Automatic Integration Library (GAIL)

The ideas presented here are being implemented in MATLAB code (code.google.com/p/gail), which also include:

- Automatic univariate function recovery via linear splines.
- Guaranteed automatic Monte Carlo algorithm for multidimensional integration.
- Guaranteed automatic quasi-Monte Carlo algorithm for multidimensional integration.
- And more.
Future Work

- Guaranteed automatic algorithms with higher order convergence rate.
- Locally adaptive algorithms.
- Relative Error.
Now we compute the lower bound by constructing fooling functions. We choose the triangle shaped function \( f_0 : x \mapsto 1/2 - |1/2 - x| \). Then

\[
|f_0|_\mathcal{F} = \|f'_0 - f_0(1) + f_0(0)\|_1 = \int_0^1 |\text{sign}(1/2 - x)| \, dx = 1,
\]

\[
|f_0|_\mathcal{F} = \text{Var}(f'_0) = 2 = \tau_{\text{min}}.
\]
For any $n \in \mathcal{J} := \mathbb{N}_0$, suppose that the one has the data $L_i(f) = f(\xi_i)$, $i = 1, \ldots, n$ for arbitrary $\xi_i$, where $0 = \xi_0 \leq \xi_1 < \cdots < \xi_n \leq \xi_{n+1} = 1$. There must be some $j = 0, \ldots, n$ such that $\xi_{j+1} - \xi_j \geq 1/(n + 1)$. The function $f_1$ is defined as a triangle function on the interval $[\xi_j, \xi_{j+1}]$:

$$
f_1(x) := \begin{cases} 
\frac{\xi_{j+1} - \xi_j - |\xi_{j+1} + \xi_j - 2x|}{8} & \xi_j \leq x \leq \xi_{j+1}, \\
0 & \text{otherwise.}
\end{cases}
$$
Lower Bound of Computational Cost, continued

This is a piecewise linear function whose derivative changes from 0 to \(1/4\) to \(-1/4\) to 0 provided \(0 < \xi_j < \xi_{j+1} < 1\), and so \(|f_1|_{\mathcal{F}} = \text{Var}(f'_1) \leq 1\).

Moreover,

\[
\text{INT}(f) = \int_0^1 f_1(x) \, dx = \frac{(\xi_{j+1} - \xi_j)^2}{16} \geq \frac{1}{16(n + 1)^2} =: g(n),
\]

\[
g^{-1}(\varepsilon) = \left\lceil \sqrt{\frac{1}{16\varepsilon}} \right\rceil - 1.
\]

Using these choices of \(f_0\) and \(f_1\), along with the corresponding \(g\) above, we can have that the complexity of the integration problem over the cone of functions \(C_\tau\) is bounded below as

\[
\text{comp}(\varepsilon, \mathcal{A}(C_\tau, \mathbb{R}, \text{INT}, \Lambda^{\text{std}}), \mathcal{B}_s) \geq \left\lceil \sqrt{\frac{(\tau - 2)s}{32\tau\varepsilon}} \right\rceil - 1.
\]