Numerical Integration Techniques

Zane Ali
https://github.com/zaneali1/

Keywords: Trapezium, Simpson's, QUADPACK

1. Introduction

To investigate numerical integration techniques, the Trapezium Rule, Simpson's rule and a QUADPACK technique (from the Fortran Library) were used to evaluate the definite integral shown in Equation 1. The integrand is also shown graphically in Fig. 1, between 0 and π.

$$I = \int_0^2 e^{-x} sin(x) dx \tag{1}$$

Evaluating the definite integral analytically provided the result shown in Equation 2. Python was used to perform each of the numerical integration techniques computationally against the definite integral. The relative errors of each approximation obtained with the Trapezium (E_T) and Simpson's Rule (E_S) were calculated against the analytic value.

$$I = \frac{1}{2} - \frac{sin(2) + cos(2)}{2e^2} \approx 0.466629662593176... \tag{2}$$

Once obtained, the values of E_T and E_S were found to vary with the number of subintervals, n, used for each approximation (see: Theory). Plots for each were produced (using a $\log_{10}$ - $\log_{10}$ scale) and then compared to theoretical predictions.

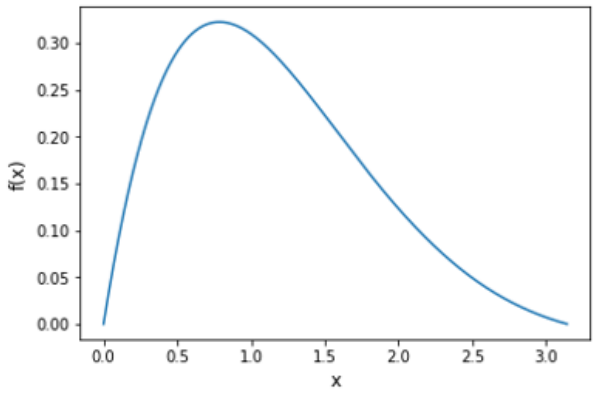

Figure 1: Plot of the function $e^{-x}sin(x)$ between 0 and π, which is to be to be integrated using different numerical integration techniques between the limits 0 and 2.

2. Theory

To evaluate a definite integral, the Trapezium Rule can be used to approximate the area under a graph $f(x)$ as a set of trapezia. It follows that

$$\int_a^b f(x) = \frac{\Delta x}{2}(f(x_0) + 2f(x_1) + ... + 2f(x_{n-1}) + f(x_n)), \tag{3}$$

where $f(x)$ is the function to be integrated, Δx is the width of each trapezium, and a and b are the limits of the integral. The number of of subintervals (or trapezia) n can therefore be can therefore be related to the width of each subinterval from equation 4 (which also holds true for Simpson's Rule).

$$n = \frac{(b-a)}{\Delta x} \tag{4}$$

The error of the approximations obtained using the Trapezium Rule is shown in equation 5. [1]

$$E_T = \frac{(b-a)^3}{12n^2} f^{(2)}(\xi), \tag{5}$$

where a value for ξ exists somewhere between a and b. To obtain the upper-bound estimate of the error, the value of ξ will be selected to maximise $f^{(2)}(\xi)$.

Simpson's Rule approximates the area under a graph $f(x)$ using parabolic

curves instead of straight line segments as with the Trapezium Rule. Simpson's Rule is shown in Equation 6.

$$\int_a^b f(x) = \frac{\Delta x}{2}(f(x_0) + 4f(x_1) + 2f(x_2) + ... + 4f(x_{n-1}) + f(x_n)) \qquad (6)$$

The error of approximations obtained using Simpson's Rule is shown analytically in equation 7. [1]

$$E_S = \frac{(b-a)^5}{180n^4} f^{(4)}(\xi), \qquad (7)$$

where, once again, a number ξ exists somewhere between a and b. To obtain the upper-bound estimate of the error, the value of ξ will be selected to maximise $f^{(4)}(\xi)$.

Both the Trapezium and Simpson's rule operate by splitting the interval to be integrated (between the limits a and b) into subintervals of equal width, Δx. To improve the efficiency of the numerical integration, it is often useful to select subintervals of different lengths. The QUADPACK technique (from the FORTRAN library) evaluates the slope and flatness of the integrand to select values for the step sizes to be used. For finite integration limits, the integration is performed using a Clenshaw-Curtis method. The integration can be performed using Python via the Scipy.integrate.quad subpackage.

3. Results and Discussion

The values of the approximations obtained by the Trapezium and Simpson's rule were found to increase in accuracy when the number of subintervals were increased. Plots for the relative errors E_T and E_S against n (using a $\log_{10}$ - $\log_{10}$ scale) are shown in figures 2 and 3 respectively.

From figures 2 and 3, it was found that beyond values of $\log_{10}(E_{T,S})$ of approximately -13 linearity began to break down. This was due to limitations with floating point arithmetic in Python; to improve on the investigation, the `long` datatype could be used instead to store values. Values which were not affected by this limitation were fit to a line of best fit in the form $y = c_{T,S}x + d$.

Equations 5 and 7 suggested values of c_T = - 2 and c_S = - 4 for each curve.

