Quadrature

M.Sc. in Mathematical Modelling & Scientific Computing
Practical Numerical Analysis

Michaelmas Term 2026, Lecture 2

Quadrature

Suppose we want to compute

$$I(f) \quad = \quad \int_a^b \mu(x) f(x) \mathrm{d}x$$

where $\mu(x)$ is a non-negative weight function (we will consider $\mu(x) \equiv 1$ for now). Unfortunately most integrals do not have closed form solutions. For example, what is

$$I(f) \quad = \quad \int_{-1}^{1} \exp(-x^2) \mathrm{d}x?$$

The idea of quadrature is to approximate $I(f)$, so

$$I(f) \quad \approx \quad I_n(f) \quad = \quad \sum_{k=0}^{n} w_k f(x_k).$$

Here

- n is the degree of the quadrature
- x_k are the quadrature nodes
- w_k are the quadrature weights

Newton Cotes Quadrature

One approach to quadrature is to approximate $f(x)$ by a polynomial interpolant, $p_n(x)$, and to integrate that interpolant. This leads to Newton-Cotes rules.

Specifically, let $p_n(x)$ be the unique polynomial of degree n which satisfies

$$p_n(x_k) \;=\; f(x_k)$$

for $0 \leq k \leq n$ where $x_k = a + k(b-a)/n$.

Then

$$I(f) \;=\; \int_a^b f(x)\mathrm{d}x \;\approx\; I_n(f) \;=\; \int_a^b p_n(x)\mathrm{d}x.$$

Integral of Linear Interpolant

The linear interpolant of $f(x)$ on $[a, b]$ can be written as

$$p_1(x) = \frac{x-a}{b-a} f(b) + \frac{b-x}{b-a} f(a).$$

Then we can write

$$\begin{aligned} I(f) \approx I_1(f) &= \int_a^b p_1(x) dx \\ &= \frac{b-a}{2}(f(a) + f(b)). \end{aligned}$$

This is the trapezium rule!

Integral of Quadratic Lagrange Interpolant

The quadratic Lagrange interpolant of $f(x)$ at the points a, $(a+b)/2$, and b can be written as

$$\begin{aligned} p_2(x) &= \frac{x-a}{b-a}\frac{2x-(a+b)}{b-a}f(b) - \frac{2(x-a)}{b-a}\frac{2(x-b)}{b-a}f\left(\frac{a+b}{2}\right) \\ &\quad + \frac{x-b}{b-a}\frac{2x-(a+b)}{b-a}f(a). \end{aligned}$$

Then we can write

$$\begin{aligned} I(f) \approx I_2(f) &= \int_a^b p_2(x)dx \\ &= \frac{b-a}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right). \end{aligned}$$

This is Simpson's rule!

Newton-Cotes

We can keep increasing n to get higher order quadrature rules.

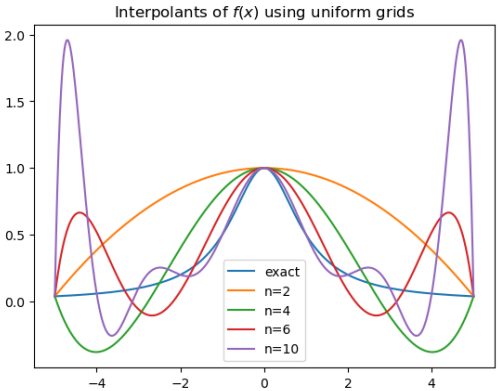
Unfortunately it turns out that the underlying interpolants can get worse as n increases, and this means the higher order quadrature rules do not work well.

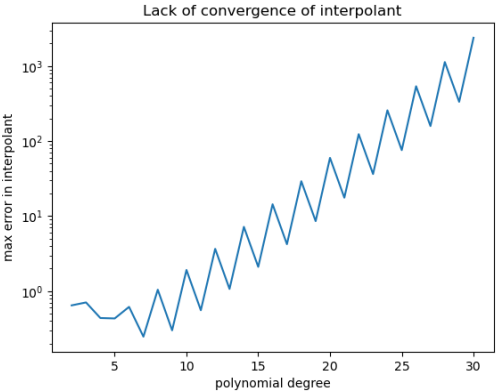
For example, consider the interpolant of the Runge function

