PY 502 - Computational Physics

Lecture notes.

1 Basic numerical algorithms: differentiation and integration.

1.1 Numerical differentiation.

We will assume that the function $f(x)$ to be differentiated is continuous with as many continuous derivatives as needed. From the Taylor series expansion

$$f(x+\delta) = f(x) + f'(x)\,\delta + \frac{1}{2}f''(x)\,\delta^2 + \frac{1}{6}f'''(x)\,\delta^3 + \frac{1}{24}f^{iv}(x)\,\delta^4 + \dots \quad (1)$$

we see that

$$f'(x) = \frac{f(x+\delta) - f(x)}{\delta} + O(\delta) \quad (2)$$

Thus the first term in the r.h.s., namely $[f(x+\delta) - f(x)]/\delta$, with small δ, can be used as an approximation to the derivative $f'(x)$. This is hardly surprising, given the mathematical definition of the derivative. This approximation is called the "forward difference approximation" to the derivative. Equation (1) also tell us that the error in the approximation is of order δ.

In a similar manner, from the expansion

$$f(x-\delta) = f(x) - f'(x)\,\delta + \frac{1}{2}f''(x)\,\delta^2 - \frac{1}{6}f'''(x)\,\delta^3 + \frac{1}{24}f^{iv}(x)\,\delta^4 + \dots \quad (3)$$

we obtain the "backward difference approximation" to the derivative

$$f'(x) \approx \frac{f(x) - f(x-\delta)}{\delta} \quad (4)$$

The error is again $O(\delta)$. We can get a better approximation subtracting Eq. (3) from Eq. (1), which gives

$$f(x+\delta) - f(x-\delta) = 2f'(x)\,\delta + \frac{1}{3}f'''(x)\,\delta^3 + \dots \quad (5)$$

From this equation we see that the so called "central difference approximation" to the derivative, namely

$$f'(x) \approx \frac{f(x+\delta) - f(x-\delta)}{2\delta} \ , \tag{6}$$

has an error of order δ^2.

The magnitude of the error is important, because one cannot reduce it arbitrarily simply by decreasing the step δ. We must remember that all calculations are affected by the finite precision of the computer arithmetic. Thus the error in the forward (or backward) difference approximations will not just be given by a term proportional to δ but will include also a term due to the round-off errors in the difference $f(x+\delta) - f(x)$. Thus the actual error, to leading order, will be given by

$$\Delta = \alpha\delta + \frac{\beta\epsilon}{\delta} \tag{7}$$

where α and β are some constants and we denoted by ϵ the magnitude of the round-off error. In order to estimate the minimum value for the error, we should assume that α and β have the same sign, e.g. positive. Otherwise there will be some value of δ which gives origin to a zero error by pure coincidence. With $\alpha > 0$ and $\beta > 0$ Δ is minimum at $\delta = \sqrt{\beta\epsilon/\alpha}$, where it takes value $\Delta = 2\sqrt{\alpha\beta\epsilon}$. If the function and its derivative are of the same order of magnitude, the two constants α and β will have comparable values and the optimal step will be approx. $\sqrt{\epsilon}$, where ϵ is the error due to the round-off in the mantissa of floating point numbers, namely $\approx 10^{-7}$ and $\approx 10^{-15}$ for single and double precision respectively. Correspondingly the best value for the step are 10^{-3}-10^{-4} and 10^{-7}-10^{-8}, respectively. The corresponding relative error in the approximation of the derivative is of the same order of magnitude. Of course, if function and derivative do not have the same order of magnitude (if the function is very flat or very steep), the optimal step should be readjusted to values which can be easily inferred from the results above or from simple rescaling arguments (change variable to $y = cx$ in such a way to make f and f' comparable).

Applying similar arguments to the central difference approximation one finds that the optimal step is proportional to $\epsilon^{\frac{1}{3}}$ and the error to $\epsilon^{\frac{2}{3}}$. Thus the optimal step will be larger than for the forward or backward approximations, but the actual error will be smaller.

If we add Eqs. (1) and (1) we obtain

$$f(x+\delta)+f(x-\delta)=2f(x)+f''(x)\,\delta^2+\frac{1}{12}f^{iv}(x)\,\delta^4+\ldots \tag{8}$$

or

$$f''(x)=\frac{f(x+\delta)+f(x-\delta)-2f(x)}{\delta^2}+O(\delta^2) \tag{9}$$

This shows that the first term in the r.h.s., namely $[f(x+\delta)+f(x-\delta)-2f(x)]/\delta^2$ can be taken as an approximation to the second derivative, with an error of order δ^2. This is called the "central difference approximation" to the second derivative. It is a convenient and reasonably accurate approximation, which is adequate for many numerical applications.

