Richard Abele January 21, 2025

Numerical Methods - Problem Set 7

Exercise 4. Application of Numerical Integration in Quantum Mechanics

In `problem 4` we are asked to implement a program that will integrate a function using the Newton-Cotes formula for a second degree polynomial, i.e., Simpson's $\frac{1}{3}$ Rule. Additionally, we are asked to calculate the error.

(a) Integrate the function $f(x) = x + x^3$ in the interval $[0, 1]$ with $n = 5$ data points. Discuss why the answers is exact (excluding rounding errors).

(b) Suppose a particle in a potential well as a wavefunction of the form

$$\Psi(x,t) = \begin{cases} \frac{1}{\sqrt{L}}\left[\sin\left(\frac{\pi x}{L}\right)e^{-i\omega_1 t} + \sin\left(\frac{2\pi x}{L}\right)e^{-i\omega_2 t}\right] & 0 < x < L \\ 0 & \text{otherwise.} \end{cases}$$

The probability distribution of the particle $P(x,t) \equiv |\Psi(x,t)|^2$. Find the probability that the particle is in the far right quarter of the potential well, given by

$$P(t) = \int_{\frac{3L}{4}}^{L} P(x,t)\,dx,$$

for the time $t \in \left\{0, \frac{\pi}{\Delta\omega}\right\}$, with $\Delta\omega = \omega_2 - \omega_1$. Assume that $L = 2, \omega_1 = 3$, and $\omega_2 = 4.5$. Compare the results for all odd data numbers n on the interval $n \in [5, 501]$. For each n print: n, $\log(E)$, and $\log(h)$. Plot $\log(E)$ vs. $\log(h)$ for both times. Are the plots compatible with the expected convergence rate?
Hint: The error for the 2nd order Newton-Coted integration scheme is given by

$$E = \frac{1}{90}h^4 f^{(4)}(x_1)$$

Solution:

(a) The given integral has an exact analytical solution as follows:

$$I = \int_0^1 x + x^3\,dx = \frac{3}{4}.$$

Navigating to the directory `problem4` and running `./qmapp newton fx1 0 1 5` will calculate the integral of the given function over the interval $[0, 1]$ using five points with Newton-Cotes integration for second degree polynomials. The calculated result of $I = 0.75, E = 0.0$ matches exactly with the expected analytical result.
The reason for this exact match is due to the specific nature of the function - integration through Simpson's $\frac{1}{3}$ rule will yield exact answers for all polynomials of degree ≤ 3.

(b) Implementing and applying Simpson's $\frac{1}{3}$ Rule in Mathematica as directed yields the following plot for both values of t (see Mathematica notebook file for more details): As can be seen in the plot, the step size aligns very nicely with the size of the error

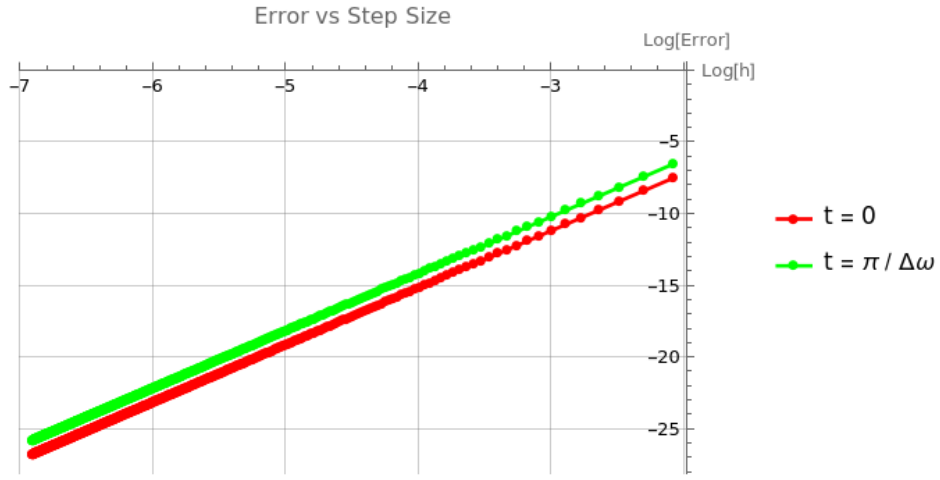

Figure 1:

over many orders of magnitude (as indicated by the logorithm).