$$f(x) = \frac{1}{1+x^2}$$

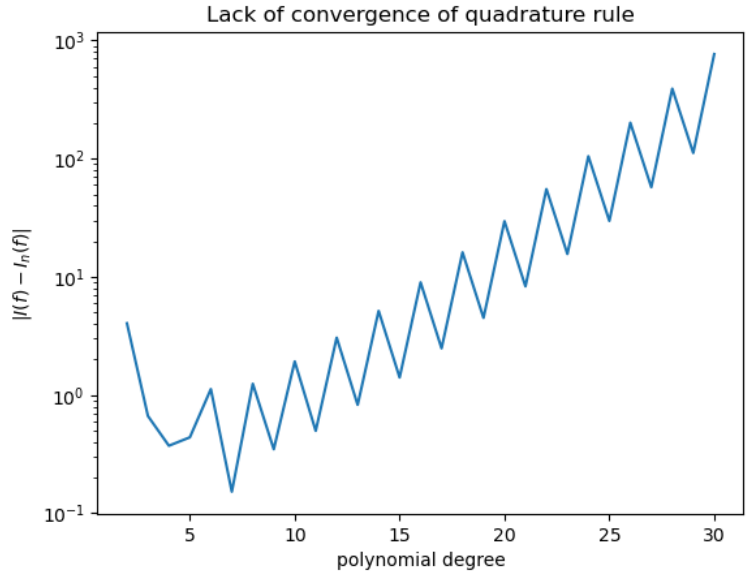
on the interval $[-5, 5]$.

Interpolant of the Runge Function





Newton Cotes Quadrature for the Runge Function



Newton-Cotes

Since Newton-Cotes is not an effective quadrature rule for high degrees we need an alternative. Possibilities are:

- Small degrees on sub-intervals — composite rules;
- Using different nodes — Gauss quadrature/ Clenshaw-Curtis.

In either case adaptivity can help improve efficiency.

Composite Trapezium Rule

Here the idea is to split the range of integration into subintervals and to apply the trapezium rule on each subinterval. Hence (with $x_k = a + kh, h = (b-a)/n$)

$$\begin{aligned}
\int_a^b f(x)\mathrm{d}x &= \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} f(x)\mathrm{d}x \\
&\approx \frac{h}{2} \sum_{k=1}^{n} (f(x_{k+1}) + f(x_k)) \\
&= \frac{h}{2} \left(f(a) + 2 \sum_{k=1}^{n-1} f(x_k) + f(b) \right) =: I_n(f).
\end{aligned}$$

Error in Composite Trapezium Rule

If $f \in C^{n+1}[a,b]$, the error in the degree n polynomial interpolant of f on $[a,b]$ can be written as

$$f(x) - p_n(x) = \frac{f^{n+1}(\xi)}{(n+1)!} \prod_{k=0}^{n} (x - x_k) \tag{1}$$

for some $\xi = \xi(x) \in (a,b)$.

For $n = 1$ this gives

$$f(x) - p_1(x) = \frac{f''(\xi)}{2}(x-a)(x-b).$$

Error in Composite Trapezium Rule

The error in the composite trapezium rule is

$$
\begin{aligned}
|I(f) - I_n(f)| &= \left| \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} f(x) - p_1(x) \mathrm{d}x \right| \\
&\leq \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} |f(x) - p_1(x)| \mathrm{d}x \\
&\leq \frac{1}{2} \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} |f''(\xi_k)(x - x_{k-1})(x - x_k)| \mathrm{d}x \\
&\leq \frac{1}{2} \max_{\xi \in [a,b]} |f''(\xi)| \sum_{k=1}^{n} \int_{x_{k-1}}^{x_k} |(x - x_{k-1})(x - x_k)| \mathrm{d}x \\
&= \frac{1}{2} \max_{\xi \in [a,b]} |f''(\xi)| \sum_{k=1}^{n} \frac{(x_k - x_{k-1})^3}{6} \\
&= \frac{h^2(b-a)}{12} \max_{\xi \in [a,b]} |f''(\xi)|.
\end{aligned}
$$

Error in Composite Trapezium Rule

Note that the points x_k do not have to be equally spaced. We can then write

$$I_n(f) \quad := \quad \sum_{k=1}^{n} \frac{x_{k+1} - x_k}{2} (f(x_{k+1}) + f(x_k)).$$

