Principles of Scientific Computing
Local Analysis

Jonathan Goodman

last revised February 15, 2006

Among the most common computational tasks are differentiation, interpolation, and integration. The simplest methods used for these operations are *finite difference* approximations for derivatives, *low order polynomial* interpolation, and *panel method* integration. Finite difference formulas, integration rules, and interpolation form the core of most scientific computing projects that involve solving differential or integral equations.

The finite difference formulas (14) range from simple *low order* approximations (14a) – (14c) to not terribly complicated *high order* methods such as (14e). Figure 2 illustrates that high order methods can be far more accurate than low order ones. This can make the difference between getting useful answers and not in serious large scale applications. The methods here will enable the reader to design professional quality highly accurate methods rather than relying on simple but often inefficient low order ones.

Many methods for these problems involve a *step size*, h. For each h there is an approximation[1] $\widehat{A}(h) \approx A$. We say $\widehat{A}$ is *consistent* if $\widehat{A}(h) \rightarrow A$ as $h \rightarrow 0$. For example, we might estimate $A = f'(x)$ using the finite difference formula (14a): $\widehat{A}(h) = (f(x+h) - f(x))/h$. This is consistent, as $\lim_{h\rightarrow 0} \widehat{A}(h)$ is the definition of $f'(x)$. The accuracy of the approximation depends on f, but the *order of accuracy* does not.[2] The approximation is *first order accurate* if the error is nearly proportional to h for small enough h. It is *second order* if the error goes like h^2. When h is small, $h^2 \ll h$, so approximations with a higher order of accuracy can be much more accurate.

The design of difference formulas and integration rules is based on *local analysis*, approximations to a function f about a *base point* x. These approximations consist of the first few terms of the *Taylor series* expansion of f about x. The *first order* approximation is

$$f(x+h) \approx f(x) + hf'(x) \; . \tag{1}$$

The *second order* approximation is more complicated and more accurate:

$$f(x+h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 \; . \tag{2}$$

Figure 1 illustrates the first and second order approximations. *Truncation error* is the difference between $f(x+h)$ and one of these approximations. For example, the truncation error for the first order approximation is

$$f(x) + f'(x)h - f(x+h) \; .$$

To see how Taylor series are used, substitute the approximation (1) into the finite difference formula (14a). We find

$$\widehat{A}(h) \approx f'(x) = A \; . \tag{3}$$

[1] In the notation of Chapter **??**, $\widehat{A}$ is an estimate of the desired answer, A.

[2] The nominal order of accuracy may be achieved if f is smooth enough, a point that is important in many applications.

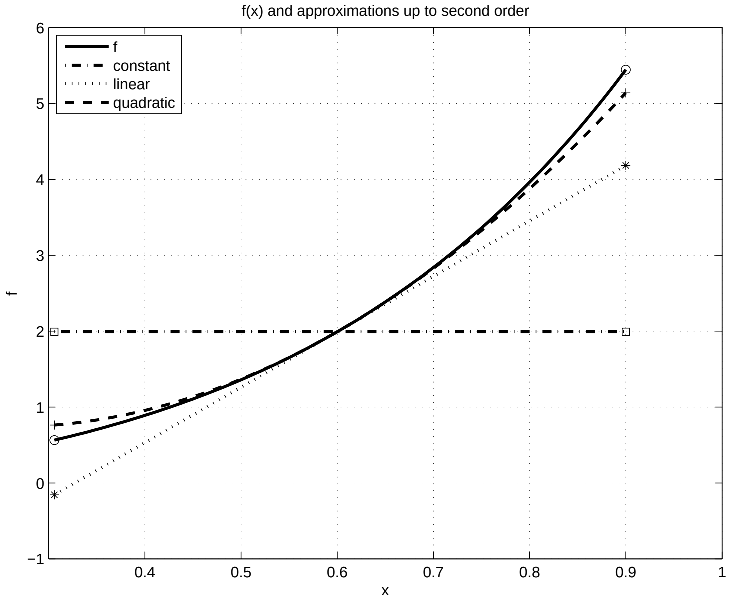

Figure 1: *Plot of $f(x) = xe^{2x}$ together with Taylor series approximations of order zero, one, and two. The base point is $x = .6$ and h ranges from $-.3$ to $.3$. The symbols at the ends of the curves illustrate convergence at fixed h as the order increases. Also, the higher order curves make closer contact with $f(x)$ as $h \to 0$.*

The more accurate Taylor approximation (2) allows us to estimate the error in (14a). Substituting (2) into (14a) gives

$$\widehat{A}(h) \approx A + A_1 h \ , \quad A_1 = \frac{1}{2} f''(x) \ . \tag{4}$$

This *asymptotic error expansion* is an estimate of $\widehat{A} - A$ for a given f and h. It shows that the error is roughly proportional to h for small h. This understanding of truncation error leads to more sophisticated computational strategies. *Richardson extrapolation* combines $\widehat{A}(h)$ and $\widehat{A}(2h)$ to create higher order estimates with much less error. *Adaptive methods* take a desired accuracy, e, and attempt (without knowing A) to find an h with $\left|\widehat{A}(h) - A\right| \leq e$. Error expansions like (4) are the basis of many adaptive methods.

This chapter focuses on truncation error and mostly ignores roundoff. In most practical computations that have truncation error, including numerical solution of differential equations or integral equations, the truncation error is much larger than roundoff. Referring to Figure **??**, the practical range of h from .01 to 10^{-5}, the computed error is roughly $4.1 \cdot h$, as (4) suggests is should be. This starts to break down for the impractically small $h = 10^{-8}$. More sophisticated high order approximations reach roundoff sooner, which can be an issue in code testing, but rarely in large scale production runs.

1 Taylor series and asymptotic expansions

The Taylor series expansion of a function f about the point x is

$$f(x+h) = \sum_{n=0}^{\infty} \frac{1}{n!} f^{(n)}(x) h^n \quad . \tag{5}$$

The notation $f^{(n)}(x)$ refers to the n^{th} derivative of f evaluated at x. The *partial sum* of order p is a degree p polynomial in h:

$$F_p(x,h) = \sum_{n=0}^{p} \frac{1}{n!} f^{(n)}(x) h^n \quad . \tag{6}$$

The partial sum F_p is the Taylor approximation to $f(x+h)$ of order p. It is a polynomial of order p in the variable h. Increasing p makes the approximation more complicated and more accurate. The order $p = 0$ partial sum is simply $F_0(x,h) = f(x)$. The first and second order approximations are (1) and (2) respectively.

The Taylor series sum *converges* if the partial sums converge to f:

$$\lim_{p \to \infty} F_p(x,h) = f(x+h) \ .$$

If there is a positive h_0 so that the series converges whenever $|h| < h_0$, then f is *analytic* at x. A function probably is analytic at most points if there is a formula for it, or it is the solution of a differential equation. Figure 1 plots a function $f(x) = xe^{2x}$ together with the Taylor approximations of order zero, one, and two. The symbols at the ends of the curves illustrate the convergence of this Taylor series when $h = \pm.3$. When $h = .3$, the series converges monotonically: $F_0 < F_1 < F_2 < \cdots \to f(x+h)$. When $h = -.3$, there are approximants on both sides of the answer: $F_0 > F_2 > f(x+h) > F_1$.

It often is more useful to view the Taylor series as an *asymptotic expansion* of $f(x+h)$ valid as $h \to 0$. We explain what this means by giving an analogy between *order of magnitude* (and powers of ten) and *order of approximation* (powers of h). If B_1 and B_2 are positive numbers, then B_2 is an order of magnitude smaller than B_1 roughly if $B_2 \leq .1 \cdot B_1$. If $E_1(h)$ and $E_2(h)$ are functions of h defined for $|h| \leq h_0$, and if there is a C so that $|E_2(h)| \leq C \cdot h \cdot |E_1(h)|$, then we say that E_2 is an order (or an order of approximation) smaller than E_1 as $h \to 0$. This implies that for small enough h ($|h| < 1/C$) $E_2(h)$ is smaller than $E_1(h)$. Reducing h further makes E_2 much smaller than E_1. It is common to know that there is such a C without knowing what it is. Then we do not know how small h has to be before the asymptotic relation $E_2 \ll E_1$ starts to hold. Figure 2 and Figure 3 show that asymptotic relations can hold for practical values of h.

An asymptotic expansion actually is a sequence of approximations, like the $F_p(x,h)$, with increasing order of accuracy. One can view decimal expansions in this way, using order of magnitude instead of order of approximation. The expansion

$$\pi = 3.141592\cdots = 3 + 1 \cdot .1 + 4 \cdot (.1)^2 + 1 \cdot (.1)^3 + 5 \cdot (.1)^4 + \cdots \tag{7}$$

is a sequence of approximations

$$\begin{array}{lcl} \widehat{A}_0 & \approx & 3 \\ \widehat{A}_1 & \approx & 3 + 1 \cdot .1 \\ \widehat{A}_2 & \approx & 3 + 1 \cdot .1 + 4 \cdot (.1)^2 \\ & & \text{etc.} \end{array}$$

The approximation $\widehat{A}_p$ is an order of magnitude more accurate than $\widehat{A}_{p-1}$. The error $\widehat{A}_p - \pi$ is of the order of magnitude $(.1)^{p+1}$, which also is the order of magnitude of the first neglected term. The error in $\widehat{A}_3 = 3.141$ is $\widehat{A}_3 - \pi \approx 6 \cdot 10^{-4}$. This error is approximately the same as the next term, $5 \cdot (.1)^4$. Adding the next term gives an approximation whose error is an order of magnitude smaller:

$$\widehat{A}_3 + 5 \cdot (.1)^4 - \pi = \widehat{A}_4 - \pi \approx -9 \cdot 10^{-5} \; .$$

The *big O* (O for *order*) notation expresses asymptotic size relations. If $B_1(h) > 0$ for $h \neq 0$, the notation, $O(B_1(h))$ as $h \to 0$ refers to any function that is less than $C \cdot B_1(h)$ when $|h| \leq h_0$ (for some C). Thus, $B_2(h) = O(B_1(h))$ as $h \to 0$ if there is a C and an $h_0 > 0$ so that $B_1(h)$ and $B_2(h)$ are defined and and $B_2(h) \leq C \cdot B_1(h)$ for $|h| \leq h_0$. We also write $F(h) = G(h) + O(B_1(h))$ to mean that the function $B_2(h) = F(h) - G(h)$ satisfies $B_2(h) = O(B_1(h))$. For example, it is correct to write $\tan(h) = O(|h|)$ as $h \to 0$ because if $|h| \leq \pi/4$, $\tan(h) \leq 2|h|$. In this case the h_0 is necessary because $\tan(h) \to \infty$ as $h \to \pi/2$ (no C could work for $h = \pi/2$). Also, $C = 1$ does not work, even though $\tan(h) \approx h$ for small h, because $\tan(h) > h$.

