**Lecture Notes for Math 5630/6630** **Fall 2024**

# Note 12: Quadrature Rules

*Tags: math.na* *Date: 11/05/2024*

**Disclaimer**: *This lecture note is for math 5630/6630 class only.*

## 1 Basic Quadrature Rules

The numerical quadrature finds the value of an integral

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

from the function values at a finite number of points. We are mostly interested in the following quadrature formula

$$\mathcal{I}_n(f) = (b-a)\sum_{j=0}^{n} w_j f(x_j)$$

where $x_j$ are the **nodes** and $w_j$ are the **weights**. Similar to the numerical derivatives, we also define some terminologies. The formula $\mathcal{I}_n$ is said to have **degree** $k$ **accuracy** if $\mathcal{I}_n$ is exact for all polynomials $f \in \Pi_k$. Since the integration formula is linear, the exactness can be rephrased as

$$\begin{aligned} \mathcal{I}(x^p) &= \mathcal{I}_n(x^p), \quad 0 \le p \le k, \\ \mathcal{I}(x^{k+1}) &\neq \mathcal{I}_n(x^{k+1}). \end{aligned}$$

### 1.1 Interpolation Based Rules

The interpolation-based idea is intuitive. Let $q_n(x)$ be the interpolation polynomial on the nodes $x_j$ with values $f(x_j)$, $j = 0, 1, \ldots, n$, respectively. We define the quadrature by interpolation formula as

$$\mathcal{I}_n(f) := \int_a^b q_n(x)dx.$$

The above quadrature formula is exact for all degree $n$ polynomials $f$, therefore it has <u>at least</u> degree $n$ accuracy.

**Remark 1.1.** *We have seen that the $L^\infty$ error between $f$ and $q_n$ could be large (e.g. Runge phenomenon) as $n \to \infty$. So the nodes would be important as well for quadratures.*

Using the Lagrange polynomials, we can represent

$$q_n(x) = \sum_{j=0}^{n} f(x_j) L_j(x), \quad L_j(x) = \prod_{k=0, k\neq j}^{n} \frac{x - x_k}{x_j - x_k}.$$

Then it is not difficult to derive

$$\begin{aligned} \mathcal{I}_n(f) &= \sum_{j=0}^{n} f(x_j) \int_a^b L_j(x) dx \\ &= (b-a) \sum_{j=0}^{n} f(x_j) \int_0^1 \prod_{k=0, k\neq j}^{n} \frac{t - t_k}{t_j - t_k} dt, \quad t_j = \frac{x_j - a}{b-a}. \end{aligned}$$

therefore the weights $w_j = \int_0^1 \prod_{k=0,k\neq j}^{n} \frac{t-t_k}{t_j - t_k} dt$.

**Example 1.2** (rectangle rule). *The rectangle rule is the simplest, where we choose $x_0 = \frac{a+b}{2}$ as the middle point. Then the quadrature rule writes*

$$\mathcal{I}_{0,rectangle}(f) = (b-a) f(x_0).$$

*Such a rule is exact for any linear function, so it has a degree of accuracy of one. We can see that the degree of exactness could exceed $n$. In the next chapter, we will see that the maximum degree of exactness for such a form is $2n+1$.*

**Example 1.3** (trapezoid rule). *The trapezoid rule takes $x_0 = a$ and $x_1 = b$.*

$$\mathcal{I}_{1,trapezoid}(f) = (b-a) \left( \frac{1}{2} f(x_0) + \frac{1}{2} f(x_1) \right).$$

*One can check that this rule is exact for $f(x) = 1, x$. Its degree of accuracy is one. It has a slightly larger constant in error estimate than the rectangle rule.*

**Example 1.4** (Simpson's rule). *The Simpson's rule takes $x_0 = a$, $x_1 = \frac{a+b}{2}$, and $x_2 = b$.*

$$\mathcal{I}_{2,Simpson}(f) = (b-a) \left( \frac{1}{6} f(x_0) + \frac{2}{3} f(x_1) + \frac{1}{6} f(x_2) \right).$$

*This rule is exact for $f(x) = 1, x, x^2, x^3$, thus the degree of accuracy is three.*

## 1.2 Error of Interpolation Based Rules

Now we try to estimate $|\mathcal{I}(f) - \mathcal{I}_n(f)|$ from above derivation.

