UNIVERSITY OF BIRMINGHAM

Numerical Integration

Mike Knee

Issue: 1

Date: November 10, 2016

School of Physics and Astronomy
University of Birmingham
Birmingham, B15 2TT

Contents

1 Introduction

In this report the use of computer programs to solve integrals numerically will be discussed. First the simple algorithms of the trapezium rule and Simpson's method are covered. The two methods will be compared and their performance at high intervals discussed. Following that is the use of the Gnu Scientific Library (GSL) routines to perform integration estimates. The performance of these routines in different circumstances will be discussed.

2 Trapezium and Simpson's Method

Two numerical methods, the trapezium method and then Simpson's rule, were used to evaluate the integral

$$\int_0^2 e^{-x} \sin x dx$$

This integral can be evaluated analytically. Which was done first in order to have some reference value which the results were compared to. Integration by parts was used to solve this integral

$$I = \int_0^2 e^{-x} \sin x dx$$

$$I = \left[-e^{-x} \cos x\right]_0^2 + \int_0^2 e^{-x} \cos x dx$$

applying integration by parts again

$$I = \left[-e^{-x} \cos x\right]_0^2 + \left[-e^{-x} \sin x\right]_0^2 - \int_0^2 e^{-x} \sin x dx$$

$$I = \left[-e^{-x} \cos x\right]_0^2 + \left[-e^{-x} \sin x\right]_0^2 - I$$

$$2I = \left[-e^{-x} \cos x\right]_0^2 + \left[-e^{-x} \sin x\right]_0^2$$

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

Which gives the value of the integral.

Figure 1 shows a plot of $e^{-x} \sin x$ against x.

2.1 Trapezium Method

The program computes a numerical estimate for the integral using the Trapezium method by finding the area of each slice, approximated as

$$A_s = \frac{1}{2}(f(a) + f(b))h \tag{2}$$

Where f is the function we are integrating, h is the width of the slice and a and b are the upper and lower bounds of the slice. See [2] for details. The sum of these slices is then provided as the final result. The absolute error is calculated by taking the absolute difference between the esitmated result and the analytically derived answer, using equation 1. The program uses double variables to store the approximate area of each slice, the sum of all slices (i.e. the estimated result) and the analytic answer.

The error in the trapezium method result can be derived by taking the Newton-Cotes formula from [4] and noticing that the error for one slice is then given by the second part, ie.

$$\epsilon_s = \frac{1}{12} h^3 f''(\eta)$$

Where h is the width of the slice, and η maximises the value of f''. From this it is clear that the error for a single slice is proportional to the width of the slice cubed