1.2 Numerical integration.

The derivation of formulae for numerical integration starts again from the Taylor series expansion of a function $f(x)$ around some point x_0:

$$f(x_0+x)=f(x_0)+f'(x_0)\,x+\frac{1}{2}f''(x_0)\,x^2+\frac{1}{6}f'''(x_0)\,x^3+\ldots \tag{10}$$

Dy integrating from x_0 to $x_1=x_0+\delta$ we get

$$\int_{x_0}^{x_1}f(x)dx=f(x_0)\delta+f'(x_0)\frac{\delta^2}{2}+O(\delta^3) \tag{11}$$

We can replace $f'(x_0)$ with its forward difference approximation since the $O(\delta)$ error in the approximation, combined with the $\delta^2/2$ factor, introduces an error of the same order of magnitude as the terms which we are neglecting already. This gives

$$\int_{x_0}^{x_1}f(x)dx=f(x_0)\delta+\frac{[f(x_1)-f(x_0)]}{\delta}\frac{\delta^2}{2}+O(\delta^3)=[f(x_0)+f(x_1)]\frac{\delta}{2}+O(\delta^3) \tag{12}$$

We derive a numerical integration formula for a whole interval between x_0 and x_n by dividing the interval into n subintervals of length $\delta=(x_n-x_0)/n$ and applying Eq. 12 to the individual subintervals. This gives

$$\int_{x_0}^{x_n}f(x)dx\approx\Big[\frac{1}{2}f(x_0)+f(x_1)+\ldots+f(x_{n-1})+\frac{1}{2}f(x_n)\Big]\delta \tag{13}$$

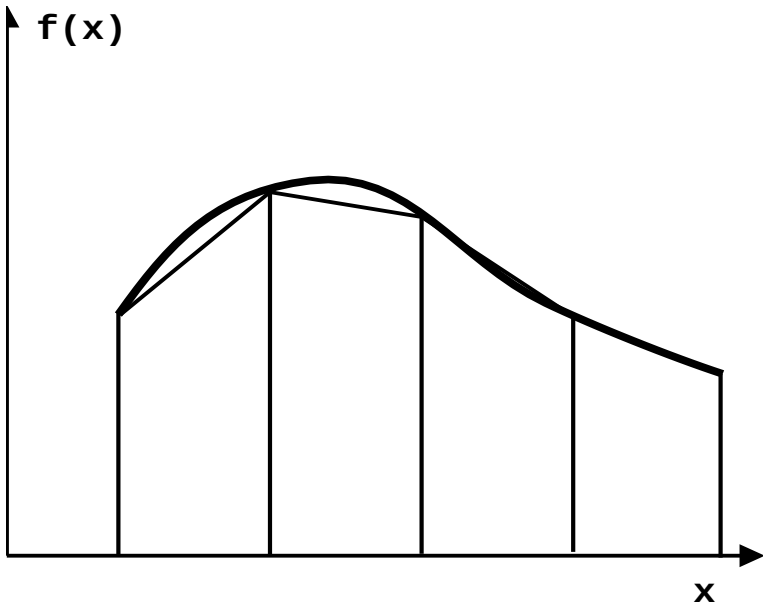

Figure 1: Illustration of the trapezoidal formula for numerical integration.

This is known as "the trapezoidal formula" for integration. It has this name because it is equivalent to replacing the function with its linear interpolation in each subinterval, as illustrated in Fig. 1. The overall error is of order δ^2 because the individual errors of order δ^3 add up with the same sign (see Fig. 1) over the regions where the function has a definite curvature (toward or away from the x-axis) and the number of subintervals in a region of finite extent is of order δ^{-1}. If we add Eq. 11 with the similar equation obtained by integrating from $x_{-1} = x_0 - \delta$ to x_0, namely

$$\int_{x_{-1}}^{x_0} f(x)dx = f(x_0)\delta - f'(x_0)\frac{\delta^2}{2} + O(\delta^3) \tag{14}$$

we obtain

$$\int_{x_{-1}}^{x_1} f(x)dx = 2f(x_0)\delta + O(\delta^3) \tag{15}$$

with no reference to $f'(x_0)$. This result is normally applied by taking x_{-1}, x_1 as the endpoints and x_0 as the midpoint of a subinterval. Combining the equations for all the n subintervals we obtain

$$\int_{x_0}^{x_n} f(x)dx \approx [f(x_{1/2}) + f(x_{3/2}) + \ldots + f(x_{n-3/2}) + f(x_{n-1/2})]\delta \tag{16}$$

where we denoted by $x_{i+1/2}$ the midpoints of the subintervals. This formula is known as "midpoint based trapezoidal formula". It also produces an overall error of order δ^2, for the same reasons as above.