**Theorem 1.5.** *Suppose the quadrature rule $\mathcal{I}_n$ has at least degree $r$ accuracy that $r \geq n$ and $f \in C^{r+1}([a,b])$. Then*

$$|\mathcal{I}(f) - \mathcal{I}_n(f)| \leq \Omega_r \frac{(b-a)^{r+2}}{(r+1)!} \max_{x\in[a,b]} |f^{(r+1)}(x)|,$$

*where the constant $\Omega_k$ is defined by*

$$\Omega_r := \min_{t_{n+1},\ldots,t_r\in[0,1]} \int_0^1 \prod_{j=0}^{r} |t-t_j| dt, \quad t_j = \frac{x_j - a}{b-a}, j = 0,\ldots,n.$$

*Proof.* Let $x_{n+1},\ldots,x_r$ be additional distinct nodes on $[a,b]$ and define $f_r$ the interpolating polynomial on the nodes $x_0,\ldots,x_r$, since the quadrature rule has degree $r$ accuracy, then

$$\mathcal{I}_n(f) = (b-a)\sum_{j=0}^{n} w_j f(x_j) = (b-a)\sum_{j=0}^{n} w_j f_r(x_j) = \mathcal{I}_n(f_r) = \mathcal{I}(f_r).$$

Using the theories developed in Interpolation, we know that

$$f(x) - f_r(x) = \frac{\omega_r(x) f^{(r+1)}(\xi)}{(r+1)!},$$

where $\omega_r(x) = \prod_{j=0}^{r}(x-x_j)$, therefore

$$|\mathcal{I}(f-f_r)| = \left| \int_a^b \frac{\omega_r(x) f^{(r+1)}(\xi)}{(r+1)!} dx \right| \leq \frac{(b-a)}{(r+1)!} \left( \int_a^b |\omega_r(x)| dx \right) \max_{x\in[a,b]} |f^{(r+1)}(x)|.$$

Since $x_{n+1},\ldots,x_r$ can be chosen arbitrarily, we select the combination that minimizes

$$\left( \int_a^b |\omega_r(x)| dx \right),$$

which will lead to our conclusion using a simple scaling. □

**Remark 1.6.** *As we can see, the error of the interpolation-based quadrature has a form similar to that of the interpolation polynomial error. This implies the Runge phenomenon would occur as well. The integration of the Runge function will not converge on uniformly distributed nodes.*

One way to overcome the issue of the Runge phenomenon is to perform piecewise integration. Suppose $[a,b]$ is divided into $N$ subintervals of equal sizes, each of which has length $H = \frac{b-a}{N}$. Then the quadrature error in each subinterval would be:

$$\Omega_r \frac{H^{r+2}}{(r+1)!} \max_{x\in[a,b]} |f^{(r+1)}(x)|$$

where $\Omega_r$ is independent of the interval length. Therefore the total quadrature error would be bounded by

$$N\Omega_r \frac{H^{r+2}}{(r+1)!} \max_{x\in[a,b]} |f^{(r+1)}(x)| = (b-a)\Omega_r \frac{H^{r+1}}{(r+1)!} \max_{x\in[a,b]} |f^{(r+1)}(x)| = \mathcal{O}((b-a)H^{r+1}).$$

**Example 1.7** (rectangle rule). *For rectangle rule, $r = 1, n = 0$, therefore*

$$\Omega_r = \min_{t_1} \int_0^1 |t - \frac{1}{2}||t - t_1|dt = \frac{1}{12}, \quad (t_1 = \frac{1}{2}).$$

*This is easiest to notice by changing variable $s = t - \frac{1}{2}$ and the symmetry, then the integral is just*

$$\Omega_r = \min_{z\in[-1/2,1/2]} \int_{-1/2}^{1/2} |s||s - z|ds = \min_{z\in[0,1/2]} \int_0^{1/2} s(|s - z|+|s + z|)ds \geq \int_0^{1/2} s(2s)ds.$$

**Example 1.8** (trapezoid rule). *For trapezoid rule, $r = n = 1$, therefore*

$$\Omega_r = \int_0^1 (1 - t)tdt = \frac{1}{6}.$$

*then the error is bounded by $\frac{(b-a)h^2}{12} \max_{x\in[a,b]}|f^{(r+1)}(x)|$.*

### 1.3 Newton-Cotes Formula

The Newton-Cotes formula is a special interpolation-based quadrature rule. The nodes are equally spaced. The rectangle and trapezoid rules are just the two simplest cases. We define

1. closed form, $x_0 = a$, $x_n = b$, $x_j = a + jh$, $h = \frac{b-a}{h}$, $n \geq 1$.
2. open form, $x_0 = a + h$, $x_n = b - h$, $h = \frac{b-a}{n+2}$, $n \geq 0$.