$$\epsilon_s \propto h^3$$

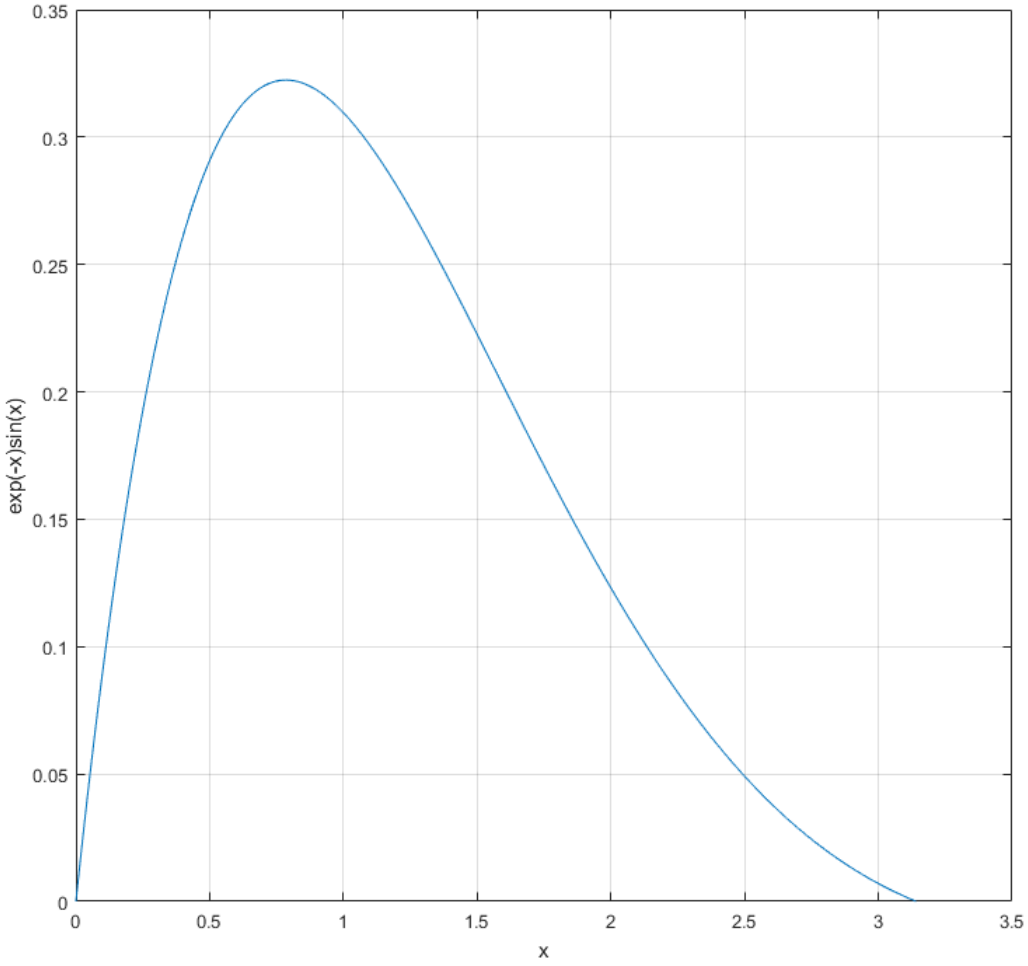

Figure 1: Plot of $f(x) = e^{-x} \sin x$, which we want to find the integral of.

So the error for the complete integral estimation will be n (the number of slices) times this

$$\epsilon_T \propto h^3 n$$

The width of each slice is given by

$$h = \frac{u - l}{n}$$

where u and l are the upper and lower bounds respectively. From this the width of each slice is proportional to $1/n$ and so

$$\epsilon_T \propto \frac{1}{n^2} \tag{3}$$

A log graph showing the relative error decreasing as the number of intervals used increases is shown in figure 2. The graph does agree with our derived formula for the error, as at each interval the relative error is proportional to $1/n^2$ (equation 3) where n is the number of intervals.

As the relative error decreases to around 10^{-13} the method starts to fail, and the error no longer decreases in the manner expected. This is because when a very high number of intervals is used the value for each slice becomes very small. This means the floating point error obtained by summing these very small numbers becomes more significant, and the fluctuations observed in the graph are produced.

2.2 Simpson's Method

The Simpson's method computes an estimate for the integral using a similar sum as the trapezium method, however the area in each slice is approximated differently, using the formula

$$A_s = \frac{h}{6}\left(f(a) + 4f(\frac{a+b}{2}) + f(b)\right) \tag{4}$$

Where again, h is the width of the slice, f is the function and a and b are the upper and lower bounds of the slice [1]. As before double variables are used for all stored data.

The error in the Simpson's method can be shown analytically in a similar manner to the trapezium method. The formula for the remainder given in [3] is

$$\epsilon_s = \frac{1}{90}h^5 f^{(4)}(\eta)$$

Where as above h is the slice width and η is the value that gives the highest value for $f^{(4)}$ for that slice. Using the same line of reasoning as above it can be shown that the error for the total estimate is

$$\epsilon_T \propto \frac{1}{n^4} \tag{5}$$

when the Simpson's method is used. This means the error will diverge more quickly than the Trapezium rule, and less iterations are needed to produce a comparable level of precision.

