A Guaranteed, Adaptive, Automatic Algorithm for Univariate Function Minimization

Xin Tong
Department of Mathematics, Statistics and Computer Science
University of Illinois at Chicago
xtong20@uic.edu
Joint work with Prof **Fred J. Hickernell** and Prof **Sou-Cheng Choi**
Department of Applied Mathematics, Illinois Institute of Technology

Motivation

Construct an adaptive algorithm for solving global univariate function minimization problem, which is guaranteed to provide an answer to within a user-specified tolerance.

For example, MATLAB's `fminbnd` uses the golden section search method. Unfortunately, `fminbnd` might give only a local minimum, and it has no guarantee of meeting the tolerance.

Problem Definition

Let $S(f) := \min_{a \le x \le b} f(x) = f(x^*)$. Given tolerances ε and δ, find approximate minimum value $U(f)$ and possible solution set $\mathcal{X}(f)$ such that

$$U(f) - S(f) \le \varepsilon \quad \textbf{or} \quad x^* \in \mathcal{X},\ \mathrm{Vol}(\mathcal{X}(f)) \le \delta$$

for $f \in \mathcal{C}_\tau := \left\{ f \in C^1[a,b] : \|f''\|_\infty \le \frac{\tau}{b-a} \left\| f' - \frac{f(b)-f(a)}{b-a} \right\|_\infty \right\}$,
i.e., f is not too *spiky*.

Data-Based Approximations

Our algorithm is based on the following approximations computed in terms of function values:

$$A_n(f)(x) := \text{linear spline of } f \text{ at } x_i = (i-1)\frac{b-a}{n-1}, \quad i = 1, \ldots, n,$$

$$\widetilde{F}_n(f) := \left\| A_n(f)' - \frac{f(b)-f(a)}{b-a} \right\|_\infty \approx \left\| f' - \frac{f(b)-f(a)}{b-a} \right\|_\infty.$$

Then for $f \in \mathcal{C}_\tau$ we have the data-based:

$$\|f''\|_\infty \le \frac{\tau}{b-a} \mathfrak{C}_n \widetilde{F}_n(f), \qquad \mathfrak{C}_n = \frac{1}{1 - \tau/(2n-2)}.$$

The bounds on f and $\min_{a \le x \le b} f(x)$

The difference between $f(x)$ and its linear spline approximation is bounded by

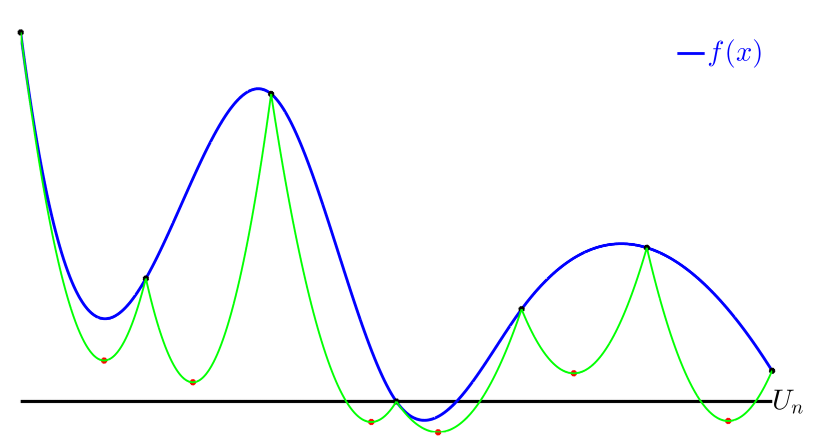
$$|f(x) - A_n(f)(x)| \le \frac{\tau}{b-a} \mathfrak{C}_n \widetilde{F}_n(f) \frac{(x-x_i)(x_{i+1}-x)}{2}.$$

