Lecture quad4: Numerical Quadrature: Gaussian Quadrature

November 21, 2022

Summary: Basic idea behind Gauß–Legendre (henceforth Gauss–Legendre) quadrature with examples.

References: See also T. Sauer's *Numerical Analysis*, Section 5.5, 1st ed. pages 274–279, 2nd ed. pages 273–278.

So far we have mostly dealt with closed Newton–Cotes quadrature rules,

$$(b-a)\sum_{k=1}^{m} w_k f(x_k) = Q_{NC(m)} \simeq I = \int_a^b f(x)dx. \tag{1}$$

In that case the *nodes* or *points*

$$x_k = a + (k-1)\frac{b-a}{m-1}, \qquad k = 1, \cdots, m \tag{2}$$

are taken as uniformly spaced, and once an m is fixed we have no freedom in choosing them. By construction, the closed Newton–Cotes rule is

$$Q_{NC(m)} = \int_a^b p_{m-1}(x)dx, \tag{3}$$

where $p_{m-1}(x)$ is the unique Newton interpolating polynomial which fits the data set $\mathcal{B} = \{(x_k, f(x_k)) : k = 1, \cdots, m\}$. For $m = 3$ (Simpson's rule), the interpolating polynomial $p_2(x)$ is quadratic and fits the data

$$\big\{\big(a, f(a)\big), \big(c, f(c)\big), \big(b, f(b)\big)\big\}, \qquad c = \tfrac{1}{2}(a+b). \tag{4}$$

Since Simpson's rule is based on a quadratic interpolating polynomial, it will *exactly* integrate the monomials $1, x, x^2$. Therefore, we may derive the weights for Simpson's rule (already seen to be `w = [1, 4, 1]/6`) by considering the following integrals

$$\begin{aligned}\int_0^1 1dx &= 1\\ \int_0^1 xdx &= \tfrac{1}{2}\\ \int_0^1 x^2dx &= \tfrac{1}{3},\end{aligned} \tag{5}$$

where we point out that $1 = x^0$ and $x = x^1$. We need to solve for 3 weights w_k, and we have 3 integrals. The idea is to replace each integral with the quadrature rule, which is *exact* for these low–order monomials. Whence, upon replacing the integrals on the left-hand side with the quadrature rule, we have the system

$$\begin{array}{ccccccc} w_1\cdot 0^0 & + & w_2\cdot(\frac{1}{2})^0 & + & w_3\cdot 1^0 & = & 1\\ w_1\cdot 0^1 & + & w_2\cdot(\frac{1}{2})^1 & + & w_3\cdot 1^1 & = & \frac{1}{2}\\ w_1\cdot 0^2 & + & w_2\cdot(\frac{1}{2})^2 & + & w_3\cdot 1^2 & = & \frac{1}{3},\end{array} \tag{6}$$

or more simply

$$\begin{aligned} w_1 + w_2 + w_3 &= 1 \\ \tfrac{1}{2}w_2 + w_3 &= \tfrac{1}{2} \\ \tfrac{1}{4}w_2 + w_3 &= \tfrac{1}{3}. \end{aligned} \tag{7}$$

We may readily check that $w_1 = \frac{1}{6}$, $w_2 = \frac{4}{6}$, and $w_3 = \frac{1}{6}$ solves the system, and also see that this solution stems from the matrix problem

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & \frac{1}{2} & 1 \\ 0 & \frac{1}{4} & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \end{pmatrix}. \tag{8}$$

As has been stressed in **Homework #12**, *the weights for a Newton–Cotes rule can be viewed as the solution to a linear system*, in the same fashion as considered above for the special case of Simpson's rule.