There are two common misuses of the big O notation, and we indulge in both below. One is to forget that h could be negative and write $O(h)$ for $O(|h|)$. The other is to say $B = O(h^p)$ to mean that B and h^p are of the same order, that is both $B(h) = O(h^p)$ and $h^p = O(B(h))$. Technically, it is correct to say that $h^3 = O(h^2)$. But this can be misleading, as h^3 actually is an order smaller than h^2. It would be like saying you ran "less than ten miles" when you actually had run less than one mile, technically true but misleading.

We change notation when we view the Taylor series as an asymptotic expansion, writing

$$f(x+h) \sim f(x) + f'(x) \cdot h + \frac{1}{2} f''(x) h^2 + \cdots \; . \tag{8}$$

This means that the right side is an *asymptotic series* that may or may not converge. It represents $f(x+h)$ in the sense that the partial sums $F_p(x,h)$ are a family of approximations of increasing order of accuracy:

$$|F_p(x,h) - f(x+h)| = O(h^{p+1}) \; . \tag{9}$$

The asymptotic expansion (8) is much like the decimal expansion (7). The term in (8)of order $O(h^2)$ is $\frac{1}{2} f''(x) \cdot h^2$. The term in (7) of order of magnitude 10^{-2} is $4 \cdot (.1)^{-2}$. The error in a p term approximation is roughly the first neglected term, since all other neglected terms are at least one order smaller.

Figure 1 illustrates the asymptotic nature of the Taylor approximations. The lowest order approximation is $F_0(x,h) = f(x)$. The graph of F_0 touches the graph of f when $h = 0$ but otherwise has little in common. The graph of $F_1(x,h) = f(x) + f'(x)h$ not only touches the graph of f when $h = 0$, but the curves are tangent. The graph of $F_2(x,h)$ not only is tangent, but has the same curvature and is a better fit for small and not so small h.

1.1 Technical points

This subsection presents two technical points for mathematically minded readers. The first proves the basic fact that underlies most of the analysis in this chapter, that the Taylor series (5) is an asymptotic expansion. The second is two examples of asymptotic expansions that converge to the wrong answer or do not converge at all.

The asymptotic expansion property of Taylor series comes from the Taylor series *remainder theorem*.[3] If the derivatives of f up to order $p+1$ exist and are continuous in the interval $[x, x+h]$, then there is a $\xi \in [x, x+h]$ so that

$$f(x+h) - F_p(x,h) = \frac{1}{(p+1)!} f^{(p+1)}(\xi) h^{p+1} \; . \tag{10}$$

[3] See any good calculus book for a derivation and proof.

If we take

$$C = \frac{1}{(p+1)!} \max_{y \in [x,x+h]} \left| f^{(p+1)}(y) \right| \ ,$$

then we find that

$$\left| F_p(x,h) - f(x+h) \right| \leq C \cdot h^{p+1} \ .$$

This is the proof of (9), which states that the Taylor series is an asymptotic expansion.

The approximation $F_p(x,h)$ includes terms in the sum (8) up to order and including order p. The first neglected term is the term of order $p+1$, which is $\frac{1}{(p+1)}f^{(p+1)}(x)$. This also is the difference $F_{p+1} - F_p$. It differs from the right side of (10) only in ξ being replaced by x. Since $\xi \in [x, x+h]$, this is a small change if h is small. Therefore, the error in the F_p is nearly equal to the first neglected term.

An asymptotic expansion can converge to the wrong answer or not converge at all. We give an example of each. These are based on the fact that exponentials *beat* polynomials in the sense that, for any n,

$$t^n e^{-t} \to 0 \quad \text{as} \ \ t \to \infty \ .$$

If we take $t = a/\left|x\right|$ (because x may be positive or negative), this implies that

$$\frac{1}{x^n} e^{-a/x} \to 0 \quad \text{as} \ \ x \to 0 \ . \tag{11}$$

Consider the function $f(x) = e^{-1/|x|}$. This function is continuous at $x = 0$ if we define $f(0) = 0$. The derivative at zero is (using (11))

$$f'(0) = \lim_{h \to 0} \frac{f(h) - f(0)}{h} = \lim_{h \to 0} \frac{e^{-1/|h|}}{h} = 0 \ .$$

When $x \neq 0$, we calculate $f'(x) = \pm \frac{1}{|x|^2} e^{-1/|x|}$. The first derivative is continuous at $x = 0$ because (11) implies that $f'(x) \to 0 = f'(0)$ as $x \to 0$. Continuing in his way, one can see that each of the higher derivatives vanishes at $x = 0$ and is continuous. Therefore $F_p(0,h) = 0$ for any p, as $f^{(n)}(0) = 0$ for all n. Thus clearly $F_p(0,h) \to 0$ as $p \to \infty$. Simply put, the Taylor series converges to zero for any h because all the terms are zero. But this is the wrong answer, since, $f(h) = e^{1/h} > 0$ if $h \neq 0$. The Taylor series, while asymptotic, converges to the wrong answer.

What goes wrong here is that although the derivatives $f^{(p)}$ happen to take the value zero when $x = 0$, they are very large for x close to zero. The remainder theorem (**??**trf*lo) implies that $F_p(x,h) \to f(x+h)$ as $p \to \infty$ if

$$M_p = \frac{h^p}{p!} \max_{x \leq \xi \leq x+h} \left| f^{(p)}(\xi) \right| \to 0 \quad \text{as} \ p \to \infty.$$

Taking $x = 0$ and any $h > 0$, function $f(x) = e^{-1/|x|}$ has $M_p \to \infty$ as $p \to \infty$.

Here is an example of an asymptotic Taylor series that does not converge at all. Consider

$$f(h) = \int_0^{1/2} e^{-x/h} \frac{1}{1-x} dx \; . \tag{12}$$

The integrand goes to zero *exponentially* as $h \to 0$ for any fixed x. This suggests[4] that most of the integral comes from values of x near zero and that we can approximate the integral by approximating the integrand near $x = 0$. Therefore, we write $1/(1-x) = 1 + x + x^2 + \cdots$, which converges for all x in the range of integration. Integrating separately gives

$$f(h) = \int_0^{1/2} e^{-x/h} dx + \int_0^{1/2} e^{-x/h} x dx + \int_0^{1/2} e^{-x/h} x^2 dx + \cdots \; .$$

We get a simple formula for the integral of the general term $e^{-x/h}x^n$ if we change the upper limit from $1/2$ to ∞. For any fixed n, changing the upper limit of integration makes an exponentially small change in the integral, see problem (6). Therefore the n^{th} term is (for any $p > 0$)

$$\begin{aligned} \int_0^{1/2} e^{-x/h} x^n dx &= \int_0^{\infty} e^{-x/h} x^n dx + O(h^p) \\ &= n! h^{n+1} + O(h^p) \; . \end{aligned}$$

Assembling these gives

$$f(h) \sim h + h^2 + 2h^3 + \cdots + (n-1)! \cdot h^n + \cdots \tag{13}$$

This is an asymptotic expansion because the partial sums are asymptotic approximations:

$$\left| h + h^2 + 2h^3 + \cdots + (p-1)! \cdot h^p - f(h) \right| = O(h^{p+1}) \; .$$

But the infinite sum does not converge; for any $h > 0$ we have $n! \cdot h^{n+1} \to \infty$ as $n \to \infty$.

In these examples, the higher order approximations have smaller ranges of validity. For (**??**), the three term approximation $f(h) \approx h + h^2 + 2h^3$ is reasonably accurate when $h = .3$ but the six term approximation is less accurate, and the ten term "approximation" is 4.06 for an answer less than .5. The ten term approximation is very accurate when $h = .01$ but the fifty term "approximation" is astronomical.

2 Numerical Differentiation

One basic numerical task is estimating the derivative of a function from given function values. Suppose we have a smooth function, $f(x)$, of a single variable,

[4] A more precise version of this intuitive argument is in exercise 6.

x. The problem is to combine several values of f to estimate f'. These *finite difference* approximations are useful in themselves, and because they underlie methods for solving differential equations of all kinds. Several common finite difference approximations are

$$\left.\begin{aligned}
f'(x) &\approx \frac{f(x+h)-f(x)}{h} && (a)\\
f'(x) &\approx \frac{f(x)-f(x-h)}{h} && (b)\\
f'(x) &\approx \frac{f(x+h)-f(x-h)}{2h} && (c)\\
f'(x) &\approx \frac{-f(x+2h)+4f(x+h)-3f(x)}{2h} && (d)\\
f'(x) &\approx \frac{-f(x+2h)+8f(x+h)-8f(x-h)+f(x+2h)}{12h} && (e)
\end{aligned}\right\} \tag{14}$$

The first three have simple geometric interpretations as the slope of lines connecting nearby points on the graph of $f(x)$. A carefully drawn figure shows that (14c) is more accurate than (14a). We give an analytical explanation of this below. The last two are more technical. The formulas (14a), (14b), and (14d) are *one sided* because they use values only on one side of x. The formulas (14c) and (14e) are *centered* because they use points symmetrical about x and with opposite weights.

The Taylor series expansion (8) allows us to calculate the accuracy of each of these approximations. Let us start with the simplest (14a). Substituting (8) into the right side of (14a) gives

$$\frac{f(x+h)-f(x)}{h} \sim f'(x) + h\frac{f''(x)}{2} + h^2\frac{f'''(x)}{6} + \cdots \;. \tag{15}$$

This may be written:

$$\frac{f(x+h)-f(x)}{h} = f'(x) + E_a(h) \;,$$

where

$$E_a(h) \sim \frac{1}{2}f''(x)\cdot h + \frac{1}{6}f'''(x)\cdot h^2 + \cdots \;. \tag{16}$$

In particular, this shows that $E_a(h) = O(h)$, which means that the one sided two point finite difference approximation is first order accurate. Moreover,

$$E_a(h) = \frac{1}{2}f''(x)\cdot h + O(h^2)\;, \tag{17}$$

which is to say that, to *leading order*, the error is proportional to h and given by $\frac{1}{2}f''(x)$.

Taylor series analysis applied to the two point centered difference approximation (14c) leads to

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + E_c(h)$$

where

$$\begin{aligned} E_c(h) &\sim \frac{1}{6} f'''(x) \cdot h^2 + \frac{1}{24} f^{(5)}(x) \cdot h^4 + \cdots \\ &= \frac{1}{6} f'''(x) \cdot h^2 + O(h^4) \end{aligned} \tag{18}$$

This centered approximation is second order accurate, $E_c(h) = O(h^2)$. This is one order more accurate than the one sided approximations (14a) and (14b). Any centered approximation such as (14c) or (14e) must be at least second order accurate because of the symmetry relation $\widehat{A}(-h) = \widehat{A}(h)$. Since $A = f'(x)$ is independent of h, this implies that $E(h) = \widehat{A}(h) - A$ is symmetric. If $E(h) = c \cdot h + O(h^2)$, then

$$E(-h) = -c \cdot h + O(h^2) = E(h) + O(h^2) \approx -E(h) \quad \text{for small } h,$$

which contradicts $E(-h) = E(h)$. The same kind of reasoning shows that the $O(h^3)$ term in (18) must be zero.