The difference is whether the endpoints are included or not. Using the previous result, we can compute the quadrature weights by

$$\begin{aligned} w_j &= \int_0^1 \prod_{k=0,k\neq j}^{n} \frac{t - t_k}{t_j - t_k} dt = \int_0^1 \prod_{k=0,k\neq j}^{n} \frac{nt - k}{j - k} dt \\ &= \frac{1}{n} \int_0^n \prod_{k=0,k\neq j}^{n} \frac{s - k}{j - k} ds. \end{aligned}$$

The computations of the weights can be efficient by noticing the symmetry.

**Lemma 1.9.** $w_j = w_{n-j}$.

*Proof.* This is because $w_j = \int_a^b L_j(x)dx = \int_a^b L_j(a + b - x)dx = \int_a^b L_{n-j}(x)dx = w_{n-j}$. □

The weights $w_j$ are only relevant to $n$ and $j$. In practice, these values are tabulated *a priori*. When $n \geq 2$, the weights include negative terms for open forms, and when $n \geq 8$, the weights include negative terms for closed forms, which could introduce numerical instability from rounding errors. Therefore one should only limit to small values of $n$.

**Remark 1.10.** *We can derive the error estimate for the Newton-Cotes formula using the result from the previous section.*

$$\begin{aligned}|\mathcal{I}(f) - \mathcal{I}_{n,NC}(f)| &\leq \Omega_r \frac{(b-a)^{r+2}}{(r+1)!} \\ &= M_n h^{r+2} \max_{x\in[a,b]} |f^{(r+1)}(x)|\end{aligned}$$

*where* $M_n = \Omega_r n^{r+2} \frac{1}{(r+1)!}$, *see the following table for a reference.*

| $n$ | $w_j$ | $r$ | $M_n$ |
|---|---|---|---|
| *1* | *(*1/2, 1/2*)* | 1 | 1/12 |
| *2* | *(*1/6, 2/3, 1/6*)* | 3 | 1/90 |
| *3* | *(*1/8, 3/8, 3/8, 1/8*)* | 3 | 3/80 |
| *4* | *(*7/90, 32/90, 12/90, 32/90, 7/90*)* | 5 | 8/945 |

*From the above table, we notice that when* $n$ *is even, the exactness is* $r = n + 1$ *for closed forms. This is a general statement.*

**Lemma 1.11.** *For* $n \geq 2$ *even, the closed forms of the Newton-Cotes formula have a degree of accuracy* $r = n + 1$.

*Proof.* First, we know that $r \geq n$. Consider any polynomial of degree $n + 1$,

$$p(x) = \sum_{j=0}^{n+1} b_j x^j,$$

we can rewrite the polynomial by

$$p(x) = b_{n+1}(x - \frac{a+b}{2})^{n+1} + \sum_{j=0}^{n} b'_j x^j$$

with another set of coefficients $b'_j$. The first term will have the integral as zero on the interval $[a, b]$. Numerically, using the Newton-Cotes formula,

$$\mathcal{I}_{n,NC}((x - \frac{a+b}{2})^{n+1}) = (b-a)\sum_{j=0}^{n} w_j (x_j - \frac{a+b}{2})^{n+1}$$

while $x_{n-j} - \frac{a+b}{2} = -(x_j - \frac{a+b}{2})$ and $w_j = w_{n-j}$, we can cancel all terms.