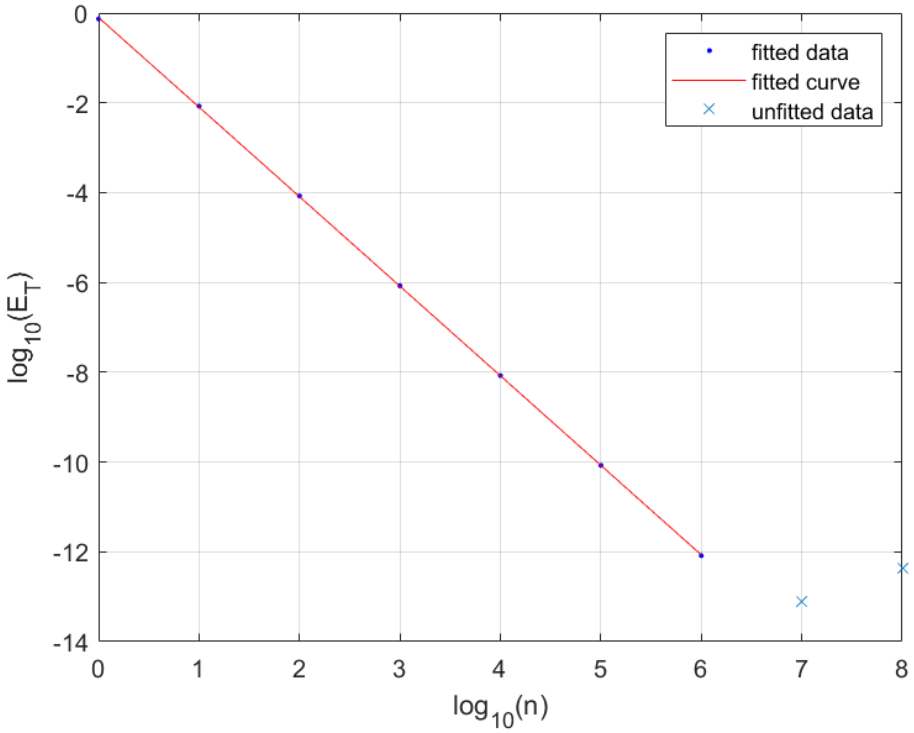

Figure 2: Plot of the relative error against the number of subintervals for the Trapezium Rule using a $\log_{10}$ - $\log_{10}$ scale

As highlighted below, the values obtained were within the range of statistical error for the theoretically predicted values.

$$\boxed{c_T = -1.995 \pm 0.010}$$

$$\boxed{c_S = -4.11 \pm 0.26}$$

The value obtained from the QUADPACK technique was found to be a relative error of 5.18 x 10^{-15}, which was much smaller than what could be obtained using the Trapezium or Simpson's Rule. The technique was also the most computationally efficient, having performed only 21 function calls to obtain this result.

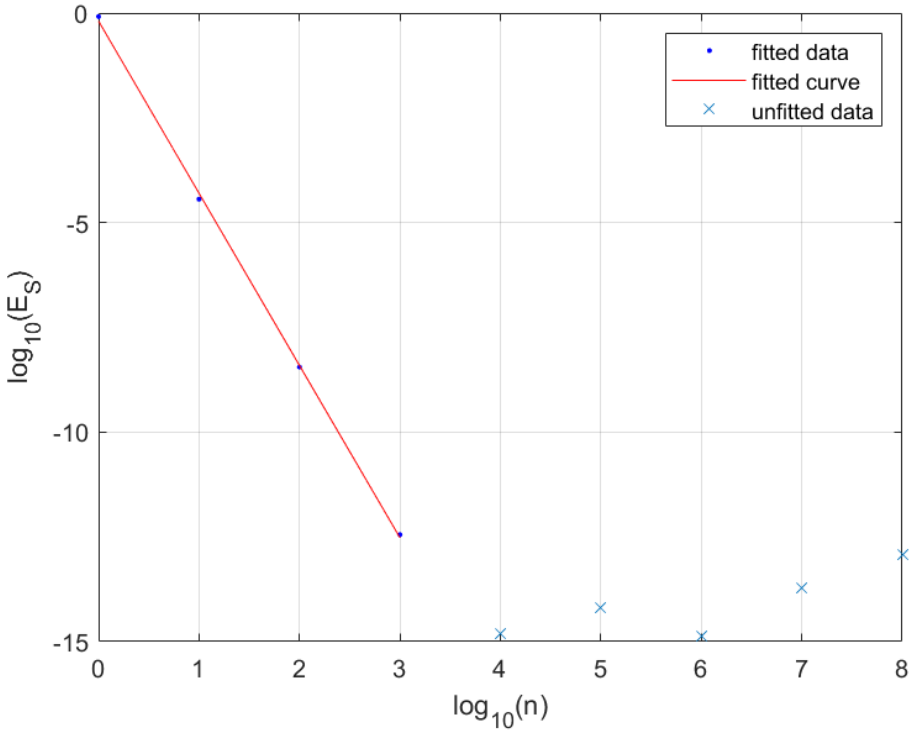

Figure 3: Plot of the relative error against the number of subintervals for the Simpson's Rule using a $\log_{10}$ - $\log_{10}$ scale

4. Conclusion

The Trapezium and Simpson's rule numerical integration techniques were tested against a definite integral using computational techniques. The relative error for each was obtained, and was found to vary with n as predicted by theory. Overall, Simpson's Rule was found to perform better than the Trapezium Rule, approaching the limit of accuracy with fewer function calls. This limit was caused by limitations in floating point arithmetic in Python which led some data to remain unfitted.

References

[1] Hoover, Dorothy M. (1955) "Estimates of Error in Numerical Integration," Journal of the Arkansas Academy of Science: Vol. 8 , Article 23.