Notes on Gaussian quadrature

Earlier we developed the NC quadrature rules by choosing \( n + 1 \) evenly-spaced nodes in \([a, b]\) and computing corresponding weights as definite integrals of the resulting Lagrange basis polynomials. Recall that the DOP of the resulting rule is at most \( n + 1 \).

Is this choice of points optimal? Can we do any better? (“Better” here means guaranteeing a higher degree of precision.) This is our motivating question, and the goal of this note is to show there are better choices.

Remark. We have encountered exactly this question before, in the context of polynomial interpolation. We observed Runge’s phenomenon when choosing an evenly-spaced grid for interpolation, and we concluded that the Chebyshev points, the roots of the Chebyshev polynomials, are in some sense optimal for interpolation.

In this note we’ll show that there is an optimal choice of points, and that (surprisingly, or perhaps not) they are also the roots of a named family of polynomials (Legendre).

The key difference then between the NC and the Gaussian quadrature rules is that: whereas to define an NC rule we first choose evenly spaced nodes and then determine the DOP as a consequence, to define a Gaussian rule we first define the DOP to be \( 2n - 1 \) and then determine the nodes which satisfy the (high) DOP requirement.

In both cases the weights are given by definite integrals of appropriate Lagrange basis polynomials. We’ll show later that, once the DOP of any (NC or Gaussian) quadrature rule is known, the weights can be obtained by solving a linear system.

For the \( n \)-point Gaussian quadrature rule we begin by requiring

\[
\int_a^b p(x)\,dx = \sum_{i=1}^n w_i p(x_i),
\]

for all polynomials \( p(x) \) with degree at most \( 2n - 1 \).
Remark. Our notation for NC rules differs from that of Gaussian rules: in NC rules
the first index is 0 while here it is 1.

Expanding the above equation in terms of an arbitrary polynomial

\[ p(x) = \sum_{j=0}^{2n-1} a_j x^j \]

of degree at most \(2n - 1\) will suggest how to compute the points and weights:

\[
\int_a^b p(x) dx = \int_a^b \left( \sum_{j=0}^{2n-1} a_j x^j \right) dx = \sum_{j=0}^{2n-1} a_j \left( \int_a^b x^j dx \right).
\]

It should be clear that our quadrature will integrate \(p(x)\) exactly only if it integrates
each of the functions \(x^j\) exactly, for \(j = 0, 1, \ldots, 2n - 1\).

Thus we require that our abscissas and weights satisfy the following system:

\[
\sum_{i=1}^{n} w_i x_i^k = \int_a^b x^k dx, \quad (1)
\]

for \(k = 0, 1, \ldots, 2n - 1\).

Remark. The size of this system roughly suggests why we set the DOP to be \(2n - 1\): it
results in \(2n\) equations, and with \(2n\) variables or degrees of freedom \((w_1, \ldots, w_n, x_1, \ldots, x_n)\),
this is the best we can reasonably seek.

To simplify our computation, note that by using the change of variable \(x = \frac{b-a}{2} t + \frac{a+b}{2}\),
we can transform our original integral into an integral over the interval
\([-1, 1]\):

\[
\int_a^b f(x) dx = \int_{-1}^{1} f \left( \frac{b-a}{2} t + \frac{a+b}{2} \right) dt, \quad (2)
\]

so it suffices to consider the above system on the symmetric interval \([-1, 1]\).

Next, note we can easily obtain the RHS, since

\[
\int_{-1}^{1} x^k dx = \begin{cases} 
\frac{2}{k+1}, & k \text{ is even}, \\
0, & k \text{ is odd}.
\end{cases}
\]
Finally, note that the system is nonlinear in the $x_i$’s except in the trivial case $n = 1$ but it is always linear in the $w_i$’s. Therefore, knowledge of the $x_i$’s allows for an easy computation of the $w_i$’s, by solving a linear system (easy in the context of the known linear algebra algorithms).

**Remark.** We can also use this strategy to compute the weights of an NC quadrature rule, since

Due to the rich structure of inner product spaces, it turns out that the correct nodes correspond to the roots of $\phi_n$, the $n$th Legendre polynomial. (These are **LEGENDRE** polynomials, not to be confused with the **LAGRANGE** polynomials.) You may find a proof of this result on pp.489 – 491 of our textbook.

The Legendre polynomials satisfy the recurrence relation

$$\phi_{n+1}(x) = \frac{2n+1}{n+1} x \phi_n(x) - \frac{n}{n+1} \phi_{n-1}(x),$$

with $\phi_0(x) = 1$, $\phi_1(x) = x$.

For reference, we provide the first few Legendre polynomials:

- $\phi_2(x) = (3x^2 - 1)/2$
- $\phi_3(x) = (5x^3 - 3x)/2$
- $\phi_4(x) = (35x^4 - 30x^2 + 3)/8$

We note that, the Legendre polynomial $\phi_n$ is even when $n$ is even, and it is odd when $n$ is odd (just like the Chebyshev polynomials!). It also turns out that the $n$ roots of $\phi_n$ lie in $[-1, 1]$. Therefore, even if we don’t know $\phi_n$, we can always assume the placement of the nodes is symmetric about 0. In particular, for the three-point rule we must have $x_2 = 0$ and $x_3 = -x_1$. In the four-point rule we must have $x_4 = -x_2$ and $x_3 = -x_1$. The symmetry in the $x_j$’s implies symmetry in the $w_j$’s. For instance, in the three-point rule we must have $w_1 = w_3$ (however we can’t assume $w_2 = 0$!). We can (should!) exploit this symmetry to simplify the solution of the nonlinear system.

To sum up, to specify the $n$-point Gaussian quadrature rule on the interval $[a, b]$ we first perform the change of variables described by Equation 2. Second, we find the abscissas by computing the roots of the $n$th Legendre polynomial (or by solving the associated system of nonlinear equations). Finally, we substitute the computed $x_i$’s into the system described by Equation 1 and solve for the $w_i$’s.

For reference, we provide the abscissas and weights for the 2-point rule. In that case, $x_1 = -\frac{1}{\sqrt{3}}$, $x_2 = -x_1 = \frac{1}{\sqrt{3}}$ and $w_1 = w_2 = 1$. You will compute the abscissas and weights for other rules in HW6.
When \( n = 2 \), symmetry guarantees \( x_2 = -x_1 \) and \( w_2 = w_1 \). There are two ways to find the nodes: we may compute the roots of \( \phi_2 \) (in general this can be done with the aid of a computer), or we may use the system of equations (1). We do both here.

Since \( \phi_2(x) = (3x^2 - 1)/2 \), each node satisfies \((\sqrt{3}x + 1)(\sqrt{3}x - 1) = 0\). This implies \( x_1 = -1/\sqrt{3} \), \( x_2 = 1/\sqrt{3} \) (by convention we assume \( x_1 < x_2 < \cdots < x_n \)).

Alternatively, we use the system defined by (1) to find the nodes. In this case it turns out to be easier to solve for the \( w_j \)'s first. Using symmetry and (1) with \( k = 0 \), we find that \( 2w_1 = 2 \), so that \( w_1 = w_2 = 1 \). Then, (1) with \( k = 2 \) implies

\[
2x_1^2 = w_1x_1^2 + w_2x_2^2 = \frac{2}{3}
\]

(recall \( x_2 = -x_1 \) and \( w_1 = w_2 = 1 \)), giving the desired result.

Remark. In general, it is best to find the nodes first, and then use the \( x_j \)'s to set up a linear system (with Vandermonde coefficient matrix) which can be solved for the \( w_j \)'s.