Lecture `quad1`: Introduction to numerical quadrature

November 14, 2022

Summary: Review of numerical integration from basic Calculus: left and right rectangle, midpoint, trapezoid, and Simpson rules. Introduction to general Newton–Cotes formulas.

References: Material here taken in part from C. F. Van Loan's *Introduction to Scientific Computing*. See also T. Sauer's *Numerical Analysis*, Section 5.2; Section 5.2

1 Need for numerical integration

Many (most) continuous functions do not have elementary antiderivatives. For example, $f(x) = \cos(x^2)$ does not. While an antiderivative $F(x)$ —defined as always up to a free constant— such that $F'(x) = \cos(x^2)$ does exist, we can't write down a simple easy–to–use expression, and it's not obvious how to evaluate $F(x)$ at a particular x value. For such a function, it's often not possible to use the *Fundamental Theorem of Calculus* to pencil–and–paper evaluate the definite integral. For example, it's not obvious how to calculate

$$\int_0^1 \cos(x^2)dx$$

as $F(1) - F(0)$. For such definite integrals we must use *numerical integration* to produce a number Q which is close, but almost certainly not exactly equal, to $F(1) - F(0)$.

The ideas associated with numerical integration are similar to those appearing in the construction of the Riemann integral. Before trying to integrate functions numerically, let's recall when it make sense to even try. Recall the following basic result from integral calculus.

Fact: Consider a function f defined on a closed interval $[a, b]$ which (i) is bounded on the interval and (ii) has only finitely many discontinuities. The function f is *integrable*. See Fig. 1 for a representation of such a function (with only one discontinuity).

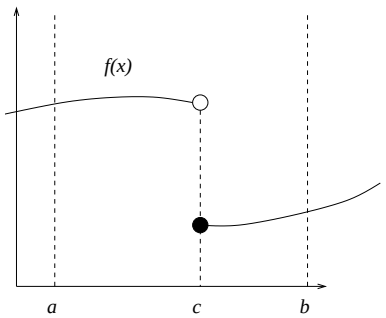

Figure 1: A discontinuous function.

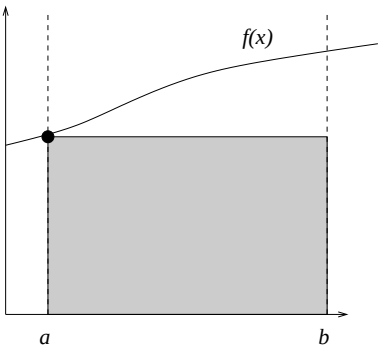

Figure 2: Left rectangle rule: $Q = (b-a)f(a)$.

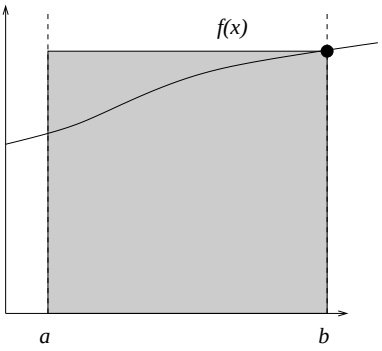

Figure 3: Right rectangle rule: $Q = (b-a)f(b)$.

This fact is a special case of *Lebesgue's Theorem*[1], and by integrable (really Riemann integrable) we mean that there is a notion of "area under the curve", that is to say the definite integral

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

is well–defined in the sense of a limit. The bounded property [recall that means $|f(x)| < M$ (some constant) for all $x \in (a,b)$] rules out some functions like $f(x) = 1/\sqrt{x}$ on $[0,1]$ which are nevertheless integrable, but for us it's enough to know the following: any function $f(x)$ which is continuous on $[a,b]$ is integrable. In fact, we shall mostly be interested in functions which are smoother still, and admit some number of derivatives on the interval.

2 Left and right rectangle and midpoint rules

Let's recall the simplest integration rules. The first two are the left and right rectangle rules shown in Figs. 2 and 3. As is evident from the figures, the rectangle rules tend to over or under count area under the curve. An improvement is the midpoint rule shown in Fig. 4. We'll see later why the midpoint rule is more *accurate*, which is to say generally yields a better approximation to I in (1).

All of these *integration rules* follow the same pattern. The function $f(x)$ is first approximated by a constant function $p_0(x) = K$, that is $f(x) \simeq p_0(x)$ on $[a,b]$. We use $p_0(x)$ as the notation here since

[1] J. E. Marsden and M. J. Hoffman, *Real Analysis*, second edition, 1993.