1 Gauss–Legendre rules

Before defining the Gauss–Legendre rules, we make some observations having nothing to do with the rules *per se.* So far, we have followed Van Loan and the standard convention that the quadrature rule is associated with $[0, 1]$, and so include a factor of $(b - a)$ in the rule when applied to $[a, b]$. We could have chosen to instead base the rules on $[-1, 1]$, in which case we would instead have defined

$$\tfrac{1}{2}(b-a)\sum_{k=1}^{m} w_k f(x_k). \tag{9}$$

The difference is reflected in whether we choose to express the basic integral $\int_a^b f(x)dx$ as

$$\int_a^b f(x) = (b-a)\int_0^1 f\big((b-a)s + a\big)ds \qquad \text{or} \qquad \tfrac{1}{2}(b-a)\int_{-1}^1 f\big(\tfrac{1}{2}(b-a)t + \tfrac{1}{2}(b+a)\big)dt. \tag{10}$$

Both stem from simple transformations of variable, but the second choice goes with (9), and we are going to work with it now.

We could repeat the analysis above for the $m = 3$ Newton–Cotes formula. Doing so would yield the system

$$\begin{array}{ccccccc} w_1 \cdot (-1)^0 & + & w_2 \cdot (\frac{1}{2})^0 & + & w_3 \cdot 1^0 & = & 2 \\ w_1 \cdot (-1)^1 & + & w_2 \cdot (\frac{1}{2})^1 & + & w_3 \cdot 1^1 & = & 0 \\ w_1 \cdot (-1)^2 & + & w_2 \cdot (\frac{1}{2})^2 & + & w_3 \cdot 1^2 & = & \frac{2}{3}, \end{array} \tag{11}$$

or

$$\begin{aligned} w_1 + w_2 + w_3 &= 2 \\ -w_1 + w_3 &= 0 \\ w_1 + w_3 &= \tfrac{2}{3}. \end{aligned} \tag{12}$$

Now the weights are $w_1 = \frac{1}{3}$, $w_2 = \frac{4}{3}$, and $w_3 = \frac{1}{3}$; a simple factor of 2 times what they were before. At the level of the quadrature rule, all that has been done is the following:

$$(b-a)\sum_{k=1}^{m} w_k f(x_k) \quad \to \quad \tfrac{1}{2}(b-a)\sum_{k=1}^{m}(2w_k)f(x_k). \tag{13}$$

We have raised these issues for the following reason. In introducing Gauss–Legendre quadrature we are, like Van Loan, going to use the interval $[-1,1]$. But this switch from $[0,1]$ is just a choice. We could have instead stuck with $[0,1]$, just has we could have worked with $[-1,1]$ when introducing the Newton–Cotes rules.

Having made these observations, let us turn to the definition of the m–point Gauss–Legendre quadrature rule $Q_{GL(m)}$. The key idea is to introduce freedom in how the points are chosen as well. We start with the 2–point rule $Q_{GL(2)}$ for which there are *four* unknowns, two points $\{x_1, x_2\}$ and two weights $\{w_1, w_2\}$. With four unknowns, we shall need four equations, and so consider the integrals

$$
\begin{aligned}
\int_{-1}^{1} 1dx &= 2 \\
\int_{-1}^{1} xdx &= 0 \\
\int_{-1}^{1} x^2dx &= \tfrac{2}{3} \\
\int_{-1}^{1} x^3dx &= 0.
\end{aligned} \tag{14}
$$

We will demand that the quadrature rule $Q_{GL(m)}$ is exact, when used to integrate the monomials $1, x, x^2, x^3$. Therefore, we may replace the left-hand sides in the last equation with the quadrature rule, thereby arriving at the system

$$
\begin{array}{ccccc}
w_1 \cdot (x_1)^0 & + & w_2 \cdot (x_2)^0 & = & 2 \\
w_1 \cdot (x_1)^1 & + & w_2 \cdot (x_2)^1 & = & 0 \\
w_1 \cdot (x_1)^2 & + & w_2 \cdot (x_2)^2 & = & \frac{2}{3} \\
w_1 \cdot (x_1)^3 & + & w_2 \cdot (x_2)^3 & = & 0,
\end{array} \tag{15}
$$

or more simply

$$
\begin{aligned}
w_1 + w_2 =& 2 \\
w_1x_1 + w_2x_2 =& 0 \\
w_1x_1^2 + w_2x_2^2 =& \tfrac{2}{3} \\
w_1x_1^3 + w_2x_2^3 =& 0.
\end{aligned} \tag{16}
$$