A Taylor series analysis shows that the three point one sided formula (14d) is second order accurate, while the four point centered approximation (14e) is fourth order. Sections 3.1 and 5 give two ways to find the coefficients 4, -3, and 8 achieve these higher orders of accuracy.

Figure 2 illustrates many of these features. The first is that the higher order formulas (14c), (14d), and (14e) actually are more accurate when h is small. For $h = .5$, the first order two point one sided difference formula is more accurate than the second order accurate three point formula, but their proper asymptotic ordering is established by $h = .01$. For $h \leq 10^{-5}$ with the fourth order centered difference formula and $h = 10^{-7}$ with the second order formula, double precision roundoff error makes the results significantly different from what they would be in exact arithmetic. The rows labeled $\widehat{E}$ give the leading order Taylor series estimate of the error. For the first order formula, (17) shows that this is $\widehat{E}(h) = \frac{1}{2} f''(x) \cdot h$. For the second order centered formula, (18) gives leading order error $\widehat{E}_c(h) = \frac{1}{6} f'''(x) \cdot h^2$. For the three point one sided formula, the coefficient of $f'''(x) \cdot h^2$ is $\frac{1}{3}$, twice the coefficient for the second order centered formula. For the fourth order formula, the coefficient of $f^{(5)}(x) \cdot h^4$ is $\frac{1}{30}$. The table shows that $\widehat{E}$ is a good predictor of E, if h is at all small, until roundoff gets in the way. The smallest error[5] in the table comes from the fourth order formula and $h = 10^{-5}$. It is impossible to have an error this small with a first

[5] The error would have been $-3 \cdot 10^{-19}$ rather than $-6 \cdot 10^{-12}$, seven orders of magnitude smaller, in exact arithmetic. The best answer comes despite some catastrophic cancellation, but not completely catastrophic.

h	(14a)	(14c)	(14d)	(14e)
.5	3.793849	0.339528	7.172794	0.543374
E	2.38e+00	-1.08e+00	5.75e+00	-8.75e-01
$\widehat{E}$	5.99e+00	-1.48e+00	-2.95e+00	-1.85e+00
.01	2.533839	1.359949	1.670135	1.415443
E	1.12e+00	-5.84e-02	2.52e-01	-2.87e-03
$\widehat{E}$	1.20e+00	-5.91e-02	-1.18e-01	-2.95e-03
$5 \cdot 10^{-3}$	1.999796	1.403583	1.465752	1.418128
E	5.81e-01	-1.47e-02	4.74e-02	-1.83e-04
$\widehat{E}$	5.99e-01	-1.48e-02	-2.95e-02	-1.85e-04
10^{-3}	1.537561	1.417720	1.419642	1.418311
E	1.19e-01	-5.91e-04	1.33e-03	-2.95e-07
$\widehat{E}$	1.20e-01	-5.91e-04	-1.18e-03	-2.95e-07
10^{-5}	1.418431	1.418311	1.418311	1.418311
E	1.20e-04	-5.95e-10	1.16e-09	-6.05e-12
$\widehat{E}$	1.20e-04	-5.91e-10	-1.18e-09	-2.95e-19
10^{-7}	1.418312	1.418311	1.418311	1.418311
E	1.20e-06	2.76e-10	3.61e-09	8.31e-10
$\widehat{E}$	1.20e-06	-5.91e-14	-1.18e-13	$-2.95 \cdot 10^{-27}$

Figure 2: Estimates of $f'(x)$ with $f(x) = \sin(5x)$ and $x = 1$ using formulas (14a), (14c), (14d), and (14e). Each group of three rows corresponds to one h value. The top row gives the finite difference estimate of$f'(x)$, the middle row gives the error $E(h)$, and the third row is $\widehat{E}(h)$, the leading Taylor series term in the error formula. All calculations were done in double precision floating point arithmetic.

or second order formula no matter what the step size. Note that the error in the (14e) column increased when h was reduced from 10^{-5} to 10^{-7} because of roundoff.

A difference approximation may not achieve its expected order of accuracy if the requisite derivatives are infinite or do not exist. As an example of this, let $f(x)$ be the function

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x^2 & \text{if } x \geq 0 \end{cases} .$$

If we want $f'(0)$, the formulas (1c) and (1e) are only first order accurate despite their higher accuracy for smoother functions. This f has a mild singularity, a discontinuity in its second derivative. Such a singularity is hard to spot on a graph, but may have a drastic effect on the numerical analysis of the function.

We can use finite differences to approximate higher derivatives such as

$$\frac{f(x+h) - 2f(x) + f(x-h)}{h^2} = f''(x) + \frac{h^2}{12} f^{(4)} + O(h^4) \ ,$$

and to estimate partial derivatives of functions depending on several variables, such as

$$\frac{f(x+h,y)-f(x-h,y)}{2h} \sim \frac{\partial}{\partial x}f(x,y) + \frac{h^2}{3}\frac{\partial^3 f}{\partial x^3}(x,y) + \cdots \ .$$

2.1 Mixed partial derivatives

There are several new features that arise only when evaluating mixed partial derivatives or sums of partial derivatives in different variables. For example, suppose we want to evaluate[6] $f_{xy} = \partial_x \partial_y f(x,y)$. Rather than using the same h for both[7] x and y, we use step size Δx for x and Δy for y. The first order one sided approximation for f_y is

$$f_y \approx \frac{f(x, y+\Delta y) - f}{\Delta y} \ .$$

We might hope this, and

$$f_y(x+\Delta x, y) \approx \frac{f(x+\Delta x, y+\Delta y) - f(x+\Delta x, y)}{\Delta y} \ ,$$

are accurate enough so that

$$\begin{aligned}
\partial_x\big(\partial_y f\big) &\approx \frac{f_y(x+\Delta x, y) - f_y}{\Delta x} \\
&\approx \frac{\dfrac{f(x+\Delta x, y+\Delta y) - f(x+\Delta x, y)}{\Delta y} - \dfrac{f(x, y+\Delta y) - f}{\Delta y}}{\Delta x} \\
f_{xy} &\approx \frac{f(x+\Delta x, y+\Delta y) - f(x+\Delta x, y) - f(x, y+\Delta y) + f}{\Delta x \Delta y} \ , \qquad (19)
\end{aligned}$$

is consistent[8].

To understand the error in (19), we need the Taylor series for functions of more than one variable. The rigorous remainder theorem is more complicated, but it suffices here to use all of the "first" neglected terms. The expansion is

$$\begin{aligned}
f(x+\Delta x, y+\Delta y) \ &\sim \ f + \Delta x f_x + \Delta y f_y \\
&\quad + \frac{1}{2}\Delta x^2 f_{xx} + \Delta x \Delta y f_{xy} + \frac{1}{2}\Delta y^2 f_{yy}
\end{aligned}$$

[6] We abbreviate formulas by denoting partial derivatives by subscripts, $\partial_x f = f_x$, etc., and by leaving out the arguments if they are (x,y), so $f(x+\Delta x, y) - f(x,y) = f(x+\Delta x, y) - f \approx \Delta x f_x(x,y) = \Delta x f_x$.

[7] The expression $f(x+h, y+h)$ does not even make sense if x and y have different physical units.

[8] The same calculation shows that the right side of (19) is an approximation of $\partial_y\left(\partial_x f\right)$. This is one proof that $\partial_y \partial_x f = \partial_y \partial_x f$.

$$
\begin{aligned}
&+\frac{1}{6}\Delta x^3 f_{xxx} + \frac{1}{2}\Delta x^2 \Delta y f_{xxy} + \frac{1}{2}\Delta x \Delta y^2 f_{xyy} + \frac{1}{6}\Delta y^3 f_{yyy} \\
&\quad + \ \cdots \\
&\quad + \frac{1}{p!}\sum_{k=0}^{p}\binom{p}{k}\Delta x^{n-k}\Delta y^k \partial_x^{p-k}\partial_y^k f + \cdots \ .
\end{aligned}
$$

If we keep just the terms on the top row on the right, the second order terms on the second row are the first neglected terms, and (using the inequality $\Delta x \Delta y \leq \Delta x^2 + \Delta y^2$):

$$
f(x+\Delta x, y+\Delta y) = f + \Delta x f_x + \Delta y f_y + O\left(\Delta x^2 + \Delta y^2\right) \ .
$$

Similarly,

$$
\begin{aligned}
&f(x+\Delta x, y+\Delta y) \\
&\qquad = f + \Delta x f_x + \Delta y f_y + \frac{1}{2}\Delta x^2 f_{xx} + \Delta x \Delta y f_{xy} + \frac{1}{2}\Delta y^2 f_{yy} \\
&\qquad + O\left(\Delta x^3 + \Delta y^3\right) \ .
\end{aligned}
$$

Of course, the one variable Taylor series is

$$
f(x+\Delta x, y) = f + \Delta x f_x + \frac{1}{2}\Delta x^2 f_{xx} + O\left(\Delta x^3\right) \ , \text{ etc.}
$$

Using all these, and some algebra, gives

$$
\begin{aligned}
&\frac{f(x+\Delta x, y+\Delta y) - f(x+\Delta x, y) - f(x, y+\Delta y) + f}{\Delta x \Delta y} \\
&\qquad\qquad = f_{xy} + O\left(\frac{\Delta x^3 + \Delta y^3}{\Delta x \Delta y}\right) \ . \qquad (20)
\end{aligned}
$$

This shows that the approximation (19) is first order, at least if Δx is roughly proportional to Δy. The fuller Taylor expansion above gives a quantitative estimate of the error:

$$
\begin{aligned}
&\frac{f(x+\Delta x, y+\Delta y) - f(x+\Delta x, y) - f(x, y+\Delta y) + f}{\Delta x \Delta y} - f_{xy} \\
&\qquad\qquad \approx \frac{1}{2}\left(\Delta x f_{xxy} + \Delta y f_{xyy}\right) \ . \qquad (21)
\end{aligned}
$$

This formula suggests (and it is true) that in exact arithmetic we could let $\Delta x \to 0$, with Δy fixed but small, and still have a reasonable approximation to f_{xy}. The less detailed version (20) suggests that might no be so.