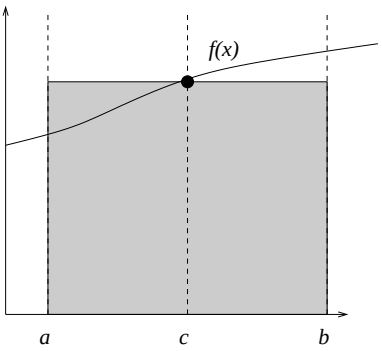

Figure 4: Midpoint rule: $Q = (b-a)f(c)$.

this constant function is a zeroth degree polynomial.[2] The considered rules arise by setting

- LRR: $p_0(x) = f(a)$.
- RRR: $p_0(x) = f(b)$.
- MR: $p_0(x) = f(c)$, where $c = \frac{1}{2}(a+b)$ is the midpoint.

We know how to integrate constants, so the midpoint rule, for example, is recovered as

$$\int_a^b f(x)dx = I \simeq Q = \int_a^b p(x)dx = (a-b)f(c), \qquad \text{where } c = \tfrac{1}{2}(a+b). \tag{1}$$

The viewpoint that an approximation Q to I arises upon integration of an approximating polynomial (here only a constant!) may seem like overkill now, but this proves to be a useful concept.

3 Trapezoid and Simpson rules

We can of course integrate more than constants, in fact it's elementary in principal to integrate any polynomial function. This suggests that we approximate $f(x)$ on $[a,b]$ by a higher order polynomial, and then use the associated integral Q to approximate I. The easiest such generalization is the trapezoid rule, shown in Fig. 5. To derive the rule, we use

$$p_1(x) = f(a) + \frac{f(b)-f(a)}{b-a}(x-a), \tag{2}$$

the unique linear (degree–1) polynomial which interpolates $f(x)$ at the endpoints of the interval. We then assume

$$\int_a^b f(x)dx = I \simeq Q = \int_a^b p_1(x)dx = \tfrac{1}{2}(b-a)(f(a)+f(b)). \tag{3}$$

To derive Simpson's rule (also known as the parabolic rule), we start with a quadratic polynomial (a parabola),

$$\begin{aligned} p_2(x) &= f(a) + f[a,c](x-a) + f[a,c,b](x-a)(x-c) \\ &= f(a) + \frac{f(c)-f(a)}{c-a}(x-a) + \frac{\dfrac{f(b)-f(c)}{b-c} - \dfrac{f(c)-f(a)}{c-a}}{b-a}(x-a)(x-c), \end{aligned} \tag{4}$$

[2] **Caution:** In earlier lectures we used $p_n(x)$ for a degree $n-1$ polynomial which interpolates at n points; but now switch to Sauer's notation where $p_{n-1}(x)$ is the degree $n-1$ polynomial interpolating at n points.

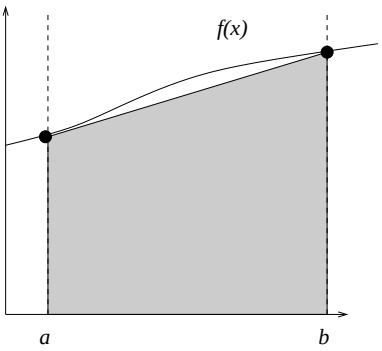

Figure 5: Trapezoid rule: $Q = \frac{1}{2}(b-a)(f(a)+f(b))$.

which interpolates $f(x)$ at $a, b, c = \frac{1}{2}(a+b)$. Note the Newton divided–differences in the first line of the last formula. However, to compute $\int_a^b p_2(x)dx$, it's perhaps better to use the Lagrange form

$$p_2(x) = f(a)\frac{(x-c)(x-b)}{(a-c)(a-b)} + f(c)\frac{(x-a)(x-b)}{(c-a)(c-b)} + f(b)\frac{(x-a)(x-c)}{(b-a)(b-c)} \tag{5}$$

of the interpolating polynomial. We then need to compute three integrals; the first is

$$\begin{aligned}
\int_a^b \frac{(x-c)(x-b)}{(a-c)(a-b)}dx &= (b-a)\int_0^1 \frac{\big((b-a)t-(c-a)\big)\big((b-a)t-(b-a)\big)}{(a-c)(a-b)}dt, \qquad \text{where } x=(b-a)t+a \\
&= (b-a)\int_0^1 \frac{\big(t-\frac{(c-a)}{(b-a)}\big)\big(t-1\big)}{\frac{c-a}{b-a}}dt \\
&= (b-a)\int_0^1 (2t-1)(t-1)dt, \qquad \text{since } \frac{c-a}{b-a} = \frac{\frac{1}{2}(b+a)-a}{b-a} = \frac{1}{2} \\
&= (b-a)\int_0^1 (2t^2-3t+1)dt \\
&= (b-a)\big(\tfrac{2}{3}t^3 - \tfrac{3}{2}t^2 + t\big)\big|_0^1 \\
&= (b-a)\big(\tfrac{2}{3} - \tfrac{3}{2} + 1\big) \\
&= \tfrac{1}{6}(b-a).
\end{aligned} \tag{6}$$