A much more accurate formula can be obtained by keeping a few extra terms in Eqs. 11,14

$$\int_{x_0}^{x_1} f(x)dx = f(x_0)\delta + f'(x_0)\frac{\delta^2}{2} + f''(x_0)\frac{\delta^3}{6} + f'''(x_0)\frac{\delta^4}{24} + O(\delta^5) \qquad (17)$$

$$\int_{x_{-1}}^{x_0} f(x)dx = f(x_0)\delta - f'(x_0)\frac{\delta^2}{2} + f''(x_0)\frac{\delta^3}{6} - f'''(x_0)\frac{\delta^4}{24} + O(\delta^5) \qquad (18)$$

If we add these two equations, all the terms with odd derivatives cancel and we are left with

$$\int_{x_{-1}}^{x_1} f(x)dx = 2f(x_0)\delta + f''(x_0)\frac{\delta^3}{3} + O(\delta^5) \qquad (19)$$

We can now replace the second derivative with its central difference approximation without making the error any worse. We thus obtain

$$\begin{aligned}\int_{x_{-1}}^{x_1} f(x)dx &= 2f(x_0)\delta + \frac{[f(x_1) + f(x_{-1}) - 2f(x_0)]}{\delta^2}\frac{\delta^3}{3} + O(\delta^5) \\ &= \Big[\frac{1}{3}f(x_{-1}) + \frac{4}{3}f(x_0) + \frac{1}{3}f(x_1)\Big]\delta + O(\delta^5) \qquad (20)\end{aligned}$$

This result is typically used by dividing an interval x_0–x_{2n} into $2n$ subintervals of length δ and applying Eqs. 20 to the pairs of subsequent subintervals. This produces the "Simpson formula:"

$$\begin{aligned}\int_{x_0}^{x_{2n}} f(x)dx \approx \Big[&\frac{1}{3}f(x_0) + \frac{4}{3}f(x_1) + \frac{2}{3}f(x_2) + \ldots \\ &+\frac{2}{3}f(x_{2n-2}) + \frac{4}{3}f(x_{2n-1}) + \frac{1}{3}f(x_{2n})\Big]\delta \qquad (21)\end{aligned}$$

Like for the trapezoidal formula, the errors over the individual pairs of subintervals generally add up, and the total error is of order $n\delta^5 = [(x_{2x} - x_0)/(2\delta)]\delta^5 = O(\delta^4)$. It is still a remarkably accurate formula. There is something intriguing in the alternation of the 2/3 and 4/3 factors that enter in the formula, but there is nothing magic in them. Indeed it is possible to derive a variation of the Simpson formula where all the intermediate weights are equal to 1 by compensating with a suitable arrangement of weights at the

end points. Also, with this variation one is no longer constrained to an even number of subintervals. The procedure is to add one-half of the formulae for the intervals x_0–x_{2n} and x_1–x_{2n-1} if the total number $N = 2n$ of subintervals is even or of the formulae for the intervals x_0–x_{2n} and x_1–x_{2n+1} if the total number $N = 2n + 1$ of subintervals is odd. This gives

$$\frac{1}{2}\int_{x_0}^{x_1} f(x)dx + \int_{x_1}^{x_{N-1}} f(x)dx + \frac{1}{2}\int_{x_{N-1}}^{x_N} f(x)dx = \Big[\frac{1}{6}f(x_0)$$
$$+f(x_1) + f(x_2) + \ldots + f(x_{N-2}) + f(x_{N-1}) + \frac{1}{6}f(x_N)\Big]\delta + O(\delta^5) \qquad (22)$$

One needs to complete this formulae by adding approximations for the missing $\frac{1}{2}\int_{x_0}^{x_1} f(x)dx$ and $\frac{1}{2}\int_{x_{N-1}}^{x_N} f(x)dx$ with an error of at most $O(\delta^4)$ (as in the full Simpson formula). This can be easily accompished by integrating Eq. 10 from x_0 to x_1 (and the corresponding equation for the last subinterval) and using the same Taylor series expansion to evaluate $f'(x_0)$ and $f''(x_0)$ in terms of $f(x_0)$, $f(x_1)$ and $f(x_2)$. We leave the details to the reader. The final result is the modified Simpson formula:

$$\int_{x_0}^{x_{2n}} f(x)dx = \Big[\frac{3}{8}f(x_0) + \frac{7}{6}f(x_1) + \frac{23}{24}f(x_2) + f(x_3) + \ldots$$
$$+f(x_{N-3}) + \frac{23}{24}f(x_{N-2}) + \frac{7}{6}f(x_{N-1}) + \frac{3}{8}f(x_N)\Big]\delta + O(\delta^4) \qquad (23)$$

This formula is very seldom used, but we learn from it that the weights can be redistributed without reducing the level of accuracy. This points to the fact that at the root of the precision of the integration formulae there is the continuity of the function and of a sufficient number of its derivatives. It is this continuity that gives global properties to the function that permit the redistribution of the weights.