It still remains to show that $\mathcal{I}(x^{n+2}) \neq \mathcal{I}_{n,NC}(x^{n+2})$. Because the $r = (n + 1)$ degree of accuracy is achievable, we borrow the previous estimate result, let $f(x) = x^{n+2}$,

$$\begin{aligned}\mathcal{I}(f) - \mathcal{I}_n(f) &= \int_a^b \frac{\omega_{n+1}(x) f^{(n+2)}(\xi)}{(n+2)!} dx = \int_a^b \omega_{n+1}(x) dx \\ &= \int_a^b \omega_n(x)(x - x_{n+1}) dx = F(x)(x - x_{n+1})|_a^b - \int_a^b F(x) dx\end{aligned}$$

where $F(x)$ is defined by

$$F(x) := \int_a^x \omega_n(t)dt.$$

Then it is simple to derive that $F(a) = F(b) = 0$ using the symmetry. Now we only have to show that

$$\int_a^b F(x)dx \neq 0.$$

We can show a stronger claim: $F(x) > 0$ over $(a, b)$. This is left as an exercise for readers. □

However, the Newton-Cotes formula definitely will fail when evaluating the integral of Runge function $f(x) = \frac{1}{1+x^2}$ on the interval $[-5, 5]$. It is more practical to combine the piecewise integral technique, which is called the composite Newton-Cotes formula. In the following, we discretize the interval $[a, b]$ into $m$ subintervals of the same size $H = \frac{b-a}{m}$, then on each subinterval, we apply the Newton-Cotes formula (say closed form) with $(n + 1)$ equally spaced nodes. Then the numerical integral would have an error bounded by

$$mM_n \left(\frac{H}{n}\right)^{r+2} \max_{x\in[a,b]} |f^{(r+1)}(x)| = (b-a)\frac{M_n}{n}\left(\frac{H}{n}\right)^{r+1} \max_{x\in[a,b]} |f^{(r+1)}(x)|$$

where $r = n$ for odd $n$ and $r = n + 1$ for even $n$, see the previous section for a quick derivation.

**Example 1.12** (composite trapezoid rule)**.** *The composite trapezoid rule is often used for practical integration especially when $f$ is periodic. Let $x_j = a + jH$, $j = 0, \ldots, m$,*

$$T(f, H) = \frac{H}{2}\left(f(a) + 2\sum_{j=1}^{m-1} f(x_j) + f(b)\right).$$

*Its error then can be estimated by*

$$\frac{1}{12}(b-a)H^2 \max_{x\in[a,b]} |f''(x)| = \mathcal{O}((b-a)H^2).$$

In the next step, we take a more careful look at the composite trapezoid rule. Recall the asymptotic Euler-Maclaurin summation formula:

$$\sum_{j=0}^{m} g(j) \sim \int_0^m g(x)dx + \frac{g(0)+g(m)}{2} + \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!}(g^{2k-1}(m) - g^{(2k-1)}(0))$$

If we take $g(j) = f(a + jH)$, then we will arrive at

$$T(f, H) \sim \int_a^b f(x)dx + \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!} H^{2k}\left(f^{(2k-1)}(b) - f^{(2k-1)}(a)\right)$$

which means there is an asymptotic expansion in the form of

$$T(f, h) = \int_a^b f(x)dx + c_2H^2 + c_4H^4 + \cdots. \qquad (**)$$

Particularly, for a smooth periodic function, the Euler-Maclaurin summation *formally* shows the numerical error is less than any polynomial of $H$.

## 1.4 Romberg Integration

The composite trapezoid rule's asymptotic expansion (∗∗) implies a Richardson extrapolation combination to accelerate the evaluation. The Romberg integration refers to the following scheme:

1. Compute the sequence $a_{l,0} = T(f, (b-a)/2^l)$, $l = 0, \ldots, L$, for the standard composite trapezoid rule with different sub-interval sizes.
2. Extrapolation by
$$a_{l,q+1} = \frac{4^{q+1} a_{l,q} - a_{l-1,q}}{4^{q+1} - 1}, \quad q = 0, \ldots, L-1 \text{ and } l = q+1, \ldots, L$$
3. Output $a_{L,L}$, which should have an error of $\mathcal{O}(H^{2L+2})$, $H = (b-a)/2^L$.

