Gaussian Quad

Wednesday, November 8, 2023 2:05 PM

What quadratures do we have so far?

Calculus - left / Right / Mid pt rule

Newton-Cotes $\begin{cases} \text{Trapezoidal} \\ \text{Simpsons} \end{cases}$

Idea: $f(x) \sim \sum_{j=0}^{n} f(x_j)\, \ell_j(x)$ where $\ell_j(x) := $ Lagrange Polynomials

Then $\int_a^b f(x)\,dx \sim \sum_{j=0}^{n} f(x_j) \underbrace{\int_a^b \ell_j(x)\,dx}_{w_j \,=\, \text{quad weight}}$

Then we also have $\underbrace{\text{Composite Quadrature}}$.

Romberg extrapolation: post-processing to get high order.

For these integration technique the nodes we fixed so only the weights were free to help us create a quad exact to certain polynomial order.

For gaussian quadrature, we say why fix the nodes to start? let's have more degrees of freedom

$$\text{nodes} = \{x_j\}_{j=0}^{n} \;\; \& \text{ weights } \{w_j\}_{j=0}^{n}$$

are free for you to chose.

How many degrees of freedom? $2(n+1) = 2n+2$

We had this much freedom when making the Hermite polynomials. So our goal is to choose the nodes & weight so we can integrate polynomials of degree $2n+1$ exactly.

Ex: Fix our interval to $[-1,1]$. Suppose we have 1 node & 1 weight to find so that we can integrate the largest degree polynomial possible.

Soln: We should be able integrate 1 & x exactly.

$$Q = w_0 f(x_0)$$

f	exact		quad.
1	$\int_{-1}^{1} 1\,dx =$	$\boxed{2 =}$	$\boxed{w_0}$
x	$\int_{-1}^{1} x\,dx =$	$0 =$	$w_0 x_0$

$x_0 = 0.$

$x \quad \int_{-1} x\,dx \quad 0 \quad w_0\, x_0$

$x_0 = 0.$

$$Q = 2f(0) \sim \int_{-1}^{1} f(x)\,dx$$

This is the Midpt Rule. It is the simplest Gaussian Quadrature.

<u>Ex:</u> let's try 2 nodes & 2 weights.
Goal: find x_0, x_1 & w_0, w_1 st

$$w_0 f(x_0) + w_1 f(x_1) = \int_{-1}^{1} f(x)\,dx$$

when f is a polynomial of degree ≤ 3.

<u>Soln:</u>

f	Exact		Quad
1	2	$=$	$w_0 + w_1$
x	0	$=$	$w_0 x_0 + w_1 x_1$
x^2	$\int_{-1}^{1} x^2\,dx = \frac{2}{3}$	$=$	$w_0 x_0^2 + w_1 x_1^2$
x^3	$\int_{-1}^{1} x^3\,dx = 0$	$=$	$w_0 x_0^3 + w_1 x_1^3$

side:
$w_0 = w_1 = 1$
$x_{0,1} = \pm \frac{\sqrt{3}}{3}$

By just adding 1 node & weight we have ended up w/ a complicated non-linear system of equations.

We would like a better way.

Don't worry! there is a better way.

let $p \in \mathbb{P}_{2n+1}$ let $\{\phi_j\}$ denote the set of Legendre polynomials.

Recall Legendre polynomials are orthogonal.

$$\langle \phi_j, \phi_k \rangle = 0 \quad \text{if } j \neq k$$

$$p(x) = g(x)\,\phi_{n+1}(x) + r(x)$$

$$\deg(\phi_{n+1}) = n+1 \qquad \deg(g) \leq n \qquad \deg(r) \leq n$$

For my Quad. I want to integrate p(x) exactly.

$$\int_{-1}^{1} p(x)\,dx = \int_{-1}^{1} g(x)\,\underset{=0}{\phi_{n+1}(x)}\,dx + \int_{-1}^{1} r(x)\,dx \quad = \text{Exact Calc}$$

$$= \sum_{j=0}^{n} g(x_j)\,\phi_{n+1}(x_j)\,w_j + \sum_{j=0}^{n} r(x_j)\,w_j \quad \text{Quad}$$

$$\int_{-1}^{1} g(x)\,\phi_{n+1}(x)\,dx = 0$$

$$\text{Since } g(x) = \sum_{\ell=0}^{n} a_\ell\,\phi_\ell(x)$$

We need $\sum_{j=0}^{n} g(x_j)\, \varphi_{n+1}(x_j)\, w_j = 0 \quad \forall$ choices of $g(x)$

$\Rightarrow$ our quad nodes are the roots of $\varphi_{n+1}(x)$.

Then choose you weights so that you can integrate polynomials of degree n exactly.

or go use Newton-cotes technique.

i.e. $w_j = \int_{-1}^{1} \ell_j(x)\, dx$

$\uparrow$ Lagrange

Ex: let's revisit the 2pt quad rule

Soln: From before $w_0 = w_0 = 1 \quad x_{0,1} = \pm \sqrt{3}/3$

$$P_2(x) = \frac{3}{2}x^2 - \frac{1}{2} \quad \text{(from recursion)}$$

roots are $x = \pm \sqrt{3}/3$

$$w_0 = \int_{-1}^{1} \underbrace{\frac{x - \sqrt{3}/3}{\left(-\frac{\sqrt{3}}{3} - \frac{\sqrt{3}}{3}\right)}}_{\text{Lagrange}} dx = 1$$

Iroperty o- Legendre Polynomials

1- They are mutually orthogonal

$$\int_{-1}^{1} P_n(x)\, P_m(x)\, dx = \frac{2}{2n+1}\,\delta_{nm}$$

$$\text{Where} \quad \delta_{mn} = \begin{cases} 0 & \text{if } m \neq n \\ 1 & \text{if } m = n \end{cases}$$

2- They are the solution of the following differential eqn.

$$\frac{d}{dx}\left((1-x^2)\frac{dP_n}{dx} \right) + n(n+1)\,P_n(x) = 0$$

3- You can show that there is an analytic formula for the weight

$$w_j = \frac{2}{(1-x_j^2)\left(P_n'(x_j)\right)^2}$$

<u>seudo-algorithm</u>

Input: I = interval , $n+1$ = number nodes
$w(x) \geq 0$ weight function

Output: nodes $\{x_i\}_{i=0}^{n}$ & weights $\{\omega_i\}_{i=0}^{n}$

Step 1: build your orthogonal polynomials via Gram-schmidt. need up P_{n+1}

Step 2: find the roots of $P_{n+1}(x)$ = quad nodes

Step 3: build weights $$\omega_j = \int_I \underbrace{l_j(x)}_{\text{Lagrange}} \underbrace{w(x)}_{\text{weight function}} dx$$

→ in Class, we made $\underbrace{\text{Gauss}}_{\text{2n+1 polynomial integrate exactly}}$ - $\underbrace{\text{Legendre}}_{\text{weight function + Interval}}$ quad

You'll also see Gauss - Chebychev

$$w(x) = \frac{1}{\sqrt{1-x^2}} \qquad I = [-1, 1]$$

NODES roots of Chebychev polynomials
weights $\omega_j = \pi/2$ ← verify