1.3 End-point singularities.

We consider the case where the function to be integrated has integrable singularities. Without loss of generality, we can assume that the singularity occurs at one end of the interval of integration. (If there are singularities at both ends, we can split the integral into two integrals having a singularity at one end only. If a singularity occurs inside the range of integration, we can similarly split the integral into two integrals having the singularity at the end point.) Sometimes the singularity makes the straightforward application

of the integration formulae plainly impossible. This is the case, for instance, of a $1/\sqrt{x}$ singularity. With $f(x)$ behaving as

$$f(x) \sim \frac{1}{\sqrt{x - x_a}} \qquad x \to x_a \tag{24}$$

and otherwise regular in the interval x_a–x_b, the integral

$$I = \int_{x_a}^{x_b} f(x)\, dx \tag{25}$$

is well defined, but one cannot insert a value for $f(x_a)$ in the integration formulae. However, the problem goes beyond the fact that the value of the function at x_a may be ill defined. Even with a behavior

$$f(x) \sim \sqrt{x - x_a} \qquad x \to x_a \tag{26}$$

the straightforward use of the trapezoidal or Simpson's integration formula would lead to very poor results, as we will see in detail below. The point is that the accuracy of integration formulae results from the assumption that the function and a sufficient number of its derivatives are defined and continuous throughout the range of integration. When this property no longer holds, the application of the standard integration formulae leads to a much poorer approximation: the contribution from neighborhood of the singularity can produce an error much worse than the overall expected error.

There are two main methods for dealing with end-point singularities. One consists in separating the singularity with some cutoff value x_c: $x_a < x_c < x_b$. The integral is correspondingly subdivided into two integrals:

$$I = I_1 + I_2 = \int_{x_a}^{x_c} f(x)\, dx + \int_{x_c}^{x_b} f(x)\, dx \tag{27}$$

I_2, which contains no singularity, is approximated by a standard integration formula. In I_1 one approximates $f(x)$ with an expansion around the singularity that can be integrated analytically. For example, with $f(x) = \sqrt{\sin(x)}/x$, one could expand

$$\frac{\sqrt{\sin(x)}}{x} = \frac{(x - \frac{x^3}{6} + \ldots)^{\frac{1}{2}}}{x} = x^{-\frac{1}{2}} - \frac{1}{12} x^{\frac{3}{2}} + \ldots \tag{28}$$

and approximate

$$I_1 \approx \int_0^{x_c} \left[x^{-\frac{1}{2}} - \frac{1}{12} x^{\frac{3}{2}} \right] dx = 2 x_c^{\frac{1}{2}} - \frac{1}{30} x_c^{\frac{5}{2}} \tag{29}$$

One should make a judicious choice of x_c: sufficiently close to x_a that the error induced by the expansion is small, but far enough that the effects of the singularity on the integral from x_c to x_b is also small.

The other possibility consists in performing a change of variable of integration that removes the singularity. In the example above, we could introduce a new independent variable y with $x = y^2$. This gives

$$I = \int_0^{x_b} \frac{\sqrt{\sin(x)}}{x} dx = \int_0^{\sqrt{x_b}} \frac{2\sqrt{\sin(y^2)}}{y} dy \tag{30}$$

where the integrand has no singularity for $y = 0$. From a practical point of view, in the implementation of this method it will typically be convenient to maintain both variables x and y in the code, in order to keep the arithmetic expressions as simple as possible (in the loop over integration points y is assigned a value first, x is assigned its value in terms of y, etc.). Also, one will typically have to exert some care in calculating the value of the new integrand at the end point, which may involve some limiting procedure. In our example it is straightforward to see that the value of the new integrand for $y = 0$ is 2, but often finding the limit will not be so easy and in some cases it may be necessary to calculate it numerically, evaluating the integrand for values close to the singularity (and paying attention to numerical round-off errors).

The treatment of integrals where one or both end points are at infinity is similar. One can partition off an interval from x_c to infinity, with x_c large enough that the integrand can be expanded in a form that allows one to evaluate the integral from x_c to ∞ analytically. The rest of the integral, now over a finite domain, is done with a numerical integration formula. For example, with $f(x) = x/[\exp(x) + 1]$, for large x one could expand

$$\frac{x}{e^x + 1} = xe^{-x} - xe^{-2x} + \ldots \tag{31}$$

and the integral of $f(x)$ from 0 to ∞ would be approximated by

$$\begin{aligned} \int_0^\infty f(x)\,dx &= \int_0^{x_c} f(x)\,dx + \int_{x_c}^\infty f(x)\,dx \\ &\approx \int_0^{x_c} f(x)\,dx + \int_{x_c}^\infty [xe^{-x} - xe^{-2x}]\,dx \\ &= \int_0^{x_c} f(x)\,dx + (x_c + 1)e^{-x_c} + (\frac{x_c}{2} + \frac{1}{4})e^{-2x_c} \end{aligned} \tag{32}$$

Otherwise, one can perform a change of variable of integration which maps the original infinite domain into a finite domain. In the example above, one possible change of variables could be $x = \tanh^{-1} y = [\log(1+y) - \log(1-y)]/2$, which maps the domain $0 \leq x \leq \infty$ into the domain $0 \leq y \leq 1$.