A partial differential equation may involve a *differential operator* that is a sum of partial derivatives. One way to approximate a differential operator is to approximate each of the terms separately. for example, the *Laplace* operator (or *Laplacian*), which is $\triangle = \partial_x^2 + \partial_y^2$ in two dimensions, may be approximated by

$$\begin{aligned}
\triangle f(x,y) &= \partial_x^2 f + \partial_y^2 f \\
&\approx \frac{f(x+\Delta x,y) - 2f + f(x-\Delta x,y)}{\Delta x^2} \\
&\quad + \frac{f(x,y+\Delta y) - 2f + f(x,y-2\Delta y)}{\Delta y^2} \ .
\end{aligned}$$

If $\Delta x = \Delta y = h$ (x and y have the same units in the Laplace operator), then this becomes

$$\triangle f \approx \frac{1}{h^2}\big(f(x+h,y) + f(x-h,y) + f(x,y+h) + f(x,y-h) - 4f\big) \ . \tag{22}$$

This is the *standard* five point approximation (seven points in three dimensions). The leading error term is

$$\frac{h^2}{12}\big(\partial_x^4 f + \partial_y^4 f\big) \ . \tag{23}$$

The simplest *heat equation* (or *diffusion* equation) is $\partial_t f = \frac{1}{2}\partial_x^2 f$. The *space* variable, x, and the *time* variable, t have different units. We approximate the differential operator using a first order forward difference approximation in time and a second order centered approximation in space. This gives

$$\partial_t f - \frac{1}{2}\partial_x^2 f \approx \frac{f(x,t+\Delta t) - f}{\Delta t} - \frac{f(x+\Delta x,t) - 2f + f(x-\Delta x,t)}{2\Delta x^2} \ . \tag{24}$$

The leading order error is the sum of the leading errors from time differencing ($\frac{1}{2}\Delta t \partial_t^2 f$) and space differencing ($\frac{\Delta x^2}{24}\partial_x^4 f$), which is

$$\frac{1}{2}\Delta t \partial_t^2 f - \frac{\Delta x^2}{24}\partial_x^4 f \ . \tag{25}$$

For many reasons, people often take Δt proportional to Δx^2. In the simplest case of $\Delta t = \Delta x^2$, the leading error becomes

$$\Delta x^2 \left(\frac{1}{2}\partial_t^2 f - \frac{1}{24}\partial_x^4 f\right) \ .$$

This shows that the overall approximation (24) is second order accurate if we take the time step to be the square of the space step.

3 Error Expansions and Richardson Extrapolation

The error expansions (16) and (18) above are instances of a common situation that we now describe more systematically and abstractly. We are trying to compute A and there is an approximation with

$$\widehat{A}(h) \to A \ \text{ as } \ h \to 0 \ .$$

The error is $E(h) = \widehat{A}(h) - A$. A general *asymptotic error expansion* in powers of h has the form

$$\widehat{A}(h) \sim A + h^{p_1}A_1 + h^{p_2}A_2 + \cdots \quad , \tag{26}$$

or, equivalently,

$$E(h) \sim h^{p_1}A_1 + h^{p_2}A_2 + \cdots \quad .$$

As with Taylor series, the expression (26) does not imply that the series on the right converges to $\widehat{A}(h)$. Instead, the asymptotic relation (26) means that, as $h \to 0$,

$$\left.\begin{array}{ll} \widehat{A}(h) - (A + h^{p_1}A_1) = O(h^{p_2}) & (a) \\[2ex] \widehat{A}(h) - (A + h^{p_1}A_1 + h^{p_2}A_2) = O(h^{p_3}) & (b) \\[2ex] \textit{and so on.} & \end{array}\right\} \tag{27}$$

It goes without saying that $0 < p_1 < p_2 < \cdots$. The statement (27a) says not only that $A + A_1h^{p_1}$ is a good approximation to $\widehat{A}(h)$, but that the error has the same order as the first neglected term, $A_2h^{p_2}$. The statement (27b) says that including the $O(h^{p_2})$ term improves the approximation to $O(h^{p_3})$, and so on.

Many asymptotic error expansions arise from Taylor series manipulations. For example, the two point one sided difference formula error expansion (15) gives $p_1 = 1$, $A_1 = \frac{1}{2}f''(x)$, $p_2 = 2$, $A_2 = \frac{1}{6}f'''(x)$, etc. The error expansion (18) for the two point centered difference formula implies that $p_1 = 2$, $p_2 = 4$, $A_1 = \frac{1}{6}f'''(x)$, and $A_2 = \frac{1}{24}f^{(5)}(x)$. The three point one sided formula has $p_1 = 2$ because it is second order accurate, but $p_2 = 3$ instead of $p_2 = 4$. The fourth order formula has $p_1 = 4$ and $p_2 = 6$.

It is possible that an approximation is p^{th} order accurate in the big O sense, $|E(h)| \leq C \cdot h^p$, without having an asymptotic error expansion of the form (26). Figure 4 has an example showing that this can happen when the function $f(x)$ is not sufficiently smooth. Most of the extrapolation and debugging tricks described here do not apply in those cases.

We often work with asymptotic error expansions for which we know the powers p_k but not the coefficients, A_k. For example, in finite difference approximations, the A_k depend on the function f but the p_k do not. Two computational techniques that use this information are are *Richardson extrapolation* and *convergence analysis*. Richardson extrapolation combines $\widehat{A}(h)$ approximations for several values of h to produce a new approximation that has greater order of accuracy than $\widehat{A}(h)$. Convergence analysis is a debugging method that tests the order of accuracy of numbers produced by a computer code.

3.1 Richardson extrapolation

Richardson extrapolation increases the order of accuracy of an approximation provided that the approximation has an asymptotic error expansion of the form

(26) with known p_k. In its simplest form, we compute $\widehat{A}(h)$ and $\widehat{A}(2h)$ and then form a linear combination that eliminates the leading error term. Note that

$$\begin{aligned}\widehat{A}(2h) &= A + (2h)^{p_1} A_1 + (2h)^{p_2} A_2 + \cdots \\ &= A + 2^{p_1} h^{p_1} A_1 + 2^{p_2} h^{p_2} A_2 + \cdots \quad ,\end{aligned}$$

so

$$\frac{2^{p_1}\widehat{A}(h) - \widehat{A}(2h)}{2^{p_1}-1} = A + \frac{2^{p_1}-2^{p_2}}{2^{p_1}-1} h^{p_2} A_2 + \frac{2^{p_3}-2^{p_2}}{2^{p_1}-1} h^{p_3} A_3 + \cdots \quad .$$

In other words, the *extrapolated* approximation

$$\widehat{A}^{(1)}(h) = \frac{2^{p_1}\widehat{A}(h) - \widehat{A}(2h)}{2^{p_1}-1} \tag{28}$$

has order of accuracy $p_2 > p_1$. It also has an asymptotic error expansion,

$$\widehat{A}^{(1)}(h) = A + h^{p_2} A_2^{(1)} + h^{p_3} A_3^{(1)} + \cdots \quad ,$$

where $A_2^{(1)} = \dfrac{2^{p_1}-2^{p_2}}{2^{p_1}-1} A_2$, and so on.

Richardson extrapolation can be repeated to remove more asymptotic error terms. For example,

$$\widehat{A}^{(2)}(h) = \frac{2^{p_2}\widehat{A}^{(1)}(h) - \widehat{A}^{(1)}(2h)}{2^{p_2}-1}$$

has order p_3. Since $\widehat{A}^{(1)}(h)$ depends on $\widehat{A}(h)$ and $\widehat{A}(2h)$, $\widehat{A}^{(2)}(h)$ depends on $\widehat{A}(h)$, $\widehat{A}(2h)$, and $\widehat{A}(4h)$. It is not necessary to use powers of 2, but this is natural in many applications. Richardson extrapolation will not work if the underlying approximation, $\widehat{A}(h)$, has accuracy of order h^p in the $O(h^p)$ sense without at least one term of an asymptotic expansion.

Richardson extrapolation, allows us to derive higher order difference approximations from low order ones. Start, for example, with the first order one sided approximation to $f'(x)$ given by (14a). Taking $p_1 = 1$ in (28) leads to the second order approximation

$$\begin{aligned} f'(x) &\approx 2 \cdot \frac{f(x+h)-f(x)}{h} - \frac{f(x+2h)-f(x)}{2h} \\ &= \frac{-f(x+2h)+4f(x+h)-3f(x)}{2h} \quad ,\end{aligned}$$

which is the second order three point one sided difference approximation (14d). Starting with the second order centered approximation (14c) (with $p_1 = 2$ and $p_2 = 4$) leads to the fourth order approximation (14e). The second order one sided formula has $p_1 = 2$ and $p_2 = 3$. Applying Richardson extrapolation to it gives a one sided formula that uses $f(x+4h)$, $f(x+2h)$, $f(x+h)$, and $f(x)$ to give a third order approximation. A better third order one sided approximation would use $f(x+3h)$ instead of $f(x+4h)$. Section 5 explains how to do this.

Richardson extrapolation may also be applied to the output of a complex code. Run it with step size h and $2h$ and apply (28) to the output. This is sometimes applied to stochastic differential equations as an alternative to making up high order schemes from scratch, which can be time consuming and intricate.

3.2 Convergence analysis

We can test a code, and the algorithm it is based on, using ideas related to Richardson extrapolation. A naive test would be to do runs with decreasing h values to check whether $\widehat{A}(h) \to A$ as $h \to 0$. A *convergence analysis* based on asymptotic error expansions can be better. For one thing, we might not know A. Even if we run a test case where A is known, it is common that a code with mistakes limps to convergence, but not as accurately or reliably as the correct code would. If we bother to write a code that is more than first order accurate, we should test that we are getting the order of accuracy we worked for.

There are two cases, the case where A is known and the case where A is not known. While we probably would not write a code for a problem to which we know the answer, it is often possible to apply a code to a problem with a known answer for debugging. In fact, a code should be written modularly so that it is easy to apply it to a range or problems broad enough to include some trivial and at least one nontrivial problem[9] with a known answer.

If A is known, we can run the code with step size h and $2h$ and, from the resulting approximations, $\widehat{A}(h)$ and $\widehat{A}(2h)$, compute

$$\begin{aligned} E(h) &\approx A_1 h^{p_1} + A_2 h^{p_2} + \cdots \ , \\ E(2h) &\approx 2^{p_1} A_1 h^{p_1} + 2^{p_2} A_2 h^{p_2} + \cdots \ . \end{aligned}$$

For small h the first term is a good enough approximation so that the ratio should be approximately the characteristic value

$$R(h) = \frac{E(2h)}{E(h)} \approx 2^{p_1} \ . \tag{29}$$

Figure 3 is a computational illustration of this phenomenon. As $h \to 0$, the ratios converge to the expected result $2^{p_1} = 2^3 = 8$. Figure 4 shows what may happen when we apply this convergence analysis to an approximation that is second order accurate in the big O sense without having an asymptotic error expansion. The error gets very small but the error ratio does not have simple behavior as in Figure 3.