Now these are *nonlinear equations* for the variables $\{x_1, x_2, w_1, w_2\}$, since, for example, x_1^3 appears in the last equation. Unlike the case with linear equations, solving nonlinear equations is not a straightforward process (even in principle). Nevertheless, for the simple case here, we can in fact solve the equations. Indeed, multiplying the second equation by x_1^2 and then subtracting the result from the last equation, we find $x_2^2 w_2(x_2^2 - x_1^2) = 0$, so that $x_1 = \pm x_2$ (since the weights must be nonzero). Now $x_1 = 0 = x_2$ is not possible, or else the third equation would be violated. Moreover, if $x_1 = x_2$, then the second equation is $(w_1 + w_2)x_1 = 0$, which, since $x_1 \neq 0$, violates the first equation. We therefore conclude that $x_1 = -x_2$. Now, we're getting somewhere, since from the second equation we now know $x_1(w_1 - w_2) = 0$, so $w_1 = w_2$. Then, from the first equation, $w_1 = 1 = w_2$. Note that $w_1 = w_2$ and $x_1 = -x_2$ automatically satisfies the last equation, so we turn to the third equation to find

$$
\tfrac{2}{3} = w_1x_1^2 + w_2x_2^2 = 2x_1^2, \tag{17}
$$

and so we pick $x_1 = -\sqrt{\frac{1}{3}} \simeq -0.57735$, $x_2 = \sqrt{\frac{1}{3}} \simeq 0.57735$.

Finding the nodes and weights for higher order rules becomes more difficult. Consider the 3–point rule $Q_{GL(3)}$ for which we need to find $\{x_1, x_2, x_3, w_1, w_2, w_3\}$. The unknowns satisfy the nonlinear system

$$\begin{array}{ccccccc} w_1\cdot(x_1)^0 & + & w_2\cdot(x_2)^0 & + & w_3\cdot(x_3)^0 & = & 2 \\ w_1\cdot(x_1)^1 & + & w_2\cdot(x_2)^1 & + & w_3\cdot(x_3)^1 & = & 0 \\ w_1\cdot(x_1)^2 & + & w_2\cdot(x_2)^2 & + & w_3\cdot(x_3)^2 & = & \frac{2}{3} \\ w_1\cdot(x_1)^3 & + & w_2\cdot(x_2)^3 & + & w_3\cdot(x_3)^3 & = & 0 \\ w_1\cdot(x_1)^4 & + & w_2\cdot(x_2)^4 & + & w_3\cdot(x_3)^4 & = & \frac{2}{5} \\ w_1\cdot(x_1)^5 & + & w_2\cdot(x_2)^5 & + & w_3\cdot(x_3)^5 & = & 0, \end{array} \tag{18}$$

corresponding to the set of integrals

$$\begin{aligned} \int_{-1}^{1} 1dx &= 2 \\ \int_{-1}^{1} xdx &= 0 \\ \int_{-1}^{1} x^2dx &= \tfrac{2}{3} \\ \int_{-1}^{1} x^3dx &= 0 \\ \int_{-1}^{1} x^4dx &= \tfrac{2}{5} \\ \int_{-1}^{1} x^5dx &= 0. \end{aligned} \tag{19}$$