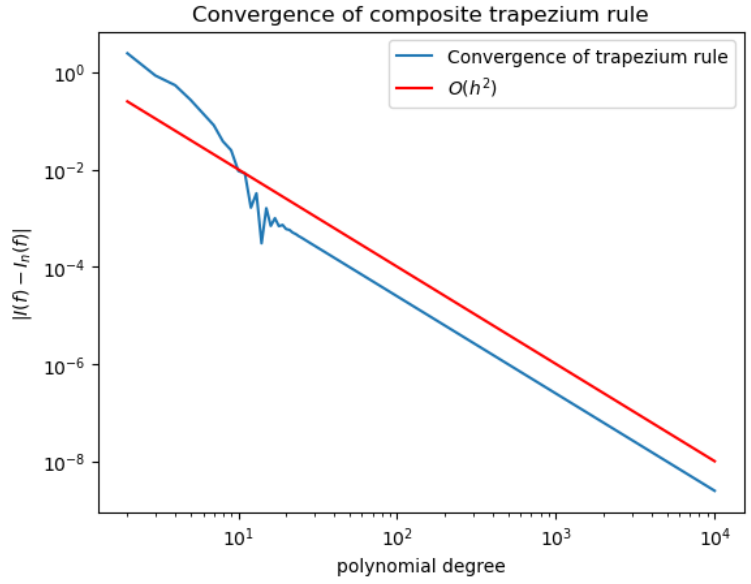
In this case, a similar analysis shows the error bound becomes

$$|I(f) - I_n(f)| \quad \leq \quad \sum_{k=1}^{n} \frac{(x_k - x_{k-1})^3}{12} \max_{\xi \in [x_{k-1}, x_k]} |f''(\xi)|,$$

and this can be used as the basis for an adaptive quadrature rule.

If f is a periodic analytic function we see geometric convergence.

Composite Trapezium for the Runge Function



Aside: Proof of Equation (1)

Clearly (1) holds for $x = x_k$, $0 \le k \le n$, so consider $x \neq x_k$ with $x \in (a, b)$. Let

$$\phi(t) \quad = \quad f(t) - p_n(t) - \frac{f(x) - p_n(x)}{\pi_{n+1}(x)} \pi_{n+1}(t)$$

where

$$\pi_{n+1}(x) \quad = \quad \prod_{k=0}^{n} (x - x_k).$$

Now $\phi(x_k) = 0$ for $0 \le k \le n$ and $\phi(x) = 0$ so $\phi(t)$ has $n + 2$ distinct roots. Rolle's theorem then implies that $\phi'(t)$ has $n + 1$ distinct roots in the interval (a, b).

Repeatedly using Rolle's theorem, we see $\phi^{n+1}(t)$ has a root $\xi(x) \in (a, b)$.

Aside: Proof of Equation (1)

We have $p_n^{n+1}(t) \equiv 0$ and $\pi_{n+1}^{n+1}(t) = (n+1)!$ so

$$\phi^{n+1}(\xi(x)) \;=\; f^{n+1}(\xi(x)) - \frac{f(x) - p_n(x)}{\pi_{n+1}(x)}(n+1)! \;=\; 0.$$

Rearranging gives the required result.

Composite Simpson's Rule

There is an analogous composite Simpson's rule

$$I_n(f) = \frac{h}{3}\left(f(a) + 2\sum_{k=1}^{n/2-1} f(x_{2k}) + 4\sum_{k=1}^{n/2} f(x_{2k-1}) + f(b)\right)$$

where n must be even. The approximation error is

$$|I(f) - I_n(f)| \leq \frac{h^4(b-a)}{180} \max_{\xi\in[a,b]} |f^{(4)}(\xi)|.$$

Clenshaw-Curtis Quadrature

In Clenshaw-Curtis quadrature, we integrate the degree n interpolant of f over $[-1, 1]$ where the interpolation is performed at second kind Chebyshev points. (We can perform a change of variable first if we wish to integrate over other intervals.)