[9] A trivial problem is one that is too simple to test the code fully. For example, if you compute the derivative of a linear function, any of the formulae (14a) – (14e) would give the exact answer. The fourth order approximation (14e) gives the exact answer for any polynomial of degree less than five.

h	Error: $E(h)$	Ratio: $E(h)/E(h/2)$
.1	4.8756e-04	3.7339e+00
.05	1.3058e-04	6.4103e+00
.025	2.0370e-05	7.3018e+00
.0125	2.7898e-06	7.6717e+00
6.2500e-03	3.6364e-07	7.8407e+00
3.1250e-03	4.6379e-08	7.9215e+00
1.5625e-03	5.8547e-09	7.9611e+00
7.8125e-04	7.3542e-10	———-

Figure 3: Convergence study for a third order accurate approximation. As $h \rightarrow 0$, the ratio converges to $2^3 = 8$. The h values in the left column decrease by a factor of two from row to row.

h	Error: $E(h)$	Ratio: $E(h)/E(h/2)$
.1	1.9041e-02	2.4014e+00
.05	7.9289e-03	1.4958e+01
.025	5.3008e-04	-1.5112e+00
.0125	-3.5075e-04	3.0145e+00
6.2500e-03	-1.1635e-04	1.9880e+01
3.1250e-03	-5.8529e-06	-8.9173e-01
1.5625e-03	6.5635e-06	2.8250e+00
7.8125e-04	2.3233e-06	———-

Figure 4: Convergence study for an approximation that is second order accurate in the sense that $|E(h)| = O(h^2)$ but that has no asymptotic error expansion. The h values are the same as in Figure 3. The errors decrease in an irregular fashion.

Convergence analysis can be applied even when A is not known. In this case we need three approximations, $\widehat{A}(4h)$, $\widehat{A}(2h)$, and $\widehat{A}(h)$. Again assuming the existence of an asymptotic error expansion (26), we get, for small h,

$$R'(h) = \frac{\widehat{A}(4h) - \widehat{A}(2h)}{\widehat{A}(2h) - \widehat{A}(h)} \approx 2^{p_1} \quad . \tag{30}$$

4 Integration

Numerical integration means finding approximations for quantities such as

$$I = \int_a^b f(x)dx \quad .$$

Rectangle	$\widehat{I}_k = h_k f(x_k)$	1^{st} order
Trapezoid	$\widehat{I}_k = \dfrac{h_k}{2}\left(f(x_k) + f(x_{k+1})\right)$	2^{nd} order
Midpoint	$\widehat{I}_k = h_k f(x_{k+1/2})$	2^{nd} order
Simpson	$\widehat{I}_k \approx \dfrac{h_k}{6}\big(f(x_k) + 4f(x_{k+1/2}) + f(x_{k+1})\big)$	4^{th} order
2 point GQ	$\widehat{I}_k = \dfrac{h_k}{2}\big(f(x_{k+1/2} - h_k\xi) + f(x_{k+1/2} + h_k\xi)\big)$	4^{th} order
3 point GQ	$\widehat{I}_k = \dfrac{h_k}{18}\big(5f(x_{k+1/2} - h_k\eta) + 8f(x_{k+1/2}) + 5f(x_{k+1/2} + h_k\eta)\big)$	6^{th} order

Figure 5: Common panel integration rules. The last two are Gauss quadrature (Gauss – Legendre to be precise) formulas. The definitions are $\xi = \frac{1}{2\sqrt{3}}$ and $\eta = \frac{1}{2}\sqrt{\frac{3}{5}}$.

We discuss only *panel methods* here, though there are other elegant methods. In a panel method, the integration interval, $[a, b]$, is divided into n subintervals, or *panels*, $P_k = [x_k, x_{k+1}]$, where $a = x_0 < x_1 < \cdots < x_n = b$. If the panel P_k is small, we can get an accurate approximation to

$$I_k = \int_{P_k} f(x)dx = \int_{x_k}^{x_{k+1}} f(x)dx \tag{31}$$

using a few evaluations of f inside P_k. Adding these approximations gives an approximation to I:

$$\widehat{I} = \sum_{k=0}^{n-1} \widehat{I}_k \ . \tag{32}$$

Some of the more common panel integral approximations are given in Figure 5, where we write $x_{k+1/2} = (x_{k+1} + x_k)/2$ for the midpoint of the panel and $h_k = x_{k+1} - x_k$ is the width. Note that x_k is the left endpoint of P_k and the right endpoint of P_{k-1}. In the trapezoid rule and Simpon's rule, we need not evaluate $f(x_k)$ twice.

For our error analysis, we assume that all the panels are the same size

$$h = \Delta x = |P_k| = x_{k+1} - x_k \text{ for all } k.$$

Given this restriction, not every value of h is allowed because $b - a = nh$ and n is an integer. When we take $h \to 0$, we will assume that h only takes allowed values $h = (b - a)/n$. The *local truncation error* is the integration error over one panel. The overall *global* error is the sum of the local truncation errors in all the panels. The global error usually is one power of h larger than the local truncation error. If the error per panel is $O(h^q)$, then the total error will be of the order of h^q multiplied by n, the number of panels. Since $n = (b - a)/h$, this suggests that the global error will be of order $h^q \cdot (b - a)/h = O(h^{q-1})$.

For the local truncation error analysis, let $P = [x_*, x_* + h]$ be a generic panel. The panel integration rule approximates the panel integral

$$I_P = \int_P f(x)dx = \int_{x_*}^{x_*+h} f(x)dx$$

with the approximation, $\widehat{I}_P$. For example, the rectangle rule (top row of Figure 5) has panel integration rule

$$\int_{x_*}^{x_*+h} f(x)dx \approx \widehat{I}_P(h) = hf(x_*) \quad .$$

To estimate the difference between I_P and $\widehat{I}_P(h)$, we expand f in a Taylor series about x_*:

$$f(x) \sim f(x_*) + f'(x_*)(x - x_*) + \frac{1}{2}f''(x_*)(x - x_*)^2 + \cdots \quad .$$

Integrating this term by term leads to

$$\begin{aligned} I_P &\sim \int_P f(x_*)dx + \int_P f'(x_*)(x - x_*)dx + \cdots \\ &= f(x_*)h + \frac{1}{2}f'(x_*)h^2 + \frac{1}{6}f''(x_*)h^3 + \cdots \quad . \end{aligned}$$

The error in integration over this panel then is

$$E(P, h) = \widehat{I}_P(h) - I_P \sim -\frac{1}{2}f'(x_*)h^2 - \frac{1}{6}f''(x_*)h^3 - \cdots \quad . \tag{33}$$

This shows that the local truncation error for the rectangle rule is $O(h^2)$ and identifies the leading error coefficient.

$$\begin{aligned} E &= \widehat{I} - I \\ &= \sum_{n=0}^{n-1} \widehat{I}_k - I_k \\ E &\sim -\sum_{k=0}^{n-1} \frac{1}{2}f'(x_k)h^2 - \sum_{k=0}^{n-1} \frac{1}{6}f''(x_k)h^3 - \cdots \quad . \end{aligned} \tag{34}$$

We sum over k and use simple inequalities to get the order of magnitude of the global error:

$$\begin{aligned} |E| &<\approx \frac{1}{2}\sum_{k=0}^{n-1} |f'(x_k)| \cdot h^2 \\ &\leq n \cdot \frac{1}{2} \max_{a \leq x \leq b} |f'(x)| \cdot h^2 \\ &= \frac{b-a}{h} O(h^2) \\ &= O(h) \ . \end{aligned}$$

This shows that the rectangle rule is first order accurate overall.

Looking at the global error in more detail leads to an asymptotic error expansion. Applying the rectangle rule error bound to another function, $g(x)$, we have

$$\sum_{k=0}^{n-1} g(x_k)h = \int_a^b g(x)dx + O(h) \quad .$$

Taking $g(x) = f'(x)$ gives

$$\sum_{k=0}^{n-1} f'(x_k)h = \int_a^b f'(x)dx + O(h) = f(b) - f(a) + O(h) \quad .$$

From (34) we have

$$\begin{aligned} E &\approx -\left(\sum_{k=0}^{n-1} f'(x_k)h\right)\frac{h}{2} \\ &\approx -\left(\int_a^b f'(x)dx\right)\frac{h}{2} \\ E &\approx -\frac{1}{2}\left(f(b) - f(a)\right)h \; . \end{aligned} \tag{35}$$

This gives the first term in the asymptotic error expansion. It shows that the leading error not only is bounded by h, but roughly is proportional to h. It also demonstrates the curious fact that if f is differentiable then the leading error term is determined by the values of f at the endpoints and is independent of the values of f between. This is not true if f has a discontinuity in the interval $[a, b]$.

To get the next term, apply (34) to the error itself, i.e.

$$\begin{aligned} \sum_{k=0}^{n-1} f'(x_k)h &= \int_a^b f'(x)dx - \frac{h}{2}\left(f'(b) - f'(a)\right) + O(h^2) \\ &= f(b) - f(a) - \frac{h}{2}\left(f'(b) - f'(a)\right) + O(h^2) \quad . \end{aligned}$$

In the same way, we find that

$$\sum_{k=0}^{n-1} f''(x_k)\frac{h^3}{6} = \left(f'(b) - f'(a)\right)\frac{h^2}{6} + O(h^3) \quad .$$

Combining all these gives the first two terms in the error expansion:

$$E(h) \sim -\frac{1}{2}\left(f(b) - f(a)\right)h + \frac{1}{12}\left(f'(b) - f'(a)\right)h^2 + \cdots \quad . \tag{36}$$

It is clear that this procedure can be used to continue the expansion as far as we want, but you would have to be very determined to compute, for example,

n	Computed Integral	Error	Error/h	$(E - A_1 h)/h^2$	$(E - A_1 h - A_2 h^2)/h^3$
10	3.2271	-0.2546	-1.6973	0.2900	-0.7250
20	3.3528	-0.1289	-1.7191	0.2901	-0.3626
40	3.4168	-0.0649	-1.7300	0.2901	-0.1813
80	3.4492	-0.0325	-1.7354	0.2901	-0.0907
160	3.4654	-0.0163	-1.7381	0.2901	-0.0453

Figure 6: Computational experiment illustrating the asymptotic error expansion for rectangle rule integration.

n	Computed Integral	Error	Error/h	$(E - A_1 h)/h^2$
10	7.4398e-02	-3.1277e-02	-3.1277e-01	-4.2173e-01
20	9.1097e-02	-1.4578e-02	-2.9156e-01	-4.1926e-01
40	9.8844e-02	-6.8314e-03	-2.7326e-01	-1.0635e-01
80	1.0241e-01	-3.2605e-03	-2.6084e-01	7.8070e-01
160	1.0393e-01	-1.7446e-03	-2.7914e-01	-1.3670e+00
320	1.0482e-01	-8.5085e-04	-2.7227e-01	-5.3609e-01
640	1.0526e-01	-4.1805e-04	-2.6755e-01	1.9508e+00
1280	1.0546e-01	-2.1442e-04	-2.7446e-01	-4.9470e+00
2560	1.0557e-01	-1.0631e-04	-2.7214e-01	-3.9497e+00
5120	1.0562e-01	-5.2795e-05	-2.7031e-01	1.4700e+00

Figure 7: Computational experiment illustrating the breakdown of the asymptotic expansion for a function with a continuous first derivative but discontinuous second derivative.

the coefficient of h^4. An elegant and more systematic discussion of this error expansion is carried out in the book of Dahlquist and Bjork. The resulting error expansion is called the *Euler McLaurin* formula. The coefficients 1/2, 1/12, and so on, are related to the *Bernoulli numbers.*

The error expansion (36) will not be valid if the integrand, f, has singularities inside the domain of integration. Suppose, for example, that $a = 0$, $b = 1$, $u = 1/\sqrt{2}$, and $f(x) = 0$ for $x \leq u$ and $f(x) = \sqrt{x - u}$ for $x \geq u$. In this case the error expansion for the rectangle rule approximation to $\int_0^1 f(x)dx$ has one valid term only. This is illustrated in Figure 7. The "Error/h" column shows that the first coefficient, A_1, exists. Moreover, A_1 is given by the formula (36). The numbers in the last column do not tend to a limit. This shows that the coefficient A_2 does not exist. The error expansion does not exist beyond the first term.