Exercise 5. Application in Mode Decomposition

Create a function to numerically integrate a data-based integrand via two of the following methods:

(a) Trapezoid Rule

(b) 2nd order Newton-Cotes

(c) Simpson's $\frac{3}{8}$ Rule

(d) Euler-Maclaurin Method

(e) Gauss-Legendre Methods of Orders 2, 4, and 8

(f) Splines

Test the functionality of the integration program on the following functions over the pre-defined intervals:

$$f_1(x) = e^x \cos(x), \qquad x \in \left[0, \frac{\pi}{2}\right]$$
$$f_2(x) = e^x, \qquad x \in [-1, 3]$$
$$f_3(x) = \begin{cases} e^{2x} & x < 0 \\ x - 2\cos(x) + 4 & x \geq 0 \end{cases}, \qquad x \in [-1, 1]$$

Plot log(Error) vs log(h) for the three functions using $n \in [5, 501]$ points.

Solution: 5

The first part of this exercise involves developing a program to numerically solve an `integral` struct. The resulting log(Error) should then be compared to the known analytical results with a dependence on the log of the stepsize h. The second part of the exercise is to be addressed in a later assignment.

To solve this task a c program was written and compiled using `make`. It was then run using `./integrate.out <function number>` with `<function number>` one through three.

For the methods of integration it was decided to use the trapezoid rule and Simpson's $\frac{3}{8}$ rule (SR). The trapezoid rule can be used for any number of points without modification, but the SR required dividing the dataset into sub intervals with 4 points on which SR could be used. It is worth noting that there were in practice $\frac{\text{numPoints } -1}{3}$ sub intervals (rounded down to the nearest whole number). This division into intervals of three is due to the reuse of the last point in an interval as the first point in the next interval.

Points that did not fit into these subintervals were then integrated by defining a new `integal` struct and integrating using the trapezoid rule. This resulted in a better approximation than using the only the trapezoid rule, but there was a substatntial loss of accuracy for results that did not fit exactly into the SR subintervals.

The exported data were then plotted using a GNU Plot script.

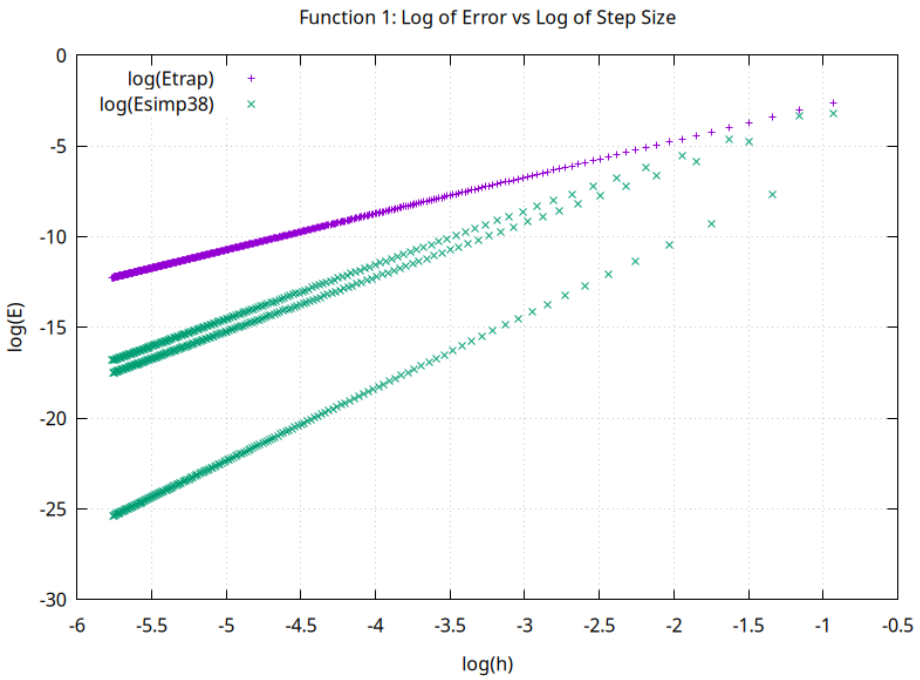

Figure 2: Function 1 Plot

As expected, the plots for the SR were usually more accurate, even for the same step size. Results were however inconsistent due to the flucuations in the numbers of steps and the application of the trapezoid rule.

Though the error of the trapezoid rule seems to consistently decline with smaller step sizes, it is worth noting that it also can flucuate for functions with incontinuities (like function 3).

Higher accuracies could be achieved by using an integration method appropraite to the number of points. Function 1, for example, is well-behaved function that could benefit from numerical integration methods that require more data (like Boole's rule) if the step size is known. The same could be said for function 2.