We conclude this section with the analysis of a simple but instructive example, which illustrates in detail how an integration formula like Simpson's fails to give an accurate result in presence of a singularity. We consider the integral between 0 and 1 of $\sqrt{x}$, which we will calculate with Simpson's formula and exactly, comparing the two results. We divide the interval into $2N$ subintervals of width $\delta = 1/2N$ and consider the approximation provided by Simpson's formula for the interval between $x = 2n\delta$ and $x = (2n+2)\delta$. This is

$$\begin{aligned} \Delta I_{approx} &= \frac{\delta}{3}\Big[\sqrt{2n\delta} + 4\sqrt{(2n+1)\delta} + \sqrt{(2n+2)\delta}\Big] \\ &= \frac{\delta^{\frac{3}{2}}}{3}\Big[\sqrt{2n} + 4\sqrt{2n+1} + \sqrt{2n+2}\Big] \end{aligned} \tag{33}$$

The exact expression for the same subinterval is

$$\Delta I_{exact} = \frac{2\delta^{\frac{3}{2}}}{3}\Big[(2n+2)^{\frac{3}{2}} - (2n)^{\frac{3}{2}}\Big] \tag{34}$$

In order to compare the two expressions, let us imagine that x takes a fixed value while $\delta \to 0$, $n \to \infty$ and let us expand the difference $\Delta I_{approx} - \Delta I_{exact}$ for large n. Many terms cancel and we are left with

$$\Delta I_{approx} - \Delta I_{exact} = \delta^{\frac{3}{2}} n^{-\frac{7}{2}} \frac{\sqrt{2}}{1536}\Big[-1 + \frac{7}{4n} - \frac{69}{32n^2} + O\Big(\frac{1}{n^3}\Big)\Big] \tag{35}$$

The cancellations are to be expected, since $1/n$ is $O(\delta)$ and thus the r.h.s. of Eq. 35 confirms that the error within the subinterval is of order δ^5. Let us now assume that we integrate $\sqrt{x}$ not from 0 to 1 but from some cutoff value x_c to 1. This means that we are adding up the contributions from $2n_c = x_c/\delta$ to $2N$ and the errors in Eq. 35 add up to a global error

$$I_{approx} - I_{exact} = -\frac{\sqrt{2}}{3840}\delta^{\frac{3}{2}} n_c^{-\frac{5}{2}}[1 + O(n_c^{-1})] = -\frac{1}{480}\delta^4 x_c^{-\frac{5}{2}}[1 + O(\delta/x_c)] \tag{36}$$

We see therefore that, so long as we integrate from a fixed value of x on up, Simpson's formula produces a result with error $O(\delta^4)$, as expected. But

if we let the lower limit x_c approach small values of the order of δ itself, the approximation deteriorates and the error becomes of order $\delta^{\frac{3}{2}}$. And the situation becomes even worse with an integrable singularity, like $1/\sqrt{x}$, in which case the error would be of order $\delta^{\frac{1}{2}}$. (One may worry about the applicability of the estimate of the error provided by Eq. 36 to values of x_c of order δ. While that estimate of the error, based on an expansion for large $n_c = x_c/\delta$, loses accuracy for small n_c, one can verify that the error becomes $O(\delta^{\frac{3}{2}})$ from a direct comparison of Eqs. 33 and 34 which gives, for instance, $\Delta I_{approx} = [(4\sqrt{3}+2)/3]\delta^{3/2}$ versus $\Delta I_{exact} = [16/3]\delta^{3/2}$ for the first interval between 0 and 2δ.)

These arguments can also be used to compare the accuracies of the two procedures described above, namely of changing the variable of integration versus using Simpson's formula from some x_c on and approximating the integrand by an expansion that can be handled analytically for $x < x_c$. We have to think, of course, that while the integrand behaves like $\sqrt{x}$ for $x \approx 0$, its form is given by a more complex expression, that we can expand as $\sqrt{x}(a_0 + a_1 x + \ldots)$ up to an error of order $\sqrt{x}\, x^m$. Integrating up to x_c the terms we neglected in the expansion of the integrand will produce an error of order $x_c^{(m+3/2)}$. Combining this with the result obtained in Eq. 36 we get a total error of order

$$c x_c^{m+\frac{3}{2}} + c' \delta^4 x_c^{-\frac{5}{2}} \tag{37}$$

where c and c' are some constants. This is minimized by $x_c \approx \delta^{\frac{4}{m+4}}$ with a resulting error of order $\delta^{\frac{4m+6}{m+4}}$. If we change variables of integration, instead, so as to eliminate the end point singularity, we expect an error of order δ^4, so we would conclude that changing variable of integration is more accurate than partitioning off the singularity and approximating the integral in its neighborhood. Of course, one must also be guided by common sense in the implementation of numerical techniques, and for many applications the results obtained by either method and a sufficiently small δ may be accurate enough. The important thing is to develop a sound understanding of how numerical methods work so as to be able to adapt them to exceptional situations with critical sense, rather than using the various formulae as black boxes, which most of the time will work, but on occasion, and unbeknownst to the unsuspecting user, can produce catastrophic results.