Then the lower bound for $f(x)$ is a piecewise quadratic function:

$$A_n(f)(x) - \frac{\tau \mathfrak{C}_n \widetilde{F}_n(f)}{b-a} \frac{(x-x_i)(x_{i+1}-x)}{2} \le f(x) \quad \text{for all } x \in [x_i, x_{i+1}].$$

The upper bound on $\min_{a \le x \le b} f(x)$ is given by

$$U_n := \min_{a \le x \le b} A_n(f)(x) = \min_{1 \le i \le n} f(x_i) \ge \min_{a \le x \le b} f(x).$$



Algorithm funmin_g

Initialize sample size n.
Stage 1. Estimate the semi-norms of f that define $\mathcal{C}_\tau$ using sample size n.
Stage 2. Check the condition for $f \in \mathcal{C}_\tau$.
Stage 3. Compute the estimated error $\mathrm{err}_n(f)$ and the possible solution set $\mathcal{X}_n(f)$.
If n is large enough to satisfy either the error tolerance or the X tolerance, i.e.,

$$\mathrm{err}_n(f) \le \varepsilon \quad \text{or} \quad \mathrm{Vol}(\mathcal{X}_n(f)) \le \delta,$$

return the approximations U_n and $\mathcal{X}_n(f)$. If not, increase n and go to **Stage 1**.

Numerical results

- A family of bump test functions

$$f(x) = \begin{cases} \frac{1}{2a^2}[-4a^2 - (x-z)^2 - (x-z-a)|x-z-a| + (x-z+a)|x-z+a|] & \text{if } |x-z| \le 2a \\ 0 & \text{otherwise} \end{cases}$$

with $\log_{10}(a) \sim \mathcal{U}[-4,-1]$ and $z \sim \mathcal{U}[2a, 1-2a]$. The table shows the empirical success rates with $\varepsilon = 10^{-8}$, $\delta = 10^{-6}$, and sample $n = 10000$.

τ	Prob($f \in \mathcal{C}_\tau$)	Success No Warning	Success Warning	Failure No Warning	Failure Warning
11	1.50% → 21.32%	21.32%	0.00%	78.68%	0.00%
101	33.28% → 53.36%	52.38%	0.00%	47.62%	0.00%
1001	66.98% → 85.37%	85.39%	0.00%	14.61%	0.00%

- Functions with two local minimum points

$$f(x) = -5\exp(-[10(x-a_1)]^2) - \exp(-[10(x-a_2)]^2), \quad 0 \le x \le 1,$$

with $a_1, a_2 \sim \mathcal{U}[0,1]$. The table shows the success rates of our algorithm compared to **fminbnd**.

	funmin_g			**fminbnd**
δ	Success	Success No Warning	Success Warning	Success
10^{-2}	100.00%	100.00%	0.00%	68.36%
10^{-4}	100.00%	100.00%	0.00%	68.36%
10^{-7}	100.00%	0.00%	100.00%	68.36%

Further Work

We are writing a paper with the following additional topics:

- **Computational cost.** Find the theoretical lower and upper bounds for the cost;
- **Experimental cost bounds.** Numerical results of the lower and upper bounds for the cost.

I would like to especially thank my advisors Fred J. Hickernell and Sou-Cheng Choi, together with my coworkers Yuhan Ding, Lan Jiang, Xuan Zhou, Yizhi Zhang, and Lluís Antoni Jiménez Rugama.

References

[1] S.-C. T. Choi, Y. Ding, F. J. Hickernell, L. Jiang, L. A. Jiménez Rugama, X. Tong, Y. Zhang, and X. Zhou, *GAIL: Guaranteed Automatic Integration Library (Version 2.1), MATLAB Software*, 2015.

[2] N. Clancy, Y. Ding, C. Hamilton, F. J. Hickernell, and Y. Zhang, *The cost of deterministic, adaptive, automatic algorithms: Cones, not balls*, Journal of Complexity, 30 (2014), pp. 21–45.

[3] X. Tong, *A guaranteed, adaptive, automatic algorithm for univariate function minimization*, Master's thesis, Illinois Institute of Technology, 2014.