**Remark 1.13.** *One of the advantages of the Romberg method is the reuse of the nodes. This is extremely helpful when evaluating $f$ is not cheap. The extrapolation process also builds a new quadrature formulation implicitly. This quadrature rule gives the error $\mathcal{O}(n^{-2})$ to $\mathcal{O}(n^{-2\log_2 n - 2})$, where $n$ is the total number of nodes. Although the computational time increases a few times, the return seems worth it when $f$ is sufficiently smooth.*

## 1.5 Adaptive Integration

Numerically, we can apply any composite quadrature rule to a successive partition of $[a, b]$ until the estimated error is within tolerance. Let $A(f, H)$ denote any composite quadrature rule (e.g. Newton-Cotes), $H = (b-a)/m$, then

$$A(f, H) = \int_a^b f(x)dx + \mathcal{O}(H^{r+1})$$

where $r$ is the degree of accuracy. Then $A(f, H/2)$ will presumably introduce an error of about $2^{-(r+1)}$ times the size of the previous case. Therefore, we obtain a rough estimate of the error by

$$\mathcal{E} \approx \left| \frac{A(f, H) - A(f, H/2)}{1 - 2^{-(r+1)}} \right|$$

One can successively halve $H$ until the estimated error is less than tolerance. However, such a method is not efficient when the quadrature on most of the subintervals is already very accurate. In this case, the best strategy is to keep those accurate subintervals and only partition the rest. This process will produce a non-uniform distribution of sub-intervals.

There are several ways to implement this. The simplest recursive algorithm can be roughly described as follows. Let $A(f, \alpha, \beta)$ be any quadrature rule on $[\alpha, \beta]$ with degree of accuracy $r$ and $\mathcal{E}(f, \alpha, \beta)$ be an estimate of error for $|A(f, \alpha, \beta) - \int_\alpha^\beta f(x)dx|$. Then

1. Initially, $\alpha = a$, $\beta = b$, $\varepsilon$ is the tolerance, $\mathcal{I} = 0$.

2. If $|\mathcal{E}(f,\alpha,\beta)| \leq \varepsilon \frac{\beta-\alpha}{b-a}$ or $|\beta-\alpha|$ is too small, $\mathcal{I} = \mathcal{I} + A(f,\alpha,\beta)$, stop. Otherwise, go to Step 3.

3. Divide $[\alpha,\beta]$ into $[\alpha,\gamma]$, $[\gamma,\beta]$, $\gamma = \frac{\alpha+\beta}{2}$.
   (a) For the first half, let $\alpha = \alpha$, $\beta = \gamma$, go to step 2.
   (b) For the second half, let $\alpha = \gamma$, $\beta = \beta$, go to step 2.

Ideally, the automatic partition will generate nonuniformly distributed subintervals. The total estimated numerical error will be bounded by $\varepsilon$.

# 2 Gauss Quadrature

The Gauss quadrature maximizes the exactness of quadrature rules. Let $x_0, \ldots, x_n$ the nodes on $[-1,1]$, in the following we will discuss the numerical quadrature for the weighted integral

$$\mathcal{I}_w(f) = \int_{-1}^{1} w(x) f(x) dx \simeq \mathcal{I}_{n,w}(f) := \sum_{j=0}^{n} c_j f(x_j),$$

where the coefficients are determined later. We first review the preliminaries for the theory behind the Gauss quadrature.

## 2.1 Orthogonal Polynomials

The orthogonal polynomials can be regarded as a special case of generalized Fourier series. Let the weight function $w(x) \geq 0$ on the interval $(-1,1)$ be an integrable function. Then we can define a sequence of polynomials $p_k \in \Pi_k$, that is, $\deg(p_k) = k$ and

$$\int_{-1}^{1} w(x) p_k(x) p_j(x) dx = 0, \quad \text{if } k \neq j.$$

Here, for convenience, we define the inner product $\langle f, g \rangle_w$ by

$$\langle f, g \rangle_w = \int_{-1}^{1} w(x) f(x) g(x) dx.$$

This inner product induces a norm $\|f\|_w = \sqrt{\langle f, f \rangle_w}$, we then define the corresponding space as

$$L^2_w = \{ f : (-1,1) \to \mathbb{R} \mid \|f\|_w < \infty \}.$$

Then for any $f \in L^2_w$, we can define the generalized Fourier series by $Sf$:

$$Sf = \sum_{j=0}^{\infty} a_j p_j, \quad a_j = \frac{\langle f, p_j \rangle_w}{\langle p_j, p_j \rangle_w}.$$

The series converges to $f$ in $L^2_w$ sense from the Parseval's equality:

$$\|f\|_w^2 = \sum_{j=0}^{\infty} a_j^2 \|p_j\|_w^2.$$

The truncated series $f_n = \sum_{j=0}^{n} a_j p_j$ is the best degree-$n$ polynomial approximation to $f$, that is,

$$\|f - f_n\|_w = \min_{q \in \Pi_n} \|f - q\|_w,$$

and the polynomial $f_n$ is the orthogonal projection of $f$ onto $\Pi_n$ in the sense of $L^2_w$.