1.4 Finding zeroes of a function of a single variable.

A good first step for finding the zeroes of a function $f(x)$ consists in searching for an interval x_a–x_b where the function changes sign. The search could be done, for instance, by starting from some lower bound x_0 for the domain where one expects to find zeroes and increasing x in steps of Δ, with $x_i = x_0 + i\Delta$. As soon as the condition $f(x_i)f(x_{i+1}) \leq 0$ is fulfilled, one stops the search and takes $x_a = x_i$, $x_b = x_{i+1}$. Of course, it is important to make sure that the step is small enough that one does not skip over two consecutive zeroes. From this point of view, it is important to fold into the search as much information on the expected location(s) of the zeroe(s) as may be available from the context of the problem. Double or multiple zeroes can be particularly tricky. If one has reasons to suspect the presence of double zeroes, one should also search for zeroes of the derivative of the function.

Once an interval where $f(x)$ changes sign has been found, under the assumption that $f(x)$ is continuous, its zero (or at least one zero, if there are several) can be determined by the "partition and search" algorithm. The algorithm proceeds by dividing the interval x_a–x_b into two typically equal subintervals by a point x_m. Then, either $f(x_m) = 0$, or for one of the two subintervals x_a–x_m, x_m–x_b the value of the function f will change sign at the end points. This subinterval is taken as the new interval x_a–x_b and the search proceeds recursively. If the midpoint $x_m = (x_b - x_a)/2$ is used at each step of the search, in N steps the width of the interval is clearly reduced by a factor of 2^{-N}. Thus the algorithm can be used to delimit the interval containing the zero up to numerical precision. As a matter of fact, one does not need to assume continuity of $f(x)$: the algorithm will always determine an arbitrarily small interval (within numerical precision) where $f(x)$ changes sign.

Although this algorithm is very sturdy, its rate of convergence is not optimal. Much faster convergence can be achieved with the Newton-Raphson algorithm or with the "secant" algorithm. For the Newton-Raphson algorithm we must assume that the function $f(x)$ is continuous and differentiable and that we can calculate its derivative, either analytically or by numerical differentiation. We also assume that we have a trial value x_0 for a zero of $f(x)$ close enough to the actual zero x_{exact} that $f(x)$ is reasonably well approximated by a linear function in the interval bewteen x_0 and x_{exact}. The search proceeds as follows. We approximate $f(x)$ by the first two terms in

its Taylor series expansion around x_0

$$f(x) \approx f(x_0) + (x - x_0)f'(x_0) \tag{38}$$

and take as next iterate x_1 for the search the zero of the r.h.s. of Eq. 38, namely

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{39}$$

The procedure is repeated, finding new successive approximations $x_2, x_3 \ldots x_i$ until either $|x_{i+1} - x_i|$ or $f(x_i)$ is smaller than some preset tolerance.

The secant algorithm is similar, but is based on a linear interpolation of $f(x)$ rather than a Taylor series expansion. Thus knowledge or even existence of $f'(x)$ are not required. We approximate

$$f(x) \approx \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1) \tag{40}$$

where x_0 and x_1 are in the neighborhood of the zero of $f(x)$ and taxe as next iterate x_2 the zero of the r.h.s. of Eq. 40:

$$x_2 = x_1 + (x_0 - x_1)\frac{f(x_1)}{f(x_1) - f(x_0)} \tag{41}$$

x_2 replaces x_0 and the procedure is repeated, starting from the two points x_1 and x_2, etc.

(Note: it is important to always replace the next to last point x_{i-1} with the newly found approximation x_{i+1}. One should not modify the procedure allowing for the replacement of either x_i or x_{i-1}, according to some criterion, for instance replacing the point with the largest value of $|f(x)|$. This can considerably slow down the rate of convergence.)

In order to study the rate of convergence of the Newton-Raphson algorithm, let us consider

$$f(x) = \alpha x + \beta x^2 \tag{42}$$

in the neighborhood of of $x = 0$. The iteration formula Eq. 39 gives

$$x_{i+1} = x_i - \frac{\alpha x_i + \beta x_i^2}{\alpha + 2\beta x_i} = \frac{\beta x_i^2}{\alpha + 2\beta x_i} \tag{43}$$

Sufficiently close to the zero, it is possible to neglect the $2\beta x_i$ term in the denominator, and the iteration takes the very simple form

$$x_{i+1} = r x_i^2 \tag{44}$$

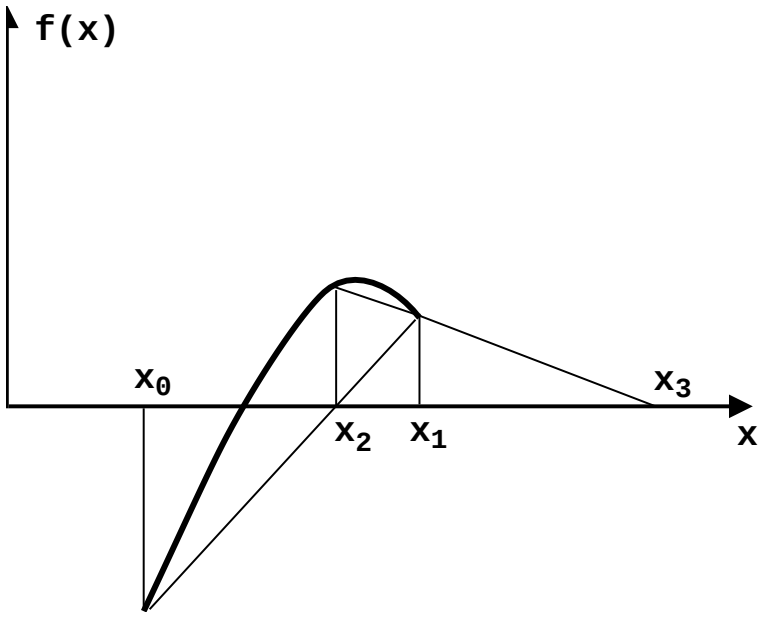

Figure 2: Instability of the secant algorithm.