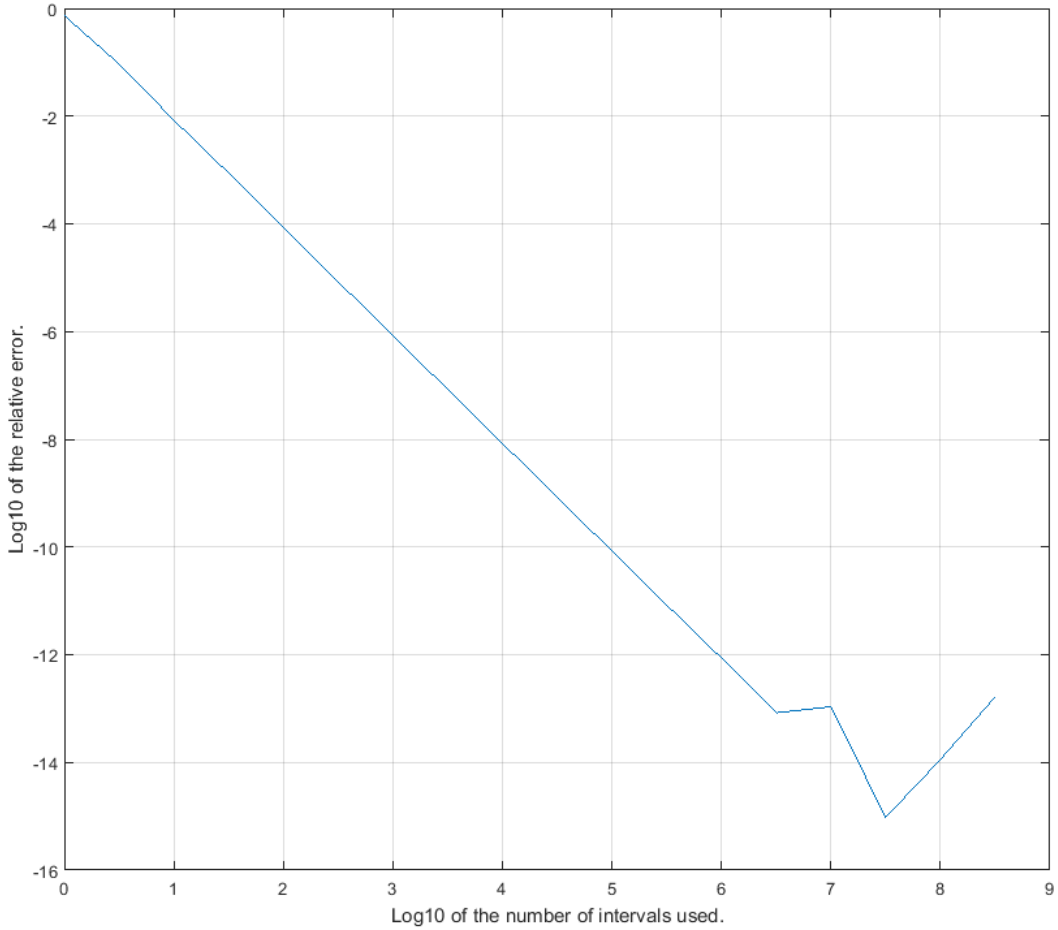

Figure 2: Log plot showing how the error decreases with the number of intervals used in the trapezium method.

In order to check the accuracy of the result the calculated result is compared to the analytic result at each iteration, the program stops and outputs the answer once the difference between the actual answer and the estimated answer is less than desired significant figures.

Similar to the one for the trapezium method a log graph of the relative error against the number of intervals used is shown in 3. The error converges more quickly than that of the trapezium rule which agrees with our analytic error analysis as we expected the error to be proportional to n^{-4} (equation 5) instead of n^{-2} (equation 3) as in the trapezium rule. It is worth noting that the relative error of value for 10^4 intervals is not generated properly. This is likely due to the relative error being a number with significant figures smaller than can be stored in a double variable, so it is rounded to 0. This means when the log is taken an undefined result is given, and the program returns -inf instead of a number.

After using a number of intervals greater than 10^4 the relative error begins to fluctuate and increase again. This is due to similar reasons as for the trapezium rule. When high numbers of intervals are used the floating point error of adding the estimate for each slice becomes significant and produces the fluctuations and increase in error that is observed.

Compared to the trapezium rule a much higher precision is achieved before failure, which is again due to floating point errors when we sum the double variables. Because a higher precision is achieved at a lower number of intervals, where the floating point error is less significant, the Simpson's method performs better, yielding a lower minimum error.

