SURF-2023-0041

Adaptive Numerical Algorithms: Convergence and Efficiency

Hui Zhang, Jiaotong-Liverpool University

Zhenpeng.Liu20, Wanqian.Chen20

SURF Summer Undergraduate Research Fellowship

Abstract

The quadrature problem for a given function f(x) is to estimate the value of

$$If = \int_0^1 f(x)dx$$

Traditional quadrature formulas give estimates of the form:

$$Q_N f = \sum_{i=1}^{N} w_i f(x_i)$$

Adaptive quadrature is a numerical integration method that approximates the integral of a function f(x) using refined subintervals. It is efficient for both "**well behaved**" and "**badly behaved**" integrands, unlike traditional algorithms that may fail with the latter.

We have made **slight improvements** to the implementation of the algorithm in MATLAB and attempted to discuss its **bounded characteristics**.

Raw implementation

```
Algorithm 1 Raw adaptive simpson algorithm
Input endpoints a,b; tolerance TOL; limit N to number of levels.
Output approximation APP which is the integrate result.
function ADAPSIMPSON(f,a,b,tol)
    c = (a + b)/2
    c1 = (a + c)/2
    c2 = (a + c)/2
    Sab = 1/6 * (b − a) * (f(a) + 4 * f(c) + f(b))
    Sac = 1/6 * (c − a) * (f(a) + 4 * f(c1) + f(c))
    Scb = 1/6 * (b − c) * (f(c) + 4 * f(c2) + f(b))
    e = abs(Sab − Sac − Scb)/15
    if e < tol then
        q = Sac + Scb; x = [a; c1; c; c2; b]
        return
    else
        [qac, eac, xac] = ADAPSIMPSON(f, a, c, tol/2)
        [qcb, ecb, xcb] = ADAPSIMPSON(f, c, b, tol/2)
        q = qac + qcb
        e = eac + ecb
        x = [xac(1 : (end − 1)); xcb]
    end if
end function
```

Note: This implementation is based on iteration and may not provide the highest computational efficiency.

Example

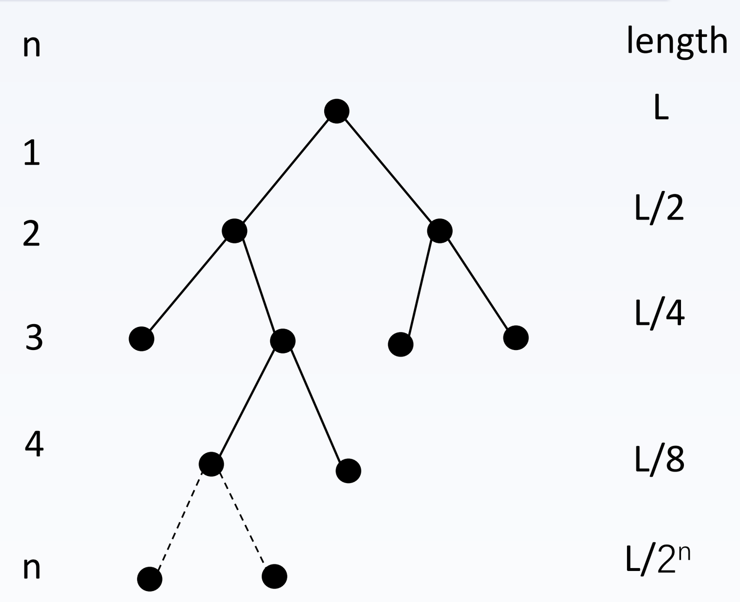
Consider the integration: $\int_{eps}^{1} \frac{1}{\sqrt{x}}dx$ It follow the assumption 1 and 2 [1] by :

$$f(x) = x^{-0.5}, w(x) = x, k = 16, \alpha = -0.5, p = 4$$

- $f^{(4)}(x) = \frac{105}{16}x^{-\frac{9}{2}} \le 16x^{-0.5-4} = 10x^{-\frac{9}{2}}$
- $E(x,k) \le \left|\frac{h^5}{90}f^{(4)}(\xi)\right| \le \frac{7}{96}\cdot 2^{-k-1}\cdot x^{-\frac{9}{2}} \le 105\cdot 2^{-64}\cdot x^{-\frac{9}{2}}$

for enough large K

Finite iteration



M' is the intervals has been divided;
M contains a distinguished interval I*.