Likewise, also using $c = \frac{1}{2}(a+b)$, we find

$$\int_a^b \frac{(x-a)(x-b)}{(c-a)(c-b)}dx = \tfrac{2}{3}(b-a), \qquad \int_a^b \frac{(x-a)(x-c)}{(b-a)(b-c)}dx = \tfrac{1}{6}(b-a). \tag{7}$$

It's then straightforward to compute

$$Q = \int_a^b p_2(x)dx = \tfrac{1}{6}(b-a)(f(a)+4f(c)+f(b)), \tag{8}$$

again an approximation to I.

4 Newton–Cotes rules

An m–point *quadrature rule* Q for approximating the definite integral

$$I = \int_a^b f(x)dx \tag{9}$$

is defined by the formula

$$Q = (b-a)\sum_{k=1}^{m} w_k f(x_k), \tag{10}$$

where the w_k are called *weights* and the x_k *nodes* (Van Loan uses the term *abscissas* in place of *nodes*), and the term *points* is sometimes also used for the x^k. For example, the Simpson rule is clearly a 3–point quadrature rule with $x_1 = a$, $x_2 = c$, $x_3 = b$, $w_1 = \frac{1}{6}$, $w_2 = \frac{4}{6}$, and $w_3 = \frac{1}{6}$. Notice that in the Simpson rule the function value $f(x_2) = f(c)$ is paired with w_2, hence is weighted more in the sum than either $f(x_1)$ or $f(x_3)$, whence the term *weights* for the w_k.

The midpoint, trapezoid, and Simpson rules are (simple) examples of Newton–Cotes quadrature rules (the left and right rectangle rules are not). The Newton-Cotes rules come in two varieties *closed* (function evaluations at the endpoints a and b are included, such as for trapezoid and Simpson) *open* (no function evaluations at the endpoints, such as for midpoint). Let's start with the closed variety. We will define

$$Q_{NC(m)} = \int_a^b p_{m-1}(x)dx, \tag{11}$$

where $p_{m-1}(x)$ is a degree–$m-1$ polynomial which interpolates $f(x)$ at the points

$$x_i = a + \frac{i-1}{m-1}(b-a), \qquad i = 1, 2, \cdots, m. \tag{12}$$

This is a collection of uniformly spaced nodes. Is it clear that, so defined, $Q_{NC(m)}$ will have the form (10)? Yes! The polynomial $p_{m-1}(x)$ can be built up via the Newton or Lagrange methods, and it will depend linearly on $f(x_k)$, as well as the x_k and the independent variable x. Therefore, upon integration we get the formula. Indeed, consider the calculation based on the Lagrange form of $p_{m-1}(x)$:

$$\int_a^b p_{m-1}(x)dx = \sum_{k=1}^{m} f(x_k)\int_a^b \ell_k(x)dx = (b-a)\sum_{k=1}^{m} f(x_k)\underbrace{\int_0^1 \ell_k\big((b-a)t+a\big)dt}_{w_k}, \tag{13}$$

where

$$\ell_k(x) = \prod_{\substack{j=1\\ j\neq k}}^{m} \frac{x-x_j}{x_k-x_j} \implies \ell_k\big((b-a)t+a\big) = \prod_{\substack{j=1\\ j\neq k}}^{m} \frac{t-t_j}{t_k-t_j}, \quad t_j = \frac{j-1}{m-1}. \tag{14}$$

See Van Loan page 138 for more detail. The first four such closed rules are

- $\frac{1}{2}(x_2-x_1)(f(x_1)+f(x_2))$,
  MATLAB weights: `w = [1 1]/2`.
- $\frac{1}{6}(x_3-x_1)(f(x_1)+4f(x_2)+f(x_3))$,
  MATLAB weights: `w = [1 4 1]/6`.
- $\frac{1}{8}(x_4-x_1)(f(x_1)+3f(x_2)+3f(x_3)+f(x_4))$,
  MATLAB weights: `w = [1 3 3 1]/8`.
- $\frac{1}{90}(x_5-x_1)(7f(x_1)+32f(x_2)+12f(x_3)+32f(x_4)+7f(x_5))$,
  MATLAB weights: `w = [7 32 12 32 7]/90`,

and (like the open rules) can be expressed in a vector notation

$$Q_{NC(m)} = (b-a)\sum_{i=1}^{m} w_i f(x_i) = (b-a)\begin{pmatrix} w_1 & w_2 & \cdots & w_m \end{pmatrix}\begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_m) \end{pmatrix}. \tag{15}$$