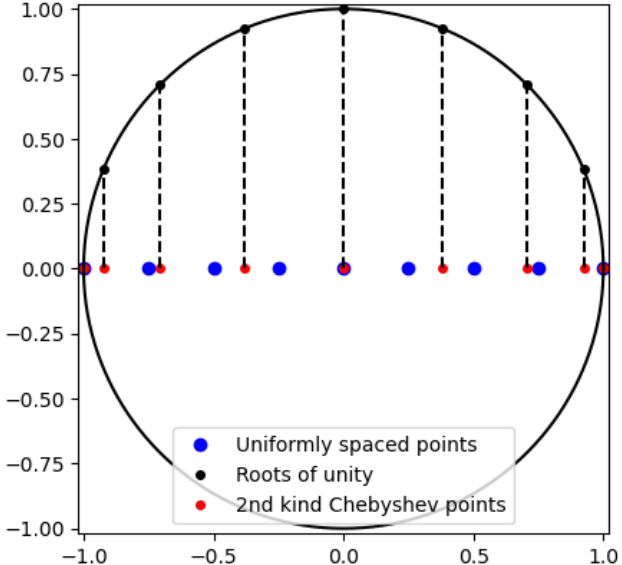
The second kind Chebyshev points are

$$x_k = \cos(k\pi/n),$$

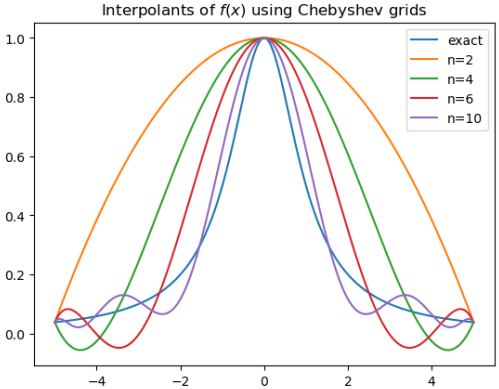
for $0 \leq k \leq n$, and they are equally spaced on the unit circle.

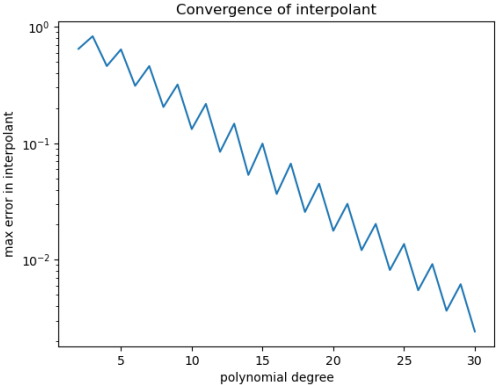
The resulting quadrature rules inherit the accuracy of the interpolant so, if f is analytic, $|I(f) - I_n(f)| \sim \mathcal{O}(\rho^{-n})$ as $n \to \infty$.

Grid Points

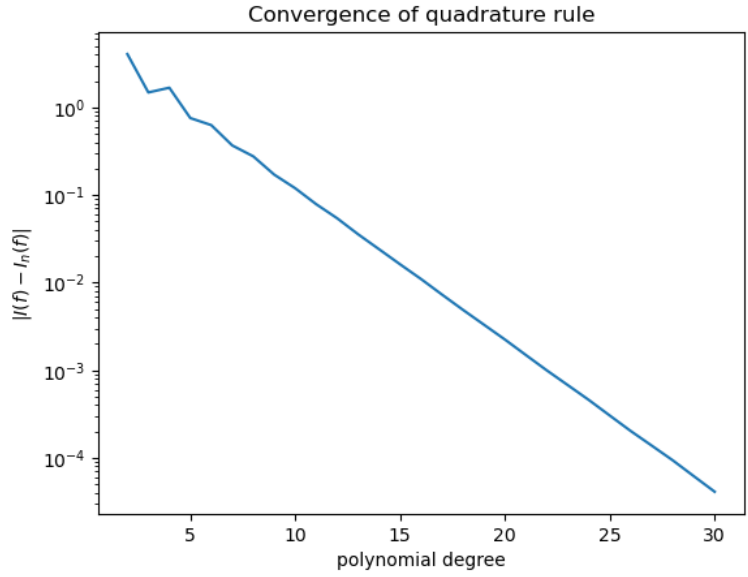

1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.0
−0.5
0.0
0.5
1.0
Uniformly spaced points
Roots of unity
2nd kind Chebyshev points

Interpolant of the Runge Function: Chebyshev Points





Clenshaw-Curtis for the Runge Function



Clenshaw-Curtis Quadrature: Coefficient Space

For a practical quadrature algorithm, it helps to re-write the interpolant as

$$p_n(x) \;=\; \sum_{k=0}^{n} \alpha_k \, T_k(x),$$

where $T_k(x)$ is the degree k first kind Chebyshev polynomial.

The interpolation conditions become

$$p_n(x_\ell) \;=\; \sum_{k=0}^{n} \alpha_k \, T_k(x_\ell) \;=\; f(x_\ell)$$

which can be written as

$$V\boldsymbol{\alpha} \;=\; \mathbf{f}$$

where $V \in \mathbb{R}^{(n+1)\times(n+1)}$ has entries $V_{\ell,k} = T_k(x_\ell)$ for $0 \le k, \ell \le n$.