with $r = \beta/\alpha$. Equation 44 in turn gives

$$x_n = r^{-1}[rx_0]^{2^n} \tag{45}$$

This shows that, provided that $rx_0 < 1$, i.e. that the starting point is close enough to the zero, the rate of convergence is extremely fast. Analogous arguments can be used with the secant algorithm, which has a similar rate of convergence. It thus appears that either the Newton-Raphson method or the secant method are preferable to the partition and search algorithm for finding zeroes of a function. The partition and search method is however more rubust, in the sense that, once an interval where $f(x)$ changes sign has been found, the iterations of the algorithm are guaranteed to converge to a zero within the interval, whereas this is not necessarily the case with the Newton-Raphson or secant method, as Fig. 2 illustrates. Frequently a good compromise consists in using a few iterations of the partition and search algorithm to delimit an interval small enough to avoid problems of instability, using then the Newton-Raphson or secant method to rapidly zoom in toward the zero.

1.5 Searching techniques.

In many applications one is given an interval x_0–x_N, which is further subdivided into N subintervals by vertices $x_1 \ldots x_{N-1}$ (with $x_0 < x_1 \ldots < x_{N-1} < x_N$). Given a variable x, with $x_0 \leq x \leq x_N$, one must find the index i of the interval x_i–x_{i+1} where x falls.

If the vertices have uniform spacing a, so that $x_i = x_0 + ia$, the problem is of immediate solution: i is given by

$$i = \mathrm{int}[(x - x_0)/a] \tag{46}$$

where $\mathrm{int}(y)$ denotes the integer part of the real variable y. The problem can also be solved in a similar manner if the points x_i can be mapped into a sequence of uniformly spaced points by a suitable function. For example, if the points x_i are in geometric progression, $x_{i+1}/x_i = \mathrm{const}$, the mapping $y = \log x$ will convert the original sequence into a sequence of uniformly spaced points y_i and the searching method can be applied to y. Such sequences, however, occur quite unfrequently. Generally one deals with either uniformly spaced points or with points that cannot be converted into a uniform sequence by any simple mapping. In the latter case, finding the interval where x falls requires repeated comparisons, that will be implemented in the code by if statements. One should be mindful, though, of the order of the comparisons. With a large number of vertices, it would be inefficient to start testing for $x < x_1$, to follow with a test for $x < x_2$ if $x \geq x_1$ etc. With a generic sequence and a generic value of x, this procedure would on the average take $N/2$ steps to find the interval containing x. Rather, one should start from the comparison of x with x_m, where $m = \mathrm{int}(N/2)$. If $x < x_m$, the original sequence x_0–x_N is replaced with x_0–x_m, otherwise it is replaced with x_m–x_N, and the procedure is repeated iteratively. The new sequence will contain at most $N/2 + 1$ intervals, and thus the search will conclude in $\sim \log_2 N$ iterations. An alternative procedure, which also entails $\sim \log_2 N$ iterations and is warranted if there is reason to expect that x will fall in a subinterval with low index, consists in testing for $x < x_1$ first, then to proceed with the test for $x < x_2$ if the former test is not satisfied, then to $x < x_4$, increasing every time the index by a factor of 2. Once an upper bound x_{2^k} has been found, the search procedure outlined above is applied to the interval $x_{2^{k-1}}$–x_{2^k}. This method may be particularly convenient if one deals not with just a single x, but with several numbers $x^{(1)} \leq x^{(2)} \leq x^{(3)} \ldots$, which must all be allocated to the appropriate intervals x_i–x_{i+1}.

1.6 Polynomial interpolation.

Polynomial interpolation consists in finding a polynomial $P_N(x)$ of degree N which takes at a specified set of $N+1$ different points $x_0, x_1 \ldots x_N$ the same values $f_0, f_1 \ldots f_N$ as a given function $f(x)$:

$$P_N(x_i) = f_i \equiv f(x_i) \qquad i = 0 \ldots N \tag{47}$$

Since Eqs. 47 constitute a set of $N+1$ linear equations with non-vanishing determinant for the $N+1$ coefficients of $P_N(x)$, there is always one unique solution. If the spacing of the points x_i is $O(\delta)$ and the function $f(x)$ is continuous with continuous derivatives of order up to $N+1$, then $|f(x) - P_N(x)| = O(\delta^{N+1})$ over the domain spanned by the points x_i.