The analysis of the higher order integration methods listed in Figure 5 is easier if we use a symmetric basic panel. From now on, the panel of length h will have x_* in the center, rather at the left end, that is

$$P = [x_* - h/2, x_* + h/2] \quad .$$

If we now expand $f(x)$ in a Taylor series about x_* and integrate term by term, we get

$$\int_P f(x)dx = \int_{x=x_*-\frac{h}{2}}^{x_*+\frac{h}{2}} f(x)dx \sim f(x_*)h + \frac{f''(x_*)}{24}h^3 + \frac{f^{(4)}(x_*)}{384}h^5 + \cdots \ .$$

For the midpoint rule, this leads to a global error expansion in even powers of h, $E \approx A_1 h^2 + A_2 h^4 + \cdots$, with $A_1 = (f'(b) - f'(a))/24$. Each of the remaining panel methods is symmetric about the center of the panel. This implies that each of them has local truncation error containing only odd powers of h and global error containing only even powers of h.

The leading power of h in the error expansion is the order of accuracy. It can be determined by a simple observation: the order of the local truncation error is one more than the degree of the lowest monomial that is not integrated exactly by the panel method. For example, the rectangle rule integrates $f(x) = x^0 \equiv 1$ exactly but gets $f(x) = x^1 \equiv x$ wrong. The order of the lowest monomial not integrated exactly is 1 so the local truncation error is $O(h^2)$ and the global error is $O(h)$. The midpoint rule integrates x^0 and x^1 correctly but gets x^2 wrong. The order of the lowest monomial not integrated exactly is 2 so the local truncation error is $O(h^3)$ and the global error is $O(h^2)$. If the generic panel has x_* in the center, then

$$\int_P (x - x_*)^n \, dx$$

is always done exactly if n is odd. This is because both the exact integral and its panel method approximation are zero by symmetry.

To understand why this rule works, think of the Taylor expansion of $f(x)$ about the midpoint, x_*. This approximates f by a sum of monomials. Applying the panel integral approximation to f is the same as applying the approximation to each monomial and summing the results. Moreover, the integral of a monomial $(x - x_*)^n$ over P is proportional to h^{n+1}, as is the panel method approximation to it, regardless of whether the panel method is exact or not. The first monomial that is not integrated exactly contributes something proportional to h^{n+1} to the error.

Using this rule it is easy to determine the accuracy of the approximations in Figure 5. The trapezoid rule integrates constants and linear functions exactly, but it gets quadratics wrong. This makes the local truncation error third order and the global error second order. The Simpson's rule coefficients 1/6 and 2/3 are designed exactly to integrate constants and quadratics exactly, which they do. Simpson's rule integrates cubics exactly (by symmetry) but gets quartics wrong. This gives Simpson's rule fourth order global accuracy. The two point Gauss quadrature also does constants and quadratics correctly but quartics wrong (check this!). The three point Gauss quadrature rule does constants, quadratics, and quartics correctly but gets $(x - x_*)^6$ wrong. That makes it sixth order accurate.

5 The method of undetermined coefficients

The method of undetermined coefficients is a general way to find an approximation formula of a desired type. Suppose we want to estimate some A in terms of given data $g_1(h)$, $g_2(h)$, ... The method is to assume a linear estimation formula of the form

$$\widehat{A}(h) = a_1(h)g_1(h) + a_2(h)g_2(h) + \cdots \ , \tag{37}$$

then determine the unknown coefficients $a_k(h)$ by matching Taylor series up to the highest possible order. The coefficients often take the form of a constant times some power of h: $a_k(h) = a_k h^{p_k}$. The algebra is simpler if we guess or figure out the powers first. The estimator is *consistent* if $\widehat{A}(h) - A \to 0$ as $h \to \infty$. Generally (but not always), being consistent is the same as being at least first order accurate. At the end of our calculations, we may discover that there is no consistent estimator of the desired type.

We illustrate the method in a simple example: estimate $f'(x)$ from $g_1 = f(x)$ and $g_2 = f(x+h)$. As above, we will leave out the argument x whenever possible and write f for $f(x)$, f' for $f'(x)$, etc. The estimator is (dropping the x argument)

$$f' \approx \widehat{A} = a_1(h)f + a_2(h)f(x+h) \ .$$

Now expand in Taylor series:

$$f(x+h) = f + f'h + \frac{1}{2}f'' + \cdots \ .$$

The estimator is

$$\widehat{A} = a_1(h)f + a_2(h)f + a_2(h)f'h + a_2(h)f''h^2 + \cdots \ . \tag{38}$$

Looking at the right hand side, we see various coefficients, f, f', and so on. Since the relation is supposed to work whatever the values of f, f', etc. may be, we choose a_1 and a_2 so that the coefficient of f is zero. From (38), this leads to

$$0 = a_1(h) + a_2(h) \ .$$

To make the estimator consistent, we try

$$1 = a_2(h)h \ .$$

These two conditions lead to

$$a_2 = \frac{1}{h} \ , \quad a_1 = \frac{-1}{h} \ , \tag{39}$$

so the estimate is

$$\begin{aligned} f'(x) \ \approx \ \widehat{A} \ &= \ \frac{-1}{h}f(x) + \frac{1}{h}f(x+h) \\ &= \ \frac{f(x+h) - f(x)}{h} \ . \end{aligned}$$

This is the first order one sided difference approximation we saw earlier. Plugging the values (39) into (38) shows that the estimator satisfies $\widehat{A} = f' + O(h)$, which is the first order accuracy we found before.

A more complicated problem is to estimate $f'(x)$ from $f(x)$, $f(x-h)$, $f(x+h)$, $f(x+2h)$. This is not centered nor is it completely one sided, but it is biased to one side. It has proven useful in high accuracy wave simulations. This time we guess that all the coefficients have a power $1/h$, as all the estimates of f' so far have this property. Thus assume the form:

$$f' \approx \widehat{A} = \frac{1}{h}\left(a_{-1}f(x-h) + a_0 f + a_1 f(x+h) + a_2 f(x+2h)\right) \quad .$$

The Taylor series expansions are

$$\begin{array}{ccccccccccc} f(x-h) & = & f & - & f'h & + & \frac{f''}{2}h^2 & - & \frac{f'''}{6}h^3 & + & \frac{f^{(4)}}{24}h^4 + \cdots \\ f(x+h) & = & f & + & f'h & + & \frac{f''}{2}h^2 & + & \frac{f'''}{6}h^3 & + & \frac{f^{(4)}}{24}h^4 + \cdots \\ f(x+2h) & = & f & + & 2f'h & + & 2f''h^2 & + & \frac{4f'''}{3}h^3 & + & \frac{2f^{(4)}}{3}h^4 + \cdots \end{array}$$

Equating powers of h turns out to be the same as equating the coefficients of f, f', etc. from both sides:

$$\begin{array}{rl} f\ ,\ \ O(h^{-1})\ : & 0 = a_{-1} + a_0 + a_1 + a_2 \\ f'\ ,\ \ O(h^{0})\ : & 1 = -a_{-1} + a_1 + 2a_2 \\ f''\ ,\ \ O(h^{1})\ : & 0 = \frac{1}{2}a_{-1} + \frac{1}{2}a_1 + 2a_2 \\ f'''\ ,\ \ O(h^{2})\ : & 0 = \frac{-1}{6}a_{-1} + \frac{1}{6}a_1 + \frac{4}{3}a_2 \end{array} \tag{40}$$

We could compute the $O(h^3)$ equation but already we have four equations for the four unknown coefficients. If we would use the $O(h^3)$ equation in place of the $O(h^2)$ equation, we loose an order of accuracy in the resulting approximation.

These are a system of 4 linear equations in the four unknowns a_{-1} through a_2, which we solve in an ad hoc way. Notice that the combination $b = -a_{-1} + a_1$ appears in the second and fourth equations. If we substitute b, these equations are

$$1 = b + 2a_2\ ,$$

$$0 = \frac{1}{6}b + \frac{4}{3}a_2\ .$$

which implies that $b = -8a_2$ and then that $a_2 = -\frac{1}{6}$ and $b = \frac{4}{3}$. Then, since $-4a_2 = \frac{2}{3}$, the third equation gives $a_{-1} + a_1 = \frac{2}{3}$. Since $b = \frac{4}{3}$ is known, we get two equations for a_{-1} and a_1:

$$a_1 - a_{-1} = \frac{4}{3}\ ,$$

$$a_1 + a_{-1} = \frac{2}{3}\ .$$

The solution is $a_1 = 1$ and $a_{-1} = \frac{-1}{3}$. With these, the first equation leads to $a_0 = \frac{-1}{2}$. Finally, our approximation is

$$f'(x) = \frac{1}{h}\left(\frac{-1}{3}f(x-h) - \frac{1}{2}f(x) + f(x+h) - \frac{1}{6}f(x+2h)\right) + O(h^3) \; .$$

Note that the first step in this derivation was to approximate f by its Taylor approximation of order 3, which would be exact if f were a polynomial of order 3. The derivation has the effect of making $\widehat{A}$ exact on polynomials of degree 3 or less. The four equations (40) arise from asking $\widehat{A}$ to be exact on constants, linear functions, quadratics, and cubics. We illustrate this approach with the problem of estimating $f''(x)$ as accurately as possible from $f(x)$, $f(x+h)$, $f'(x)$ and $f'(x+h)$. The estimator we seek has the form

$$f'' \approx \widehat{A} = af + bf(x+h) + cf' + df'(x+h) \; .$$

We can determine the four unknown coefficients a, b, c, and d by requiring the approximation to be exact on constants, linears, quadratics, and cubics. It does not matter what x value we use, so let us take $x = 0$. This gives, respectively, the four equations:

$$\begin{array}{ll}
0 = a + b & (\text{constants, } f = 1) \; , \\
0 = bh + c + d & (\text{linears, } f = x) \; , \\
1 = b\dfrac{h^2}{2} + dh & (\text{quadratics, } f = x^2/2) \; , \\
0 = b\dfrac{h^3}{6} + d\dfrac{h^2}{2} & (\text{cubics, } f = x^3/6) \; .
\end{array}$$

Solving these gives

$$a = \frac{-6}{h^2} \;\; , \quad b = \frac{6}{h^2} \;\; , \quad c = \frac{-4}{h} \;\; , \quad d = \frac{-2}{h} \;\; .$$

and the approximation

$$f''(x) \approx \frac{6}{h^2}\big(-f(x) + f(x+h)\big) - \frac{2}{h}\big(2f'(x) + f'(x+h)\big) \; .$$

A Taylor series calculation shows that this is second order accurate.