$\gamma, \beta < 1, \epsilon > 0$

$$2 + \sum_{k=1}^{d_0-1} 2^{d_k} \le 2 + \sum_{k=1}^{d_0-1} 2^{\left[\frac{\log_2 \frac{\epsilon}{\beta^k \eta_0}}{\log_2 \gamma}\right]}$$

$$\le 2 + \sum_{k=1}^{d_0-1}\left[\frac{\epsilon}{\beta^k \eta_0}\right]^{\frac{1}{\log_2\gamma}} \le 2 + \frac{\epsilon^{\frac{1}{\log_2\gamma}}}{\eta_0}\sum_{k=1}^{\infty}\beta^{\frac{-k}{\log_2\gamma}}$$

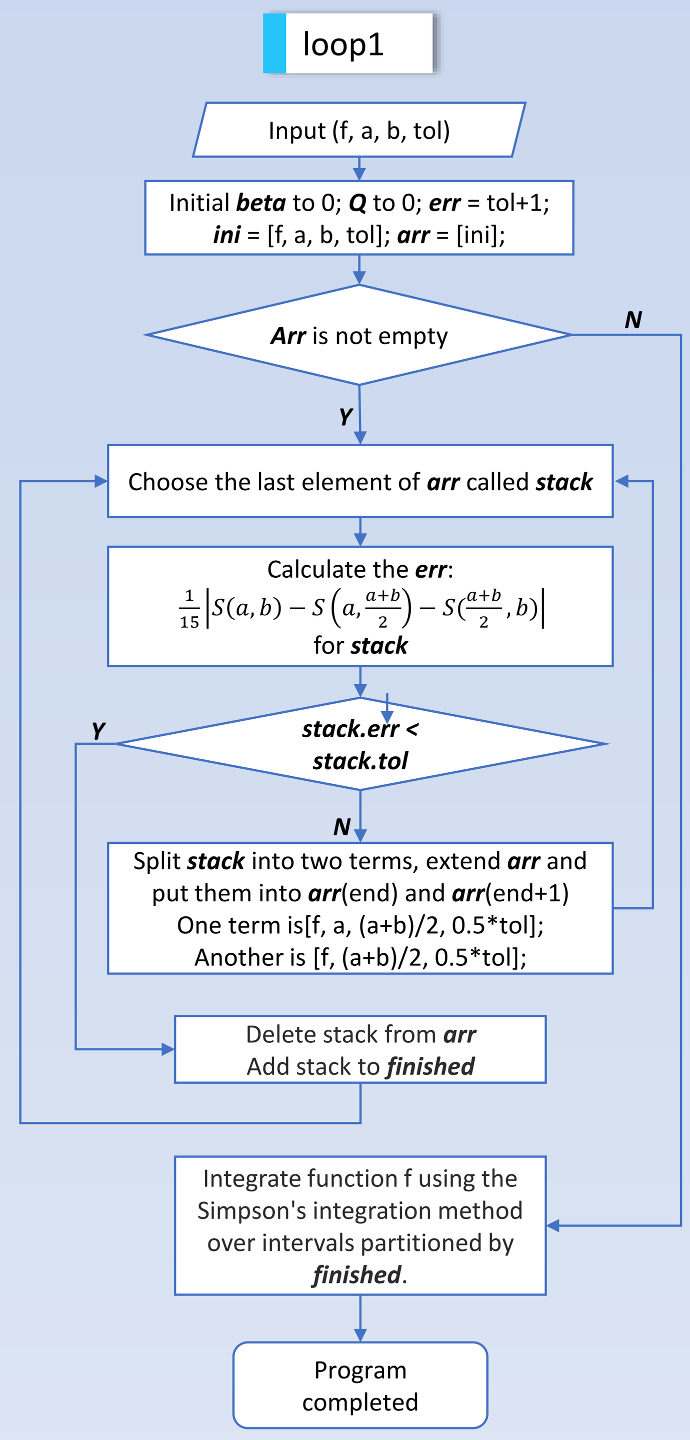
The geometric series in **convergent** since $\gamma, \beta < 1$
And the bounded is:

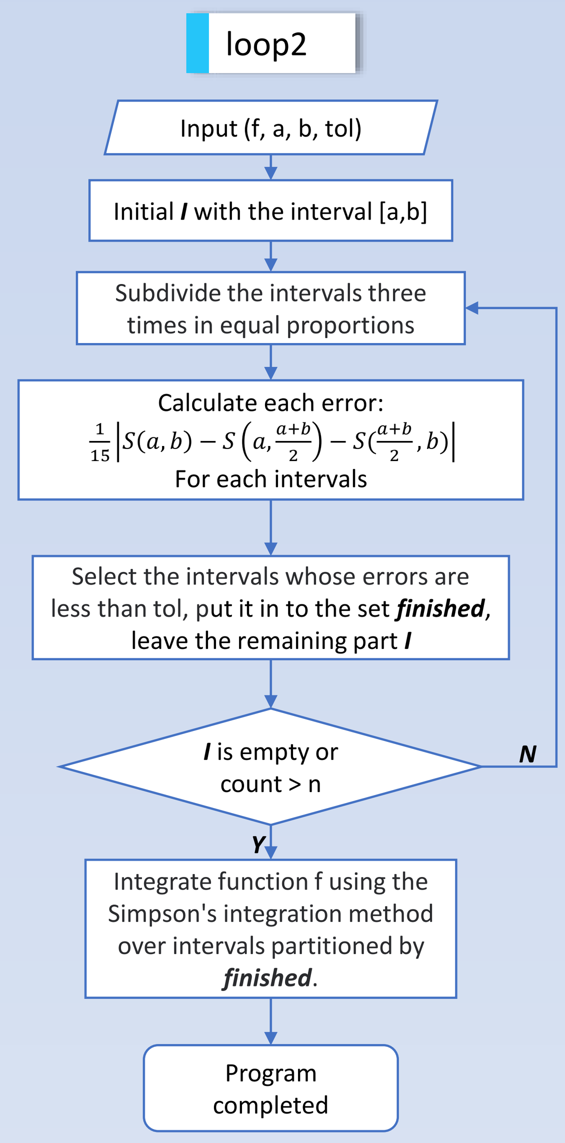
$$2 + \frac{\epsilon^{\frac{1}{\log_2\gamma}}}{\eta_0}\left[1 - \beta^{\frac{-k}{\log_2\gamma}}\right] = o(\epsilon^{1/\log_2\gamma})$$

Further research

- Using parallel calculation to increase the integration speed.
- Find a smaller bound for adaptive Simpson. And calculate its convergence rate.