Clenshaw-Curtis Quadrature: Coefficient Space

The Chebyshev polynomials are defined as

$$T_k(x) = \cos(k \cos^{-1}(x))$$

for $x \in [-1, 1]$. Thus we have

$$\begin{aligned}
\int_{-1}^{1} T_k(x)\mathrm{d}x &= \int_{-1}^{1} \cos(k \cos^{-1}(x))\mathrm{d}x \\
&= \int_{0}^{\pi} \cos(k\theta)\sin(\theta)\mathrm{d}\theta \\
&= \begin{cases} 0 & \text{for } k \text{ odd} \\ \frac{2}{1-k^2} & \text{for } k \text{ even.} \end{cases}
\end{aligned}$$

Thus we may write the quadrature rule as

$$I(f) \approx I(p_n) = \sum_{\substack{k=0 \\ k \text{ even}}}^{n} \frac{2\alpha_k}{1-k^2}.$$

Clenshaw-Curtis Quadrature: Value Space

An alternative approach is to find the weights, w_k, such that $I(p_n) = \sum_{k=0}^{n} w_k f(x_k)$.

We can do this by ensuring that the quadrature rule integrates each T_k exactly, i.e. we require

$$\sum_{k=0}^{n} w_k T_j(x_k) \quad = \quad \int_{-1}^{1} T_j(x)\mathrm{d}x,$$

for $0 \leq j \leq n$. This time we solve $V^T \mathbf{w} = \mathbf{b}$ where

$$\begin{aligned} b_j &= \int_{-1}^{1} T_j(x)\mathrm{d}x \\ &= \begin{cases} 0 & \text{for } j \text{ odd} \\ \frac{2}{1-j^2} & \text{for } j \text{ even.} \end{cases} \end{aligned}$$

Gauss Quadrature

So far we have fixed the nodes of a quadrature rule and then chosen the weights to integrate the corresponding polynomial interpolant exactly.

An alternative is to choose both the weights and nodes in the formula

$$I_n(f) \quad = \sum_{k=0}^{n} w_k f(x_k).$$

Here there are $2n+2$ unknowns ($n+1$ nodes and $n+1$ weights) and so these can be chosen to integrate all polynomials of degree $2n+1$ exactly. This is the idea behind Gauss quadrature.

Gauss Quadrature: Derivation

Orthogonal polynomials on an interval $[a, b]$ with respect to a weight function $\mu(x)$ are defined to be the polynomials $P_0(x), P_1(x), \ldots$ such that $P_k(x)$ is a polynomial of degree k and the orthogonality property

$$\int_a^b \mu(x) P_j(x) P_k(x) \mathrm{d}x \quad = \quad 0,$$

holds, whenever $j \neq k$.

Now define $\mathbb{P}_k$ to be the set of polynomials of degree k. Then, for $f_{2n+1} \in \mathbb{P}_{2n+1}$ we may write

$$f_{2n+1}(x) \quad = \quad q_n(x) P_{n+1}(x) + r_n(x)$$

for some $q_n, r_n \in \mathbb{P}_n$.

Gauss Quadrature: Derivation

Then we have

$$
\begin{aligned}
I(f_{2n+1}) &= \int_a^b \mu(x) f_{2n+1}(x)\mathrm{d}x \\
&= \int_a^b \mu(x) q_n(x) P_{n+1}(x)\mathrm{d}x + \int_a^b \mu(x) r_n(x)\mathrm{d}x \\
&= \sum_{k=0}^{n} \alpha_k \int_a^b \mu(x) P_k(x) P_{n+1}(x)\mathrm{d}x + \int_a^b \mu(x) r_n(x)\mathrm{d}x \\
&= 0 + \int_a^b \mu(x) r_n(x)\mathrm{d}x.
\end{aligned}
$$

Gauss Quadrature: Derivation

Now let the $\{x_k\}_{k=0}^n$ be the $n+1$ roots of $P_{n+1}(x)$, then the quadrature rule gives

$$\begin{aligned} I_n(f_{2n+1}) &= \sum_{k=0}^{n} w_k f_{2n+1}(x_k) \\ &= \sum_{k=0}^{n} w_k q_n(x_k) P_{n+1}(x_k) + \sum_{k=0}^{n} w_k r_n(x_k) \\ &= \sum_{k=0}^{n} w_k r_n(x_k). \end{aligned}$$