6 Adaptive parameter estimation

In most real computations, the computational strategy is not fixed in advance, but is adjusted *adaptively* as the computation proceeds. If we are using one of the approximations $\widehat{(A)}(h)$, we might not know an appropriate h when we write the program, and the user might not have the time or expertise to choose h for each application. For example, exercise 12 involves hundreds or thousands of numerical integrations. It is out of the question for the user to experiment manually to find a good h for each one. We need instead a systematic computational procedure for finding an appropriate step size.

Suppose we are computing something about the function f, a derivative or an integral. We want a program that takes f, and a desired level of accuracy[10], e, and returns $\widehat{A}$ with $\left|\widehat{A} - A\right| \leq e$ with a high degree of confidence. We have $\widehat{A}(h)$ that we can evaluate for any h, and we want an automatic way to choose h so that $\left|\widehat{A}(h) - A\right| \leq e$. A natural suggestion would be to keep reducing h until the answer stops changing. We seek a quantitative version of this.

Asymptotic error expansions of Section 3 give one approach. For example, if $\widehat{A}(h)$ is a second order accurate approximation to an unknown A and h is small enough we can estimate the error using the leading term:

$$E(h) = \widehat{A}(h) - A \approx A_1 h^2 \; .$$

We can estimate $A_1 h^2$ from $\widehat{A}(h)$ and $\widehat{A}(2h)$ using the ideas that give (28). The result is the Richardson extrapolation error estimate

$$E(h) \approx A_1 h^2 \approx \frac{1}{3}\big(\widehat{A}(2h) - \widehat{A}(h)\big) \; . \tag{41}$$

The adaptive strategy would be to keep reducing h by a factor of two until the estimated error (41) is within the tolerance[11]:

```
for  (
  evaluate $\widehat{A}(2h)$ and $\widehat{A}(h)$ ;          // initialize
  $\left|\widehat{A}(2h) - \widehat{A}(h)\right| \geq 3\epsilon$ ;                // stopping test        (42)
  { h = h/2;  evaluate $\widehat{A}(h)$ } ;     // increment
) ;
```

A natural strategy might be to stop when $\left|\widehat{A}(2h) - \widehat{A}(h)\right| \leq e$. Our quantitative asymptotic error analysis shows that this strategy is off by a factor of 3. We achieve accuracy roughly e when we stop at $\left|\widehat{A}(2h) - \widehat{A}(h)\right| \leq 3e$. This is because $\widehat{A}(h)$ is more accurate than $\widehat{A}(2h)$.

We can base reasonably reliable software on refinements of the basic strategy (42). Some drawbacks of (42) are that

1. It needs an *initial guess*, a starting value of h.

2. It may be an infinite loop.

3. It might terminate early if the initial h is outside the *asymptotic range* where error expansions are accurate.

4. If $\widehat{A}(h)$ does not have an asymptotic error expansion, the program will not detect this.

[10] This is absolute error. We also could seek a bound on relative error: $\left|\widehat{A} - A\right| / |A| \leq \epsilon$.

[11] In the increment part we need not evaluate $\widehat{A}(2h)$ because this is what we called $\widehat{A}(h)$ before we replaced h with $h/2$.

5. It does not return the best possible estimate of A.

A plausible initial guess, h_0, will depend on the scales (length or time, etc.) of the problem. For example 10^{-10} meters is natural for a problem in atomic physics but not in airplane design. The programmer or the user should supply h_0 based on understanding of the problem. The programmer can take $h_0 = 1$ if he or she thinks the user will use natural units for the problem (Ångströms for atomic physics, meters for airplanes). It might happen that you need $h = h_0/1000$ to satisfy (42), but you should give up if $h = h_0 \cdot \epsilon_{\text{mach}}$. For integration we need an initial $n = (b-a)/h$. It might be reasonable to take $n_0 = 10$, so that $h_0 = (b-a)/10$.

Point 2 says that we need some criterion for giving up. As discussed more in Section 7 we should anticipate the ways our software can fail and report failure. When to give up should depend on the problem. For numerical differentiation, we can stop when roundoff or propagated error from evaluating f (see Chapter **??**, Section?) creates an error as big as the answer. For integration limiting the number of refinements to 20, would limit the number of panels to $n_0 \cdot 2^{20} \approx n_0 \cdot 10^6$. The revised program might look like

```
h = h0;
hMin = 10*macheps*h0;   //macheps = machine precision
while  $\left(\left|\widehat{A}(2h) - \widehat{A}(h)\right| \geq 3e\right)$
  if ( h <= hMin ) {                                          (43)
    Print an error message.
    errorCode = HMIN_REACHED; return -1;}
  $h = h/2$ ;
return  $A(h)$ ;
```

```
h = h0;
hMin = 10*macheps*h0;   //macheps = machine precision
for  (
  evaluate $\widehat{A}(2h)$ and $\widehat{A}(h)$ ;           // initialize
  $\left|\widehat{A}(2h) - \widehat{A}(h)\right| \geq 3\epsilon$ ;               // stopping test
  { h = h/2;  evaluate $\widehat{A}(h)$ } ;     // increment              (44)
 )  {
     if ( h <= hMin ) {
     Print an error message;
     errorCode = HMIN_REACHED;
     return -1;}
    }
```

We cannot have perfect protection from point 3, though *premature termination* is unlikely if h_0 is sensible and e (the desired accuracy) is small enough. A more cautious programmer might do more convergence analysis, for example asking that the $\widehat{A}(4h)$ and $\widehat{A}(2h)$ error estimate be roughly 2^p times larger than the $\widehat{A}(2h)$ and $\widehat{A}(h)$ estimate. There might be irregularities in $f(x)$, possibly jumps in some derivative, that prevent the asymptotic error analysis but do not prevent convergence. It would be worthwhile returning an error flag in this case, as some commercial numerical software packages do.

Part of the risk in point 4 comes from the possibility that $\widehat{A}(h)$ converges more slowly than the hoped for order of accuracy suggests. For example if $\widehat{A}(h) \approx A + A_1 h^{1/2}$, then the error is three times that suggested by (42). The extra convergence analysis suggested above might catch this.

Point 5 is part of a paradox afflicting many error estimation strategies. We estimate the size of $E(h) = \widehat{A}(h) - A$ by estimating the value of $E(h)$. This leaves us a choice. We could ignore the error estimate (41) and report $\widehat{A}(h)$ as the approximate answer, or we could subtract out the estimated error and report the more accurate $\widehat{A}(h) - \widehat{E}(h)$. This is the same as applying one level of Richardson extrapolation to $\widehat{A}$. The corrected approximation probably is more accurate, but we have no estimate of its error. The only reason to be dissatisfied with this is that we cannot report an answer with error less than e until the error is far less than e.

7 Software

There are several things a scientific programmer can do to make codes easier to debug and more reliable. Everyone has had the experience of *breaking* a code, making a change that for some reason makes the whole thing stop working. A program that is designed to be changed is *flexible*, less likely to be broken in this way. *Modular* programs have different pieces in separate procedures (methods, subroutines) that can be tested separately. Many of the modules can be tested using *convergence analysis* described in Section 3.2. Programmers should go far as possible to prevent *silent failure*. It is better to have no answer than a wrong one.

7.1 Flexible programming

Suppose you want to compute $I = \int_0^2 f(x)dx$ using a panel method. The rectangle rule $\widehat{I} = \Delta x \sum_{k=0}^{n-1} f(x_k)$ with $n = 100$ could be coded:

```
double I = 0;                     // line 1
for ( int k = 0; k < 100; k++)    // line 2
   I += .02*f(.02*k);             // line 3
```

Here the number $n = 100$ is *hard wired*, which means built into the code in a way that makes it hard to change on a whim. It would be easy to break this code by changing line 2 to `for ( int k = 0; k < 90; k++)` but forgetting to change .02 to $2/90 \approx .0222$, or changing it in only one place: `I += (2/90)*f(.02*k);`. By the way, this last has the bug that `(2/90)` evaluates to zero because it is an integer divide.

A more flexible version would be:

```
int n    = 100;   // The number of points
double a = 0;     // The left endpoint for integration
double b = 2;     // The right endpoint for integration
double dx = ( b - a ) / n;   // Width of a panel

double I = 0;                    // line 1
double x;
for ( int k = 0; k < 100; k++) { // line 2
   x  = k*dx;
   I += dx*f(x);                // line 3
  }
```

Changing to `int n = 90;` would give a correct program. Because the variable `dx` has a name, you can check in the debugger that it has the correct value. The more flexible version has a few more lines of code which take a few extra seconds to type.

You probably will want to test each module of your program on a problem you know the answer to. For example, if you are calculating $H(T) = \int_0^T e^{\sin(x)}dx$, you would code it in a way that it is easy to apply to $G(T) = \int_0^T e^x dx$.

7.2 Modular programming

Modular programming is helpful for any software project. In scientific computing we design procedures that can be tested separately on their own test problems. For example, if we apply adaptive Richardson extrapolation as in Section 6 for an integral, we would write a generic adaptive Richardson extrapolation program and test it on data that does not come from an integrator, then we would write an integrator and test it outside the Richardson procedure. If the two procedures work correctly separately, they have a good chance to work well together.

Designing scientific software, or any software, involves more than choosing data structures and procedure interfaces. You also need a testing plan. The code should be designed so that the modules can be tested separately. For example, even if I know my production integration code will use $n = 50$ panels, I might put the number of panels as a calling argument so I can do a convergence study.

7.3 Report failure

Error handling is another indispensable part of software design. Any module that can fail (most can) must have a way to report failure. The designer will choose an appropriate mechanism. The simplest is just to print something and stop the program when something goes wrong. This is not appropriate for commercial software but could be fine for a research code that only the author will run. More sophisticated might be to write to a log file and return an error flag, or even to throw an exception that could be caught by the calling routine.