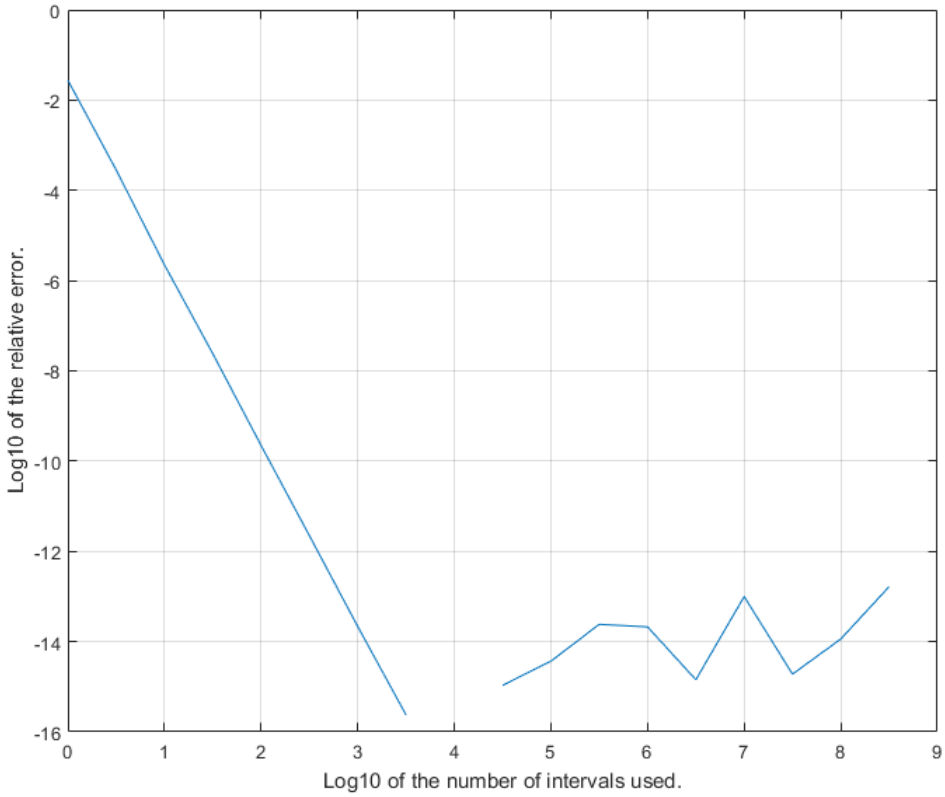

Figure 3: Log plot showing the relative error decreasing as the number of intervals used increases.

3 Gnu Scientific Library

3.1 Basic Integration

For this part routines from the Gnu Scientific Library (GSL) were used to estimate the integral from before. The source code for this is given in "question6.cpp", and uses the QAG adaptive integration routine from the GSL.

The results of this program show that even for one iteration the estimated error is 5×10^{-15}, which is comparable to the best result for both the trapezium and Simpson's rules. However, the number of intervals and correspondingly time taken to calculate a result in the trapezium and Simpson's methods is much longer.

As the 41 point Gauss-Kronrod rules were used the function was called 41 times, to evaluate the function at each of the Kronrod points (see reference [5], chapters 4.6 and 4.7). This is less than the trapezium or Simpson rules, which calls the function 2 and 3 times respectively for each slice. To get a high accuracy in those methods the function must be called a very large number of times, in the order 10^7 for the trapezium rule and 10^4 for the Simpson's rule, whereas in the GSL routine a high accuracy is achieved with only 41 calls.

3.2 Oscillatory Integration

Next the value of the integral

$$\int_0^{2\pi} x \sin 30x \cos x \tag{6}$$

is considered. Figure 4 shows a plot of $f(x) = x \sin 30x \cos x$. The problem with this function is that it is highly oscillating, which makes it more difficult for a computer to estimate numerically. This is due to aliasing effects, which can cause the computer to miscalculate an integral estimate. The GSL provides a routine specifically designed for estimating integrals of oscillatory functions, called "gsl_integration_qawo". The source code "question7.cpp" uses this routine to estimate the value of the integral above.

The results of this program show that a low error tolerance can be achieved quickly by the routine, with only four intervals needed to provide an error estimate less than 10^{-15}. For comparison alternative versions of "question6.cpp" and "question2.cpp" (called "question6a.cpp" and "question2a.cpp") which use this oscillatory function instead are provided for comparison.

The results from running these other programs show that both the trapezium and Simpson's method are capable of producing an estimate for the integral when run with a high number of intervals, for example 10^7. The GSL routine gives an accurate result quicker, needing only 4 subdivision intervals to produce comparable or better accuracy (less than 10^{-15} estimated error).