The expression for $P_N(x)$ can be found in a straightforward manner as follows. $P_N(x)$ must be a linear combination of the values f_i and the coefficients themselves will be polynomials of degree N. We denote these coefficients by $c_i p_N^{(i)}(x)$, where $p_N^{(i)}(x)$ is normalized in such a way that the coefficient of the x^N term is 1 and c_i is a normalization coefficient. Thus

$$P_N(x) = \sum_i c_i p_N^{(i)}(x) f_i \tag{48}$$

On the other hand, $P_N(x)$ must reduce to f_i for $x = x_i$. This means that all $p_N^{(j)}(x)$ with $j \neq i$ must vanish at x_i or, equivalently, that $p_N^{(i)}(x)$ must vanish for all $x_{j \neq i}$. This gives

$$p_N^{(i)}(x) = (x - x_0)(x - x_1) \ldots (x - x_{i-1})(x - x_{i+1}) \ldots (x - x_N) \tag{49}$$

Substituting into Eq. 48 with $x = x_i$ and considering Eq. 47 we then find

$$c_i = [(x_i - x_0)(x_i - x_1) \ldots (x_i - x_{i-1})(x_i - x_{i+1}) \ldots (x_i - x_N)]^{-1} \tag{50}$$

For small values of N Eqs. 49 and 50 take a sufficiently simple form that one can write them directly into one's code. For example, with $N = 3$ and equally spaced points $x_0, x_1 = x_0 + a, x_2 = x_0 + 2a, x_3 = x_0 + 3a$ the cubic interpolation polynomial takes the form

$$\begin{aligned} P_3(x) = &-\frac{(x - x_1)(x - x_2)(x - x_3)}{6a^3} f_0 + \frac{(x - x_0)(x - x_2)(x - x_3)}{2a^3} f_1 \\ &-\frac{(x - x_0)(x - x_1)(x - x_3)}{2a^3} f_2 + \frac{(x - x_0)(x - x_1)(x - x_2)}{6a^3} f_3 \end{aligned} \tag{51}$$

For larger values of N it becomes more efficient to implement a recursion procedure. To illustrate this procedure, let us relabel as $P_N^{(0)}(x)$ the polynomial $P_N(x)$ and, more generally, let us denote by $P_M^{(i)}(x)$ the polynomial of degree M which interpolates $f(x)$ through the points $x_i, x_{i+1} \ldots x_{i+M}$. We observe that $P_N^{(0)}(x)$ can be expressed in terms of $P_{N-1}^{(0)}(x)$ and $P_{N-1}^{(1)}(x)$ as follows

$$P_N^{(0)}(x) = \frac{x - x_N}{x_0 - x_N} P_{N-1}^{(0)}(x) + \frac{x - x_0}{x_N - x_0} P_{N-1}^{(1)}(x) \tag{52}$$

Indeed, the r.h.s. of Eq. 52 is clearly a polynomial of degree N which takes value $P_{N-1}^{(0)}(x_0) = f_0$ for $x = x_0$ and $P_{N-1}^{(1)}(x_N) = f_N$ for $x = x_N$. Moreover, it also takes value f_i at the intermediate points x_i, $i = 1 \ldots N-1$, since for $x = x_i$ both $P_{N-1}^{(0)}(x)$ and $P_{N-1}^{(1)}(x)$ take the common value f_i and thus

$$P_N^{(0)}(x_i) = \Big[\frac{x_i - x_N}{x_0 - x_N} + \frac{x_i - x_0}{x_N - x_0}\Big] f_i = f_i \tag{53}$$

But Eq. 52 clearly gives origin to a recursive procedure, since $P_{N-1}^{(0)}(x)$ and $P_{N-1}^{(1)}(x)$ can similarly be expressed in terms of $P_{N-2}^{(0)}(x)$, $P_{N-2}^{(1)}(x)$ and $P_{N-2}^{(2)}(x)$ etc., until one arrives at the polynomials of degree 0: $P_0^{(0)}, P_0^{(1)} \ldots P_0^{(N)}$. These are obviously given by $f_0, f_1 \ldots f_N$ and thus, staring from the latter, one can proceed up the ladder until one finds $P_N^{(0)}(x)$. The procedure can be used to calculate the value of $P_N^{(0)}(x)$ for a given x and also to calculate its coefficients.

We would like to conclude with a note of caution. Although, as we have just seen, it is rather straightforward to find an interpolating polynomial of any degree, one should be wary of using interpolating polynomials of high degree. A polynomial of degree N will "accommodate" the $N+1$ values f_i of $f(x)$ whatever they are, and if these are affected by some error or imprecisely known, $P_N(x)$ will not smooth the error out, but will wiggle through the points, leading possibly to a very poor interpolation, especially insofar as the derivatives are concerned. As a further important observation, an interpolating polynomial should not be used to extrapolate a function outside the range of interpolation x_0–x_N. The polynomial, especially if of high degree, will typically grow very fast in absolute value as one moves away from the interval x_0–x_N and depart drastically from the function $f(x)$.