Improved implementations

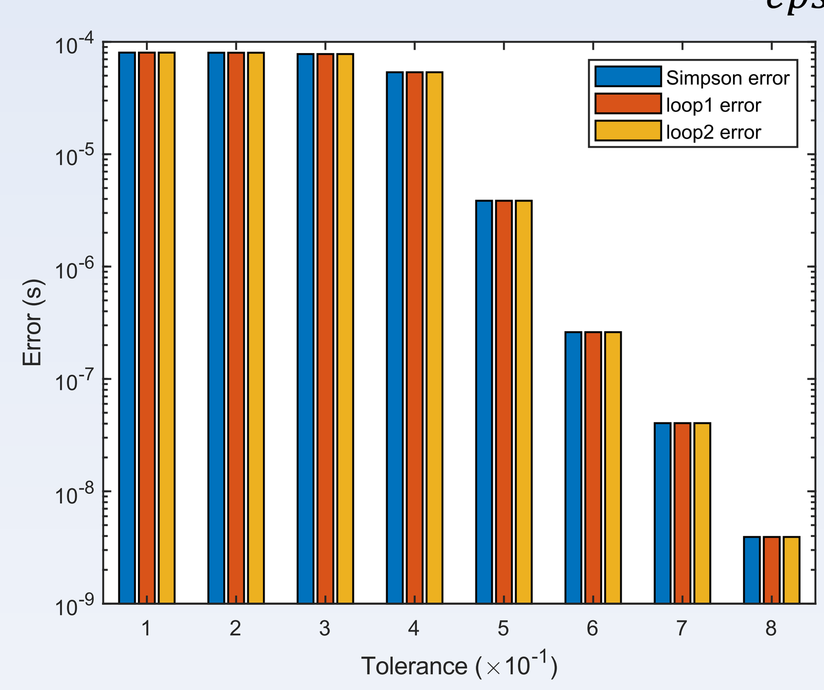


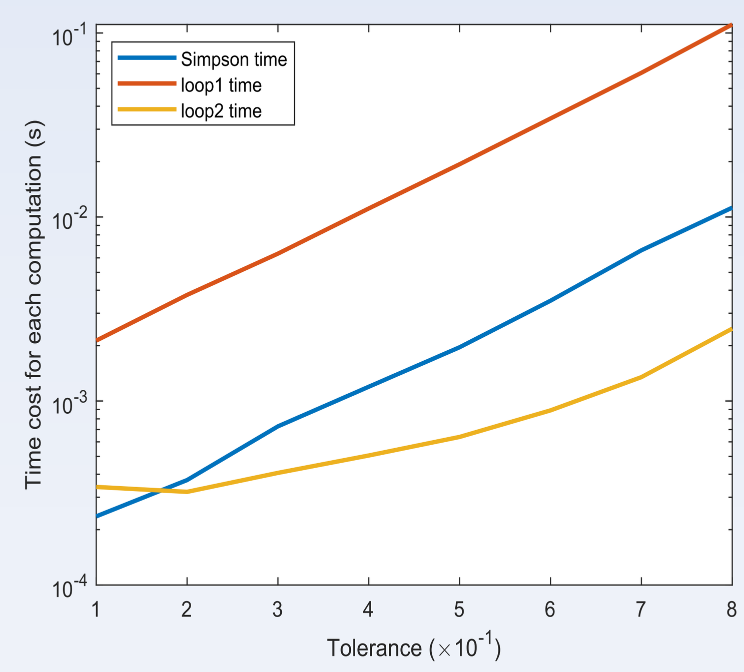


Note: The biggest difference between loop1 and loop2 is that loop2 can be almost fully vectorized in supported languages (e.g., MATLAB), whereas loop1 is more difficult to vectorize.

Comparison

The calculation for the example: $\int_{eps}^{1} \frac{1}{\sqrt{x}}dx$





It is obvious they will have the same error because of the same algorithm. But the **time** taken for the same content varies. While their time complexity seems to be equivalent, loop2 is significantly faster.

Conclusion

Theoretical Insights and Convergence

- Delving deep into numerical algorithms, our study meticulously examines the **convergence rate**, subsequently delving into the exploration of **convergence termination criteria**. Through theoretical analysis, we present pragmatic solutions for real-world computational challenges.

Empirical Validation and Exemplification

- We substantiate our findings through **concrete examples**, demonstrating the efficacy of our proposed methodologies. These instances not only comprise theoretical analysis but also encompass tangible solutions to real-world problems, establishing the **versatility and robustness** of our approaches.

Innovations in Adaptive Simpson's Algorithm

- Propelling beyond conventional boundaries, we have two enhancements to the traditional implement of **adaptive Simpson's algorithm**. By applying our innovations to real-world scenarios, we not only drastically amplify computational speed though uphold the algorithm's original **time complexity**.

Our research propels numerical algorithms towards greater horizons!

We sincerely appreciate your thoughtful engagement!

References: 1. John R. Rice. 1975. A Metalgorithm for Adaptive Quadrature. J. ACM 22, 1 (Jan. 1975), 61–82. https://doi.org/10.1145/321864.321870.
2. Agarwal, S., Povolotskyi, M., Kubis, T. et al. Adaptive quadrature for sharply spiked integrands. J Comput Electron 9, 252–255 (2010). https://doi.org/10.1007/s10825-010-0338-3.