**Theorem 2.1.** *The following recursive formula generates (unnormalized) orthogonal polynomials $p_j \in \Pi_j$.*

$$p_{j+1} = (x - \alpha_j)p_j(x) - \beta_j p_{j-1}(x), \quad j \geq 0.$$

*The initial conditions are $p_{-1} = 0$, $p_0 = 1$. The constants $\alpha_j$ and $\beta_j$ are*

$$\langle p_{j+1}, p_j \rangle_w = 0 \Rightarrow \alpha_j = \frac{\langle x p_j, p_j \rangle_w}{\langle p_j, p_j \rangle_w}.$$

$$\langle p_{j+1}, p_{j-1} \rangle_w = 0 \Rightarrow \beta_j = \frac{\langle p_j, p_j \rangle_w}{\langle p_{j-1}, p_{j-1} \rangle_w}.$$

*Proof.* The proof is straightforward by induction. Notice that $p_{j+1}$ defined in this formula will be automatically orthogonal to $\{p_k\}_{k=0}^{j-2}$, thus only have to determine the parameters $\alpha_j$ and $\beta_j$ to fulfill the orthogonality with $p_{j-1}$ and $p_j$. □

### 2.1.1 Chebyshev Polynomials

The Chebyshev polynomials are generated by using the weight $w(x) = (1-x^2)^{-1/2}$ on $(-1, 1)$. The corresponding space is

$$L^2_w = \{f : (-1, 1) \to \mathbb{R} \mid \int_{-1}^{1} f^2(x)(1 - x^2)^{-1/2} dx < \infty\}.$$

It is clear that by setting $p_k(x) = \cos(k \arccos x)$, the integral

$$\int_{-1}^{1} p_k(x) p_j(x)(1 - x^2)^{-1/2} dx = \int_0^{\pi} \cos(k\theta)\cos(j\theta) d\theta = \begin{cases} \pi & k = j = 0 \\ \frac{\pi}{2} & k = j \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Therefore, the generalized Fourier series with Chebyshev polynomials is

$$f(x) = \sum_{j=0}^{\infty} a_j p_j(x)$$

where $p_j(x) = T_j(x)$ the $j$-th Chebyshev polynomial and $a_j = \frac{1}{\pi}\langle f, p_j \rangle_w$ if $j = 0$ and $a_j = \frac{2}{\pi}\langle f, p_j \rangle_w$ otherwise.

### 2.1.2 Legendre Polynomials

The Legendre polynomials are generated using the weight $w(x) = 1$. The corresponding space is normal $L^2(-1, 1)$. The recursive formula for Legendre polynomials is

$$L_{j+1}(x) = \frac{2j+1}{j+1} x L_j(x) - \frac{j}{j+1} L_{j-1}(x)$$

and $\langle L_j, L_j \rangle_w = \frac{2}{2j+1}$. Therefore, the generalized Fourier series is

$$f(x) = \sum_{j=0}^{\infty} a_j L_j(x), \quad a_j = \frac{2j+1}{2} \langle f, L_j \rangle_w.$$

The first few Legendre polynomials are

$$L_0 = 1, \quad L_1(x) = x, \quad L_2(x) = \frac{1}{2}(3x^2 - 1).$$

**Remark 2.2.** *Both of the above examples are special cases of Jacobi polynomials which are generated by weight function* $w(x) = (1-x)^\alpha (1+x)^\beta$.

## 2.2 Gauss Quadrature on Bounded Domain

We borrow the same accuracy concept from the previous chapter ($w(x) = 1$). It is clear that with $n+1$ nodes, the highest possible accuracy is at least degree $n$, in the later derivation we will see that the Gauss quadrature can have an accuracy of $r = n + m$ for certain $m > 0$.

**Theorem 2.3.** *Let* $p_k \in \Pi_k$ *and* $\varpi(x) = \prod_{j=0}^{n}(x - x_j)$, *then*

$$\langle \varpi(x), p_k \rangle_w = 0, \quad 0 \le k \le m-1$$

*if and only if the associated quadrature rule has an accuracy at the order of* $n + m$.