The Matlab function (see Van Loan, p. 140)

```
  function Q = ClosedQNC(f,a,b,m)
% function Q = ClosedQNC(f,a,b,m)
% Integrates a function of the form f(x) passed as a handle from a to b.
% f must be defined on [a,b] and it must return a column vector if x is a
% column vector. m is an integer that satisfies 2 <= m <= 11. Q is the
% m-point closed Newton-Cotes approximation of the integral int_a^b f(x)dx.
  w  = NewtonCotesClosedWeights(m);
  x  = linspace(a,b,m)';
  fx = f(x);
  Q = (b-a)*(w*fx);  % w is a row vector and fx a column vector.
```

will do this, where

```
   function w = NewtonCotesClosedWeights(m)
%  function w = NewtonCotesClosedWeights(m)
%  w is a column m-vector consisting of the weights for the m-point closed
%  Newton-Cotes rule. m is an integer that satisfies 2 <= m <= 11. From
%  M. Abramowitz and I. Stegun.
   switch m,
     case 2,
       w = [1 1]/2;
     case 3,
       w = [1 4 1]/6;
     case 4,
       w = [1 3 3 1]/8;
     case 5,
       w = [7 32 12 32 7]/90;
     case 6,
       w = [19 75 50 50 75 19]/288;
     case 7,
       w = [41 216 27 272 27 216 41]/840;
     case 8,
       w = [751 3577 1323 2989 2989 1323 3577 751]/17280;
     case 9,
       w = [989 5888 -928 10496 -4540 10496 -928 5888 989]/28350;
     case 10,
      w = [2857 15741 1080 19344 5778 5778 19344 1080 15741 2857]/89600;
     case 11,
       w = [16067 106300 -48525 272400 -260550 427368 ...
          -260550 272400 -48525 106300 16067]/598752;
      otherwise,
       error = 'm not between 2 and 11 in NewtonCotesClosedWeights'
       pause
   end
```

5 Remarks on open rules

Van Loan gives little attention to the open rules, which is why he uses just $Q_{NC(m)}$ for the closed case, and not, say, the more awkward notations $Q^{\text{closed}}_{NC(m)}$ and $Q^{\text{open}}_{NC(m)}$. We'll also mostly focus on the closed case, and $Q_{NC(m)}$ will always be understood as the m–point closed rule. However, the open rules are similarly defined. For example,

```
  function Q = OpenQNC(f,a,b,m)
% function Q = OpenQNC(f,a,b,m)
% Integrates a function of the form f(x) passed as a handle from a to b.
% f must be defined on [a,b] and it must return a column vector if x is a
% column vector. m is an integer that satisfies 2 <= m <= 7. Q is the
% m-point closed Newton-Cotes approximation of the integral int_a^b f(x)dx.
  w  = NewtonCotesOpenWeights(m);
  h  = (b-a)/(m+1);
  x  = [a+h:h:b-h]';
  fx = f(x);
  Q = (b-a)*(w*fx);  % w is a row vector and fx a column vector.
```

where the m–point open nodes are defined as $x_i^{\text{open}} = a + i(b-a)/(m+1), i = 1, \cdots, m$. The open weights are given by

```
   function w = NewtonCotesOpenWeights(m)
%  function w = NewtonCotesOpenWeights(m)
%  w is a column m-vector consisting of the weights for the m-point closed
%  Newton-Cotes rule. m is an integer that satisfies 2 <= m <= 7. From
%  M. Abramowitz and I. Stegun.
   switch m,
     case 1,
       w = [1];
     case 2,
       w = [1 1]/2;
     case 3,
       w = [2 -1 2]/3;
     case 4,
       w = [11 1 1 11]/24;
     case 5,
       w = [11 -14 26 -14 11]/20;
     case 6,
       w = [611 -453 562 562 -453 611]/1440;
     case 7,
       w = [460 -954 2196 -2459 2196 -954 460]/945;
     otherwise,
       error = 'm not between 2 and 7 in NewtonCotesOpenWeights'
       pause
   end
```

Notice that the open weights "go negative" quicker (already for $m = 3$). When integrating strictly positive functions, it's desirable to have purely positive weights, since then subtraction errors are not at issue. This issue indicates that the closed rules are preferable.