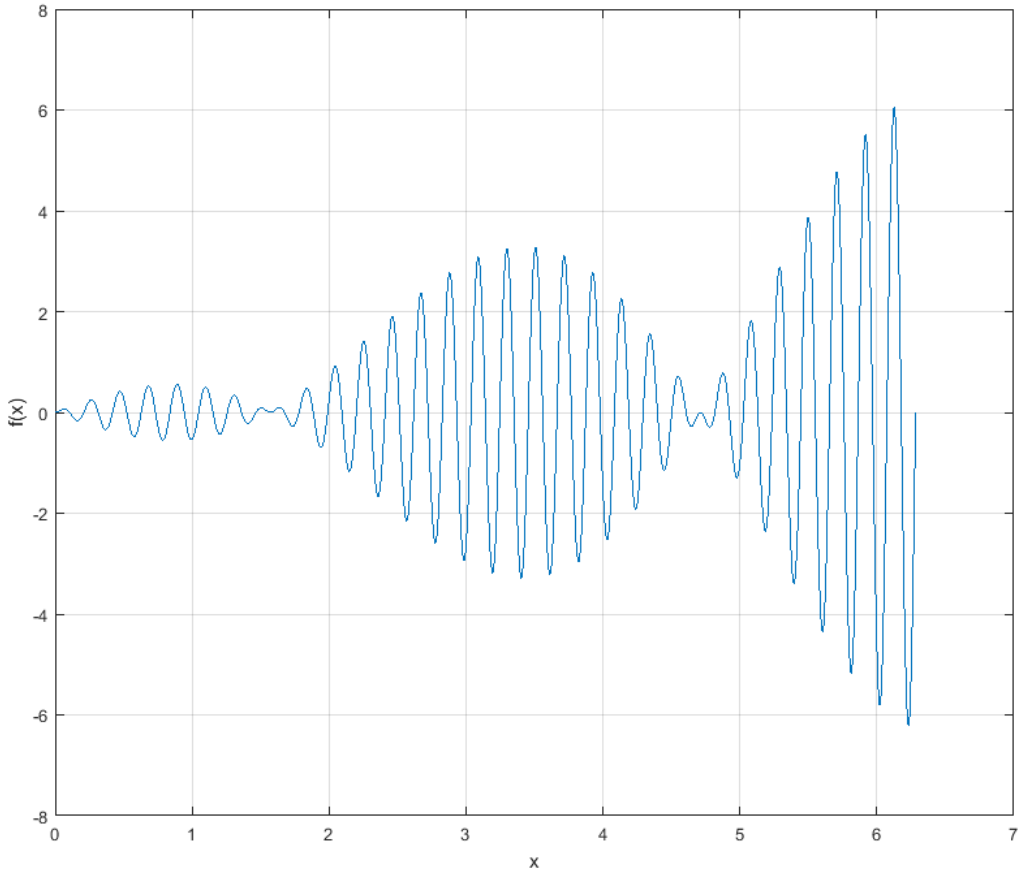

Figure 4: Plot of $f(x)$ against x, where $f(x) = x \sin 30x \cos x$.

Using the adaptive integration routine, however, produces an incorrect result. This is due to the aliasing effects discussed, and is the reason that separate oscillatory routines must exist.

4 Conclusion

The two simple rules for numerical integration (trapezium and Simpson's) are both able to provide us with satisfactory results, with the Simpson's method performing better by reaching a higher accuracy quicker. Floating point errors produced by summing small double variables when high intervals are used cause both of these methods to eventually stop working however.

In contrast the GSL routines are able to provide very high accuracy, and both reach this accuracy much quicker than the simple rules. They do not suffer from the floating point errors of the previous methods, but using an inappropriate method for certain integrations can produce incorrect results due to factors such as aliasing.

References

[1] *Simpson's Rule*. Visited on: 28/10/2016. URL: https://en.wikipedia.org/wiki/Simpson%27s_rule.

[2] *Trapezoidal Rule*. Visited on: 28/10/2016. URL: https://en.wikipedia.org/wiki/Trapezoidal_rule.

[3] Eric W. Weisstein. *Simpson's Rule.* Visited on: 27/10/2016. URL: http://mathworld.wolfram.com/TrapezoidalRule.html.

[4] Eric W. Weisstein. *Trapezoidal Rule.* Visited on: 27/10/2016. URL: http://mathworld.wolfram.com/TrapezoidalRule.html.

[5] William T. Vetterling Brain P. Flannery Willian H. Press Saul E. Teukolsky. *Numerical Recipes (third edition)*. Cambridge University Press. URL: http://apps.nrbook.com/empanel/index.html#.