Again, the idea is that $Q_{GL(3)}$ is exact on a set of low–order monomials, in this case 1, x, x^2, x^3, x^4, x^5. We shall not solve the above equations, but list decimal solutions in the following MATLAB function.

```
  function [w,x] = GaussLegendreWeightsNodes(m)
% [w,x] = GaussLegendreWeightsNodes(m)
% Input m is number of points, and here satisfies 2 <= m <= 6.
% Output is the following.
% w is a column m-vector of weights for m-point GL rule.
% x is a column m-vector of nodes for m-point GL rule on [-1,1].
% This is essentially Van Loan's GLWeights, page 156 of textbook.

w = ones(m,1);
x = ones(m,1);
switch m,
  case 2,
    w(1) =  1.000000000000000; w(2) =  w(1);
    x(1) = -0.577350269189626; x(2) = -x(1);
  case 3,
    w(1) =  0.555555555555555; w(3) =  w(1);
    w(2) =  0.888888888888888;
    x(1) = -0.774596669241484; x(3) = -x(1);
    x(2) =  0.000000000000000;
  case 4,
    w(1) =  0.347854845137455; w(4) =  w(1);
    w(2) =  0.652145154862546; w(3) =  w(2);
    x(1) = -0.861136311594052; x(4) = -x(1);
    x(2) = -0.339981043584856; x(3) = -x(2);
  case 5,
    w(1) =  0.236926885056189; w(5) =  w(1);
    w(2) =  0.478628670499368; w(4) =  w(2);
    w(3) =  0.568888888888887;
    x(1) = -0.906179845938664; x(5) = -x(1);
    x(2) = -0.538469310105682; x(4) = -x(2);
    x(3) =  0.000000000000000;
  case 6,
    w(1) =  0.171324492379173; w(6) =  w(1);
    w(2) =  0.360761573048136; w(5) =  w(2);
    w(3) =  0.467913934572692; w(4) =  w(3);
    x(1) = -0.932469514203152; x(6) = -x(1);
    x(2) = -0.661209386466264; x(5) = -x(2);
    x(3) = -0.238619186083198; x(4) = -x(3);
  otherwise,
    error = 'Only 2 <= m <= 6 possible in GaussLegendreWeightsNodes'
    pause
end
```

Using the function above, we have another function which performs the Gauss–Legendre rule through order 6. Namely,

```
  function numI = QGaussLegendre(fname,a,b,m)
% numI = QGaussLegendre(fname,a,b,m)
% Integrates a function f(x) named by a string fname 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 <= 6.
% numI is the m-point Gauss-Legendre approximation of int_a^b f(x) dx.
% This is essentially Van Loan's QGL, page 157 of textbook.

[w,x] = GaussLegendreWeightsNodes(m);
fvals = feval(fname,0.5*(b-a)*x+0.5*(a+b)*ones(m,1));
numI  = 0.5*(b-a)*w'*fvals;
```

Compare the argument of `feval` in `QGaussLegendre` with the second integral in (10). To test these routines, we consider

$$\int_{\pi}^{3\pi/2} \sin x dx = -1.$$

In MATLAB we then perform the following commands

```
>> error = QGaussLegendre('sin',pi,3*pi/2,2)+1
error =
    0.0015
>> error = QGaussLegendre('sin',pi,3*pi/2,3)+1
error =
  -8.1216e-06
>> error = QGaussLegendre('sin',pi,3*pi/2,4)+1
error =
   2.2803e-08
>> error = QGaussLegendre('sin',pi,3*pi/2,5)+1
error =
  -3.9565e-11
>> error = QGaussLegendre('sin',pi,3*pi/2,6)+1
error =
   4.6185e-14
```

By way of comparison

```
>> error = ClosedQNC('sin',pi,3*pi/2,6)+1
error =
   4.7386e-06
```

So we see that for 6 points the Gauss–Legendre rule is accurate to about 8 more digits than the closed Newton–Cotes rule. This might not be a fair comparison, since the Newton–Cotes rules are better for odd powers. But

```
>> error = ClosedQNC('sin',pi,3*pi/2,7)+1
error =
  -2.5837e-08
```

is also much worse.