Equally important is detecting failure. This means checking and forwarding all the internal failure reporting, such as:

```
double *vec;                     // Treat vec as an array.
vec = new double[n];             // Allocate memory for n values
if ( vec == NIL ) .. REPORT ..   // Complain if didn’t work.
```

It also means checking calling arguments for plausibility:

```
#define MAXN 10000  /* Largest allowed # of panels */
int Integrate(                // Integrate f(x)
   double (*f)(double),       // supply f implicitly
   double  *I,                // return the estimated integral
   int      n) {              // using n panels.

   if ( n <= 0 ) {
      .. complain that n is too small ..;
      return 1;  // The error flag for negative n.
   if ( n > MAXN ) {
      cout << "In Integrate, got n = " << n << " greater than"
           << " MAXN = << MAXN << endl;
      return 2; // The error flag for n too large.

    . . .do the integral . . .
   *I = . . . answer . . .
   return 0;  //  error flag = 0 means it worked.
 }
```

Of course, this only works if the calling program checks the error flag, i.e. `eFlag = Integrate( f, @I, n); if (eFlag) ..  WRONG ANSWER ..  ;`

Lastly, it means thinking of all loops that could be infinite loops, such as

```
while ( error > targetError ){ // try to get error < targetError

  ...  make the solution more accurate...;

 }
```

This could be an infinite loop if `targetError` is too small or the asymptotic error expansion is wrong. Instead, at least put in a trip counter

```
#define MAX_TRIP_COUNT 1000  /* stop a runaway refinement loop  */
int tripCount = 0;
while ( error > targetError )  {

  ...  make the solution more accurate...;

if ( ++ tripCount > MAX_TRIP_COUNT ) report error & quit.
}
```

8 References and further reading

For a review of one variable calculus, I recommend the Schaum outline. The chapter on Taylor series explains the remainder estimate clearly.

There several good old books on classical numerical analysis. Two favorites are *Numerical Methods* by Germund Dahlquist and Åke Björk, and *Analysis of Numerical Methods* by Gene Isaacson and Herb Keller. Particularly interesting subjects are the symbolic calculus of finite difference operators and Gaussian quadrature.

There are several applications of convergent Taylor series in scientific computing. One example is the *fast multipole method* of Leslie Greengard and Vladimir Rokhlin.

9 Exercises

1. Verify that (23) represents the leading error in the approximation (22). *Hint*, this does not require multidimensional Taylor series. Why?

2. Use multidimensional Taylor series to show that the *rotated* five point operator

$$\frac{1}{2h^2}\big(f(x+h,y+h)+f(x+h,y-h)+f(x-h,y+h)+f(x-h,y-h)-4f\big)$$

   is a consistent approximation to $\triangle f$. Use a symmetry argument to show that the approximation is at least second order accurate. Show that the leading error term is

$$\frac{h^2}{12}\left(\partial_x^4 f + 6\partial_x^2\partial_y^2 f + \partial_y^4 f\right) \, .$$

3. What coefficient should we use in place of $\frac{4}{3}$ if $\widehat{A}(h)$ is first order accurate? Find the coefficient as a function of p, the order of accuracy.

4. Find a formula that estimates $f''(x)$ using the four values $f(x)$, $f(x+h)$, $f(x+2h)$, and $f(x+3h)$ with the highest possible order of accuracy. What is this order of accuracy? For what order polynomials does the formula give the exact answer?

5. Suppose we have panels P_k as in (31) and panel averages $F_k = \int_{P_k} f(x)dx/(x_{k+1} - x_k)$.

   (a) What is the order of accuracy of F_k as an estimate of $f((x_k + x_{k+1})/2) = f(x_{k+1/2})$?

   (b) Assuming the panels all have size h, find a higher order accurate estimate of $f(x_{k+1/2})$ using F_k, F_{k-1}, and F_{k+1}.

6. This discusses the function (12) more carefully.

(a) Show that the Taylor series for $g(x) = 1/(1-x)$ about $x = 0$ is the geometric series $1/(1-x) = \sum_{n=0}^{\infty} x^n$, which converges for $|x| < 1$.

(b) Prove the simple remainder formula $g(x) = \sum_{n=0}^{p} x^n + R_p(x)$ with $R_p(x) = x^{p+1}/(1-x)$. Hint: factor x^{p+1} out of $R_p(x) = \sum_{n=p+1}^{\infty} x^n$.

(c) Use the formula $-h\partial_x e^{-x/h} = e^{-x/h}$ to prove that $\int_0^{\infty} e^{-x/h} x^n dx = nh \int_0^{\infty} e^{-x/h} x^{n-1} dx$, and then $\int_0^{\infty} e^{-x/h} x^n dx = n! h^{n+1}$.

(d) Use the same integration by parts and (11) to show that $\int_{1/2}^{\infty} e^{-x/h} x^n dx = O(x^p)$ for any p and n.

(e) Note that for $0 \leq x \leq 1/2$, $1/(1-x) \leq 2$. Use this and part b to prove the remainder bound

$$\begin{aligned} \int_0^{1/2} e^{-x/h} R_p(x) dx &\leq 2 \int_0^{1/2} e^{-x/h} x^{p+1} dx \\ &\leq 2 \int_0^{\infty} e^{-x/h} x^{p+1} dx = O(h^{p+1}) \; . \end{aligned}$$

(f) Show that $\int_0^{1/2} e^{-x/h} x^n dx = \int_0^{\infty} e^{-x/h} x^n dx + O(x^p)$ for any n and p.

(g) Conclude that (13) indeed is a valid asymptotic expansion.

7. An application requires accurate values of $f(x) = e^x - 1$ for x very close to zero.

(a) Show that the problem of evaluating $f(x)$ is well conditioned for small x.

(b) How many digits of accuracy would you expect from the code `f = exp(x) - 1;` for $x \sim 10^{-5}$ and for $x \sim 10^{-10}$ in single and in double precision?

(c) Let $f(x) = \sum_{n=1}^{\infty} f_n x^n$ be the Taylor series about $x = 0$. Calculate the first three terms, the terms involving x, x^2, and x^3. Let $p(x)$ be the degree three Taylor approximation of $f(x)$ about $x = 0$.

(d) Assuming that x_0 is so small that the error is nearly equal to the largest neglected term, estimate $\max |f(x) - p(x)|$ when $|x| \leq x_0$.

(e) We will evaluate $f(x)$ using

```
if ( abs(x) > x0 ) f = exp(x) - 1;
else               f = p3(x); // given by part c.
```

What x_0 should we choose to maximize the accuracy of $f(x)$ for $|x| < 1$ assuming double precision arithmetic and that the exponential function is evaluated to full double precision accuracy (exact answer correctly rounded)?

8. Suppose that $f(x)$ is a function that is evaluated to full machine precision but that there is ϵ_{mach} rounding error in evaluating $\widehat{A} = (f(x+h) - f(x))/h$. What value of h minimizes the total error including both rounding and truncation error? This will be $h_*(\epsilon_{mach}) \sim \epsilon^q_{mach}$. Let $e_*(\epsilon_{mach})$ be the resulting best estimate of $f'(x)$. Show that $e_* \sim \epsilon^r_{mach}$ and find r.

9. Repeat Exercise 8 with the two point centered difference approximation to $f'(x)$. Show that the best error possible with centered differencing is much better than the best possible with the first order approximation. This is one of the advantages of higher order finite difference approximations.

10. Verify that the two point Gauss quadrature formula of Figure 5 is exact for monomials of degree less than six. This involves checking the functions $f(x) = 1$, $f(x) = x^2$, and $f(x) = x^4$ because the odd order monomials are exact by symmetry. Check that the three point Gauss quadrature formula is exact for monomials of degree less than 8.

11. Find the replacement to adaptive halting criterion (41) for a method of order p.

12. We want to know how the function,

$$f(t) = \int_0^1 \cos\left(tx^2\right) dx \ \ , \tag{45}$$

behaves for large t. There is an approximation,

$$f(t) \sim \sqrt{\frac{\pi}{8t}} + \frac{1}{2t}\sin(t) - \frac{1}{16t^2}\cos(t) + \cdots \ \ , \tag{46}$$

that is supposed to hold for large t. We want to know how accurate this approximation is and how large t has to be before it is useful. This exercise goes through some of the steps that go into creating scientific software to investigate this question numerically: (a) Design the basic code to be modular and robust, (b) Check that is gives the right answer in one case you can check (note: not the problem of interest), (c) Check that the basic computational method gives the accuracy it should. (d) Write an automatic adaptive version of the code that allows it to produce accurate answers *hands off*, without manual tuning of computational parameters (Δx in this case) for each run. (e) Do the science. To appreciate the value of part (d), the science step involves thousands of integrations to evaluate $f(t)$ for various t values. Nobody wants to do a thousand runs choosing Δx by hand for each one.

    (a) Write a procedure (or "method") to estimate $f(t)$ using a panel integration method with uniformly spaced points. The procedure should be well documented, robust, and clean. Robust will mean many things in later exercises. Here is means: (i) that it should use the correct number of panels even though the points x_k are computed in inexact floating point arithmetic, and (ii) that the procedure will return an error code and possibly print an error message if one of the calling arguments is out of range (here, probably just $n \leq 0$). It should take as inputs t and n and return the approximate integral with that t, and $\Delta x = 1/n$. If your panel integration formula uses endpoints of the panel, you must write the code so that $f(x_k)$ is evaluated only once. This routine should be written so that another person could easily substitute a different panel method or a different integrand by changing a few lines of code.

(b) Check the correctness of the procedure from part (a) by seeing whether it gives the right answer for small t. We can estimate $f(t)$ for small t using a few terms of its Taylor series. From $\cos(u) \approx 1 - \frac{1}{2}u^2 + \frac{1}{24}u^4 + \cdots$, we get $\cos(tx^2) \approx 1 - \frac{1}{2}t^2x^4 + \cdots$. Integrating this approximation over the x interval gives $f(t) \approx 1 - \frac{1}{10}t^2 + \cdots$. Compute the next few terms and use them to get an approximate value of $f(t)$ for small t. Then write a *driver* program that calls both the integration procedure, and the procedure that approximates f using Taylor series, and compares the results. Use reasonable but not huge values of n.

(c) With $t = 1$, do a convergence study to verify the second order accuracy of the trapezoid rule and the fourth order accuracy of Simpson's rule. This requires you to write a different driver to call the integration procedure with several values of n and compare the answers in the manner of a convergence study. Once you have done this for the trapezoid rule, it should take less than a minute to redo it for Simpson's rule. This is how you can tell whether you have done part (a) well.

(d) Write a procedure that uses the basic integration procedure from part (a), together with Richardson error estimation to find an n that gives $f(t)$ to within a specified error tolerance. The procedure should work by repeatedly doubling n until the estimated error, based on comparing approximations, is less than the tolerance given. This routine should be robust enough to quit and report failure if it is unable to achieve the requested accuracy. The input should be t and the desired error bound. The output should be the estimated value of f, the number of points used, and an error flag to report failure. Before applying this procedure to the panel integration procedure, apply it to the fake procedure `fakeInt.c` or `fakeInt.C`. Note that these testers have options to make the Richardson program fail or succeed. Try it both ways, to make sure the robustness feature of your Richardson procedure works. Include with your homework output illustrating the behavior of your Richardson procedure when it fails.

(e) Here is the science part of the problem. Make a few plots showing f and its approximations using one, two and all three terms on the right side of (46) for t in the range $1 \leq t \leq 1000$. In all cases we want to evaluate f so accurately that the error in our f value is much less than the error of the approximation (46). Note that even for a fixed level of accuracy, more points are needed for large t. Plot the integrand to see why.