*Proof.* The key idea is to represent any polynomial $q$ of $\Pi_{n+m}$ by

$$q(x) = \varpi(x) s(x) + t(x)$$

where $s(x) \in \Pi_{m-1}$ and $t \in \Pi_n$. Therefore the quadrature over the reminder term $t(x)$ is exact,

$$\sum_{j=0}^{n} c_j t(x_j) = \int_{-1}^{1} t(x) w(x) dx = \int_{-1}^{1} q(x) w(x) dx - \underline{\int_{-1}^{1} \varpi(x) s(x) w(x) dx}.$$

□

It is clear that the maximum of $m \leq n+1$, otherwise, we can choose $s(x) = \varpi(x)$, then $\langle \varpi(x), \varpi(x) \rangle_w > 0$ violates the above theorem. This leads to the following corollary for the Gauss quadrature.

**Corollary 2.4.** *The quadrature rule*

$$\mathcal{I}_{n,w}(f) = \sum_{j=0}^{n} c_j f(x_j)$$

*has maximum accuracy of degree* $2n+1$.

The next question would be whether the maximum $2n+1$ is achievable. This requires that

$$\int_{-1}^{1} \varpi(x) p_k(x) w(x) dx = 0, \quad 0 \leq k \leq n. \tag{***}$$

This formula indicates that $\langle \varpi(x), p_k \rangle_w = 0$ for any $p_k \in \Pi_k$, $0 \leq k \leq n$.

**Theorem 2.5.** *Let* $\{p_k\}_{k \geq 0}$ *be a sequence of orthogonal polynomials, then the only possible choice of* $\varpi$ *satisfying* $(***)$ *is* $p_{n+1}$ *(up to scaling).*

*Proof.* Without loss of generality, we assume $p_{n+1}$'s leading power's coefficient is one (for example, using the recursive formula definition). Since $p_{n+1}$ also satisfies the orthogonal relation. Therefore,

$$\int_{-1}^{1} (\varpi(x) - p_{n+1}(x)) s(x) w(x) dx = 0, \quad s \in \Pi_n$$

while $\varpi - p_{n+1} \in \Pi_n$, then we have a contradiction if we take $s(x) = \varpi - p_{n+1} \neq 0$. □

The above theorem implies that the nodes $x_j$ are the zeros of $p_{n+1}$ (we still need to show that they are simple roots). The corresponding quadrature rule is called the Gauss quadrature, and the accuracy is of degree $2n+1$.

**Remark 2.6.** *For* $w = 1$*, the Legendre polynomial's roots are not at the endpoints, while sometimes the endpoints* $\pm 1$ *are useful to be included in the quadrature nodes, therefore we may want to generate a similar Gauss quadrature with the end nodes as well. Following the same idea, in order to make* $\pm 1$ *as the roots of* $\varpi(x)$ *but also keep* $\varpi$ *concentrated on* $p_k$ *for large* $k$*, we can define*

$$\widetilde{\varpi}(x) := p_{n+1}(x) + A p_n(x) + B p_{n-1}(x),$$

*the constants* $A, B$ *are used to control* $\widetilde{\varpi}(\pm 1) = 0$*. The corresponding* $\widetilde{\varpi}$ *has a similar property as* $\varpi$ *but only provides an accuracy of degree* $2n-1$*. The nodes are roots of* $\widetilde{\varpi}$*, called Gauss-Lobatto nodes. In the following, we prove some of the properties of Gauss quadrature.*

**Theorem 2.7.** *All roots* $x_j$ *of* $p_{n+1}$ *are real and distinct.*

*Proof.* The polynomial $p_{n+1}$ must have $n+1$ real roots. Otherwise, one can generate a polynomial $q(x)$ with degree $< (n+1)$ that $\langle q, p_{n+1}\rangle > 0$. Assume some of the roots are not simple. Pick out all roots with odd multiplicity, say $z_0 < z_1 < \cdots < z_M$, then

$$q(x) = (x - z_0)\cdots(x - z_M)$$

should satisfy $q(x)p_{n+1}(x)w(x) > 0$ except for the roots. However, if $M \neq n$, we would have an issue since

