11. Adaptive Quadrature

Previously, we considered a simple iterative algorithm based on interval doubling for computing an approximation to the value of an integral. In such a method, based on the composite trapezoidal rule, for example, a simple stopping criterion might be to quit when two successive approximations agree to a given tolerance. The problem with such a procedure is that at each iteration, all the subintervals are cut in half without taking into account that we may already have a very good approximation on some subintervals. Thus, the procedure is inefficient since we are doing function evaluations that do not lead to improved accuracy.

For example, if we consider the composite trapezoidal rule on N not necessarily equally spaced subintervals, we have:

$$I(f) = \int_a^b f(x)\,dx = \sum_{i=1}^N I_i(f) = \sum_{i=1}^N \int_{x_{i-1}}^{x_i} f(x)\,dx = \sum_{i=1}^N (x_i - x_{i-1})[f(x_{i-1}) + f(x_i)]/2 \\ - \sum_{i=1}^N f''(\eta_i)(x_i - x_{i-1})^3/12, \quad \eta_i \in (x_{i-1}, x_i).$$

Hence the contribution to the error from the ith subinterval, i.e., the term $f''(\eta_i)(x_i - x_{i-1})^3/12$, depends on both the size of the subinterval and the size of f'' on the subinterval. Where f'' is large, we expect to have to take smaller subintervals to control the error.

11.1. **Estimating the error on a subinterval.** Since we don't know the size of f'' on the subintervals, we need some way to estimate the error on the subinterval. There are several ways this can be done. One is to make use of the interval doubling strategy on a subinterval. For example, if we break up the subinterval (x_{i-1}, x_i) into two equal sized subintervals and use the trapezoidal rule on each, then we get the formula

$$\begin{aligned} I_i(f) &= (x_{i-1/2} - x_{i-1})[f(x_{i-1}) + f(x_{i-1/2})]/2 + (x_i - x_{i-1/2})[f(x_{i-1/2}) + f(x_i)]/2 \\ &\quad - f''(\eta_i^1)(x_{i-1/2} - x_{i-1})^3/12 - f''(\eta_i^2)(x_i - x_{i-1/2})^3/12 \\ &= (x_i - x_{i-1})[f(x_{i-1}) + 2f(x_{i-1/2}) + f(x_i)]/4 - f''(\xi_i)(x_i - x_{i-1})^3/48, \end{aligned}$$

where we have use the fact that $x_{i-1/2} - x_{i-1} = x_i - x_{i-1/2} = (x_i - x_{i-1})/2$.

Equating this to the formula for $I_i(f)$ given by the trapezoidal rule using only one subinterval plus error term, we have

$$\begin{aligned} I_i(f) &= (x_i - x_{i-1})[f(x_{i-1}) + f(x_i)]/2 - f''(\eta_i)(x_i - x_{i-1})^3/12 \\ &= (x_i - x_{i-1})[f(x_{i-1}) + 2f(x_{i-1/2}) + f(x_i)]/4 - f''(\xi_i)(x_i - x_{i-1})^3/48. \end{aligned}$$

Then, if f'' does not change much on the subinterval (x_{i-1}, x_i), we might assume that $f''(\eta_i) \approx f''(\xi_i)$. For example, if f''' is continuous on (x_{i-1}, x_i), then by the Mean Value Theorem,

$$|f''(\eta_i) - f''(\xi_i)| = |f'''(\delta_i)(\eta_i - \xi_i)| \le M_3|\eta_i - \xi_i| \le M_3|x_i - x_{i-1}|.$$

Setting $\xi_i = \eta_i$, we get

$$f''(\eta_i)(x_i - x_{i-1})^3/12 - f''(\eta_i)(x_i - x_{i-1})^3/48 \\ \approx (x_i - x_{i-1})[f(x_{i-1}) + f(x_i)]/2 - (x_i - x_{i-1})[f(x_{i-1}) + 2f(x_{i-1/2}) + f(x_i)]/4.$$

Hence,

$$f''(\eta_i)(x_i - x_{i-1})^3/16 \approx (x_i - x_{i-1})[f(x_{i-1}) - 2f(x_{i-1/2}) + f(x_i)]/4.$$

Thus an estimate for the error (using the trapezoidal rule on one subinterval) is given by:

$$|I_i(f) - (x_i - x_{i-1})[f(x_{i-1}) + f(x_i)]/2| = |f''(\eta_i)(x_i - x_{i-1})^3/12| \\ \approx |(1/3)(x_i - x_{i-1})[f(x_{i-1}) - 2f(x_{i-1/2}) + f(x_i)]|.$$