Thus, if we choose the weights w_k such that polynomial interpolants through the x_k are integrated exactly, we have

$$I(f_{2n+1}) = I_n(f_{2n+1}).$$

Gauss Quadrature: Examples

Different ranges of integration and different weight functions lead to different sets of orthogonal polynomials which define Gauss quadrature rules. Common examples are:

Name	Interval	Weight
Gauss-Legendre	$[-1, 1]$	1
Gauss-Chebyshev	$[-1, 1]$	$1/\sqrt{(1-x^2)}$
Gauss-Jacobi	$[-1, 1]$	$(1+x)^{\alpha}(1-x)^{\beta}$
Gauss-Laguerre	$[0, \infty)$	$\exp(-x)$
Gauss-Hermite	$(-\infty, \infty)$	$\exp(-x^2)$

Gauss Quadrature: Computing the Nodes

Orthogonal polynomials satisfy a recurrence relation

$$\gamma_k P_{k-1}(x) + \beta_k P_k(x) + \gamma_{k+1} P_{k+1}(x) \quad = \quad x P_k(x).$$

(Note this symmetry also requires the polynomials to be orthonormal.)

We can write this in matrix form as

$$\begin{pmatrix} \beta_0 & \gamma_1 & & & \\ \gamma_1 & \beta_1 & \gamma_2 & & \\ & \gamma_2 & \beta_2 & \gamma_3 & \\ & & \ddots & \ddots & \ddots \\ & & & \gamma_n & \beta_n \end{pmatrix} \begin{pmatrix} P_0(x) \\ P_1(x) \\ P_2(x) \\ \vdots \\ P_n(x) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ \gamma_{n+1} P_{n+1}(x) \end{pmatrix} = x \begin{pmatrix} P_0(x) \\ P_1(x) \\ P_2(x) \\ \vdots \\ P_n(x) \end{pmatrix}$$

or equivalently

$$T\mathbf{P}(x) + \gamma_{n+1} P_{n+1}(x) \mathbf{e}_{n+1} = x\mathbf{P}(x).$$

Thus the roots of $P_{n+1}(x)$ are the solution of a symmetric tridiagonal eigenvalue problem, i.e. $P_{n+1}(x) = 0$ if and only if

$$T\mathbf{v} \quad = \quad x\mathbf{v}.$$

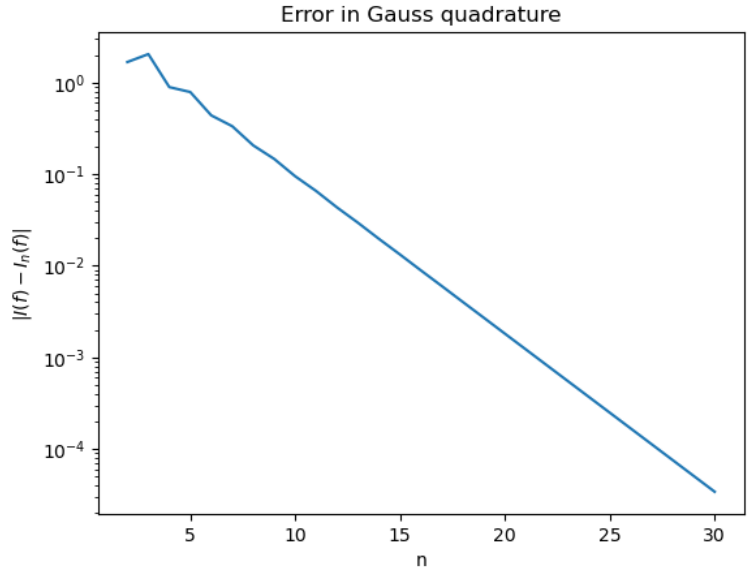
Gauss Quadrature: Computing the Weights

Once we have computed the nodes x_k, we can compute the weights as

$$w_k = \frac{v_{1,k}^2}{(P_0(x_k))^2}$$

where $v_{1,k}$ is the first entry of the normalised eigenvector corresponding to eigenvalue (node) x_k.

Gauss Legendre for the Runge Function



Summary

- Newton Cotes quadrature is not a good choice for high degrees. We haven't even looked at an algorithm.
- Using low degree Newton Cotes on subintervals (composite rules) works well.
- Composite trapezium is usually $\mathcal{O}(h^2)$, but we get geometric convergence for periodic analytic functions.
- Clenshaw-Curtis — which uses second Chebyshev points — works well for high degrees. The error is $\mathcal{O}(\rho^{-n})$ if f is analytic.
- Another option is Gauss quadrature which can also be easily extended to high degrees.