Function 3, on the other hand, requires careful consideration of the incontinuity at $x = 0$, requiring there to be a division into two areas of integration for the highest accuracy possible.

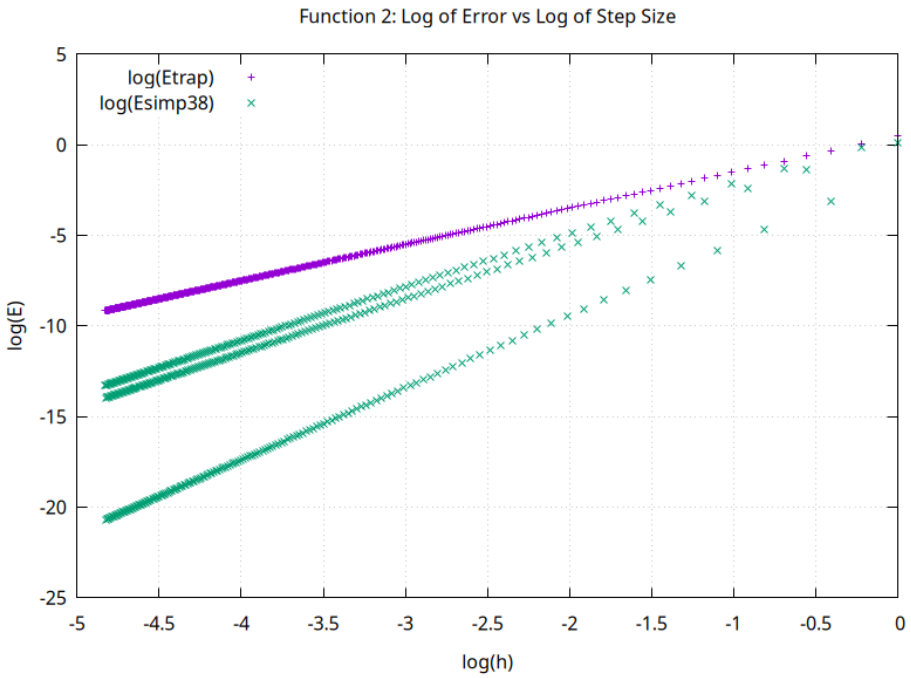

Figure 3: Function 2 Plot

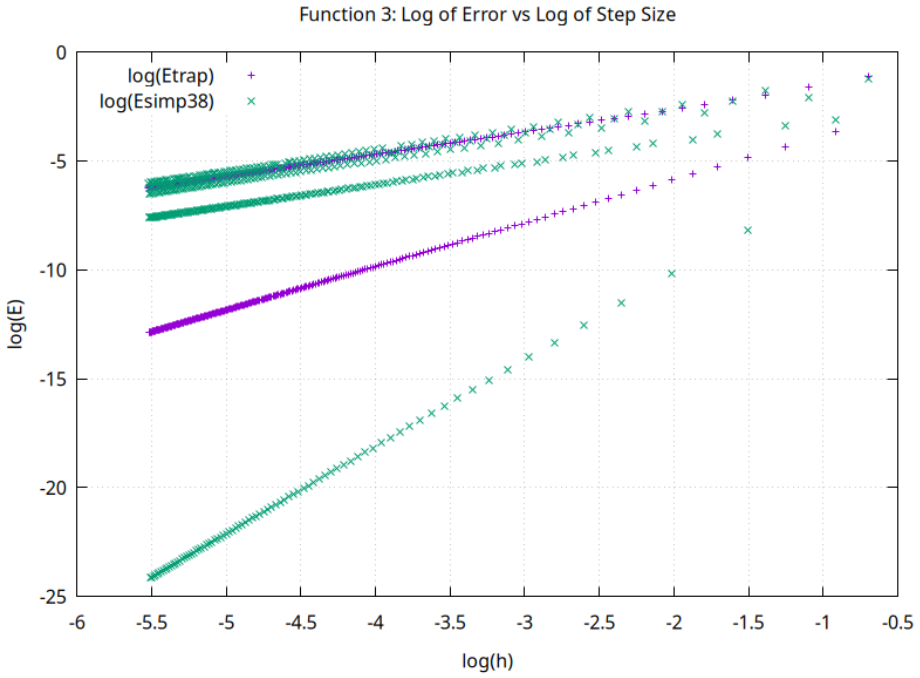

Figure 4: Function 3 Plot