An estimate for the error (using the trapezoidal rule on two subintervals) is given by:

$$|I_i(f) - (x_i - x_{i-1})[f(x_{i-1}) + 2f(x_{i-1/2}) + f(x_i)]/4| = |f''(\xi_i)(x_i - x_{i-1})^3/48| \\ \approx |(1/12)(x_i - x_{i-1})[f(x_{i-1}) - 2f(x_{i-1/2}) + f(x_i)]|.$$

A second approach is to approximate the integral on a subinterval by two different formulas, say $A_i^1(f)$ and $A_i^2(f)$, with different orders of accuracy, e.g.,

$$|I_i(f) - A_i^1(f)| = O(h^r), \qquad |I_i(f) - A_i^2(f)| = O(h^s),$$

where we assume $s > r$. Then

$$|I_i(f) - A_i^1(f)| \le |I_i(f) - A_i^2(f)| + |A_i^2(f) - A_i^1(f)| \le |A_i^2(f) - A_i^1(f)| + O(h^s).$$

Since $|I_i(f) - A_i^1(f)| = O(h^r)$, $|A_i^2(f) - A_i^1(f)|$ gives an estimate for the dominant part of the error. For example, we could approximate $I_i(f)$ by the trapezoidal rule using two subintervals, or using the same function evaluations, by Simpson's rule

$$I_i(f) \approx (x_i - x_{i-1})[f(x_{i-1}) + 2f(x_{i-1/2}) + f(x_i)]/6.$$

Then, an estimate for the error using the trapezoidal rule on two subintervals is given by:

$$|(x_i - x_{i-1})[f(x_{i-1}) + 4f(x_{i-1/2}) + f(x_i)]/6 - (x_i - x_{i-1})[f(x_{i-1}) + 2f(x_{i-1/2}) + f(x_i)]/4| \\ = |(x_i - x_{i-1})|f(x_{i-1}) - 2f(x_{i-1/2}) + f(x_i)|/12|.$$

Of course, we expect that the use of Simpson's rule gives better accuracy, so if we accept the approximation, we should use the approximation given by Simpson's rule.

11.2. **An adaptive algorithm.** In an adaptive algorithm for numerical integration, we use our estimates for the local error on each subinterval to decide whether we have a sufficiently accurate approximation or whether we need to decrease the size of some subintervals to increase the accuracy of the approximation. We now discuss how this can be done. First, we need to supply an error tolerance. This might be an absolute error tolerance, i.e., $|E| \le \epsilon_a$ or a relative error tolerance, $|E| \le \epsilon_r I(f)$. Many codes ask for both, and quit when $|E| \le \epsilon_a + \epsilon_r A(f)$, where $A(f)$ denotes the current approximation to $I(f)$. Note that $A(f)$ is used instead of $I(f)$, since $I(f)$ is unknown.

Suppose we are trying to make the total error $\le \epsilon$. One way to proceed (simpler to program) is to allow an error of $\epsilon h_i/(b-a)$ on a subinterval of length h_i, where $b-a$ denotes

the total length of the interval of integration. Then, if we had say N subintervals, i.e., $\sum_{i=1}^{N} h_i = b - a$, the total error would be

$$\sum_{i=1}^{N} \epsilon h_i/(b-a) = \epsilon.$$

If on any subinterval, our estimate of the local error exceeds this threshold, then we cut the subinterval size in half and apply our method on each piece with the aim of satisfying the above criterion. Note that if we cut the interval size in half, then we are allowing an error of only $\epsilon h_i/(2[b-a])$ on each piece. However, even using the trapezoidal rule, our local error estimate is of order h_i^3, so cutting h_i in half, should reduce the error by a factor of 8. This is why we expect that halving the subinterval size will reduce the total error. This way of proceeding assumes we want to equidistribute the error in the sense that we are trying to make the error proportional to the subinterval size. However, the goal is to make the total error less than ϵ, so we may be doing unnecessary work. Another possibility is the following. Suppose we have on each interval $I_i = [x_{i-1}, x_i]$ an estimate of the error given by ϵ_i. Then the total error is given by $\sum_i \epsilon_i$. We now simply pick the largest ϵ_i and divide the corresponding interval I_i in half.

An adaptive code based on Simpson's rule in used in *Matlab*. As in the trapezoidal rule discussion above, rather than compare Simpson's rule based on one and two subintervals, the code compares Simpson's rule based on two subintervals with a higher order formula obtained by extrapolation of Simpson's rule (i.e., as done in Romberg integration). Note that the extrapolated formula does not require any new function evaluations.