$$\int_{-1}^{1} q(x)p_{n+1}(x)w(x)dx = 0.$$

□

**Theorem 2.8.** *The weights $c_j$ for Gauss quadrature rules*

$$\mathcal{I}_n(f) = \sum_{j=0}^{n} c_j f(x_j)$$

*are all positive.*

*Proof.* The polynomials $p_k$ satisfy the recursive formula

$$p_{k+1}(x) = (x - \alpha_k)p_k(x) - \beta_k p_{k-1}(x)$$

then

$$\begin{pmatrix} \alpha_0 & 1 & 0 & 0 & \cdots & 0 \\ \beta_1 & \alpha_1 & 1 & 0 & \ddots & 0 \\ 0 & \beta_2 & \alpha_2 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \beta_{n-1} & \alpha_{n-1} & 1 \\ 0 & 0 & 0 & 0 & \beta_n & \alpha_n \end{pmatrix} \begin{pmatrix} p_0(x_l) \\ p_1(x_l) \\ p_2(x_l) \\ \vdots \\ p_{n-1}(x_l) \\ p_n(x_l) \end{pmatrix} = x_l \begin{pmatrix} p_0(x_l) \\ p_1(x_l) \\ p_2(x_l) \\ \vdots \\ p_{n-1}(x_l) \\ p_n(x_l) \end{pmatrix}$$

which means the matrix $A$ on the left side has $(n+1)$ eigenpairs $\{x_l, (p_j(x_l))_{j=0}^n\}$. The tridiagonal matrix has a similar transform to a symmetric matrix, denoted by $S = D^{-1}AD$ with $D$ as a diagonal matrix; then $S$ has the same set of eigen pairs, which means that the transformed vectors

$$D^{-1} \begin{pmatrix} p_0(x_l) \\ p_1(x_l) \\ p_2(x_l) \\ \vdots \\ p_{n-1}(x_l) \\ p_n(x_l) \end{pmatrix}$$

are the eigenvectors of $S$. Since all the eigenvalues are distinct, these vectors are orthogonal, which implies

$$v_k^T D^{-2} v_j = r_j \delta_{jk} \quad r_j > 0.$$

where $v_j = (p_i(x_j))_{i=0}^n$. Take $f = p_k$ in the quadrature rule,

$$\sum_{j=0}^{n} c_j v_{jk} = \delta_{0k}$$

Combined with the two equations, we have

$$\begin{aligned}\sum_{k=0}^{n}\sum_{j=0}^{n} c_j v_{jk} v_{lk} D_{kk}^{-2} &= \sum_{k=0}^{n} v_{lk} D_{kk}^{-2} \sum_{j=0}^{n} c_j v_{jk} = D_{00}^{-2} > 0 \\ &= \sum_{j=0}^{n} c_j \sum_{k=0}^{n} v_{lk} D_{kk}^{-2} v_{jk} = r_l c_l.\end{aligned}$$

□

This result should be compared with the Newton-Cotes formula, where the weights are not all positive if $n$ is large; hence, the Gauss quadrature has better numerical stability. Finally, we briefly state the error estimate for the Gauss quadrature using the previously proved Theorem **??**, where $r = 2n + 1$.

**Corollary 2.9.** *For $f \in C^{2n+2}[-1, 1]$, the error of Gauss quadrature satisfies*

$$|\mathcal{I}(f) - \mathcal{I}_n(f)| \leq \Omega_{2n+1} \frac{(b-a)^{2n+3}}{(2n+2)!} \max_{x\in[-1,1]} |f^{(2n+2)}(x)|.$$

**Remark 2.10.** *For Chebyshev polynomials. the weight $w(x) = (1-x^2)^{-1/2}$, the Gauss quadrature nodes are roots of $T_{n+1}(x) = \cos((n+1)\arccos x)$,*

$$x_j = \cos\left(\frac{2j+1}{2(n+1)}\pi\right), \quad c_j = \frac{\pi}{n+1}.$$

*which is exactly the set of Chebyshev interpolation nodes. The Gauss-Lobatto nodes are*

$$\tilde{x}_j = \cos\left(\frac{j\pi}{n}\right), \quad c_j = \frac{\pi}{d_j n}$$

*where $d_j = 2$ if $j = 0$ or $j = n$, otherwise $d_j = 1$. This is exactly the composite trapezoid rule.*