Numerical Integration and Differentiation

(Theories and its Implementation)

Changmin Hyun[1]

[1]Department of Computational Science and Engineering,
Yonsei University

July 31, 2017

Contents

Current Contents

Differentiation via Polynomial Interpolation

Let $\{(x_i, y_i)\}_{i=0}^{n}$ be given. Then, we are able to determine unique $p(x)$ such that $p \in P_n$.

Consider Lagrange ith basis $l_i(x)$ defined by

$$l_i(x) := \prod_{j=0}^{n} \frac{(x - x_j)}{(x_i - x_j)}.$$

Then,

$$p(x) = \sum_{i=0}^{n} f(x_i) l_i(x).$$

Furthermore, $f(x) \approx p(x)$. So,

$$f'(x) = \sum_{i=0}^{n} f(x_i) l_i'(x).$$

Error Analysis

Suppose $f \in C^{n+1}(\Omega)$. Then,

$$f'(x_\alpha) = \sum_{i=0}^{n} f(x_i) l_i'(x_\alpha) + \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{j=0, j\neq\alpha}^{n} (x_\alpha - x_j)$$

This theorem implies approximation of first order derivative at given node is more exact, if the number of points is more larger.

Introduction to finite difference approach

Differentiation in 1-D is defined by

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

for fixed x Infinite sense can be replaced by finite sense.
So, Computing $f'(x)$ can be accomplished by

$$f'(x) = \frac{f(x+h) - f(x)}{h} \text{ for small enough } h$$

This is called **Finite Difference Method(FDM).**

Type of FDM

First Derivative Approximation

1. $$f'(x) \approx \frac{f(x+h) - f(x)}{h}. \text{ (Forward Difference)}$$
2. $$f'(x) \approx \frac{f(x) - f(x-h)}{h}. \text{ (Backward Difference)}$$
3. $$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}. \text{ (Central Difference)}$$

Q1) What is the difference between each methods?

Q2) How do we derive this formula?

Consider $f(x)$,where $f : \Omega \subset R \to R$ and $f \in C^3(\Omega)$.
By Taylor theorem, for some $\xi_1 \in [x, x+h], \xi_2 \in [x-h, x]$,

$$f(x+h) = f(x) + f'(x)h + f''(x)\frac{h^2}{2!} + f^{(3)}(\xi_1)\frac{h^3}{3!} \tag{1}$$

$$f(x-h) = f(x) - f'(x)h + f''(x)\frac{h^2}{2!} - f^{(3)}(\xi_2)\frac{h^3}{3!} \tag{2}$$

From equation (1),

$$f'(x) = \frac{f(x+h) - f(x)}{h} - f''(x)\frac{h}{2!} - f^{(3)}(\xi_1)\frac{h^2}{3!}$$

From equation (2),

$$f'(x) = \frac{f(x) - f(x-h)}{h} + f''(x)\frac{h}{2!} - f^{(3)}(\xi_1)\frac{h^2}{3!}$$

From combination between equation (1) and equation(2),

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - (f^{(3)}(\xi_1) + f^{(3)}(\xi_2))\frac{h^2}{2 \times 3!}$$

Summary

1. Forward Difference Formula $\sim O(h)$
2. Backward Difference Formula $\sim O(h)$
3. Central Difference Formula $\sim O(h^2)$

In same argument,

Second Derivative Approximation

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{2h^2} \text{ with } O(h)$$

Introduction to Richardson Extrapolation

Let $\phi(h)$ be approximation of L. Notice that $\phi(h)$ is dependent to interval length h. For easy explaination, we restrict ourselves as assuming $L = \phi(h) + a_1 h + a_2 h^2 + \cdots$. Then,

$$L = \phi(h) + a_1 h + a_2 h^2 + \cdots \tag{3}$$

$$L = \phi(\frac{h}{2}) + a_1 \frac{h}{2} + a_2 \frac{h}{2}^2 + \cdots \tag{4}$$

Combining equation (3) and equation (4) for removing first error term, we get

$$L = 2\phi(\frac{h}{2}) - \phi(h) - \frac{a_2}{2} h^2 + \cdots$$

In result, we have more accurate approximation $2\phi(\frac{h}{2}) - \phi(h)$ if step size is fixed.

General Richardson Extrapolation

In general, Assume $L = \phi(h) + a \times h^p + O(h^r)$, where $p \geq 1, r > p, q \in \mathbb{N}$. Then,

$$L = \phi(h) + a \times h^p + O(h^r) \tag{5}$$

$$L = \phi(\frac{h}{q}) + a \times (\frac{h}{q})^p + O(h^r) \tag{6}$$

Combining equation (3) and equation (4) for removing first error term, we get

$$L = \phi(h) + \frac{\phi(h) - \phi(\frac{h}{q})}{q^{-p} - 1} + O(h^r).$$

Richardson Extrapolation Algorithm

Algorithm

1. Select a convenient h (say $h = 1$) and compute
$$D(n,0) = \phi(\frac{h}{2^n})$$
for $n = 0, 1, \cdots, M$.
2. Compute additional quantities by the formula
$$D(n,k) = \frac{4^k}{4^k - 1} D(n, k-1) - \frac{1}{4^k - 1} D(n-1, k-1)$$
for $k = 1, 2, \cdots, M$ and $n = k, k+1, \cdots, M$.

Error Analysis

For fixed k,

$$D(n,k) = L + O(h^{2k+2})$$

. Precisely,

$$D(n,k) = L + \sum_{j=k+1}^{\infty} A_{j(k+1)} (\frac{h}{2^n})^2 j$$

Current Contents

Introduction to Quadrature Rule

Consider definite integral $I(f) = \int_a^b f(x)dx$. Many numerical integration rule can be represented by $I(f) \approx Q(f_n)$,where f_n is approximation of f.
In brief,

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} A_i f(x_i).$$

Furthermore, $\{A_i\}$ is called as weights.

Newton-Cote Rule

The main idea of Newton-Cote rule is to approximate f as interpolant f_n when n+1 nodes are given.
Consider family of basis functions $\{\phi_i(x)\}$. then,

$$f(x) \approx \sum_{i=0}^{n} c_i \phi_i(x)$$

for some c_i. If we choose $l_i(x)$, which is ith lagrange basis function, as $\phi_i(x)$, then, $c_i = f(x_i)$. So,

$$f(x) \approx f_n(x) := \sum_{i=0}^{n} f(x_i) l_i(x).$$

Therefore,

$$\int_a^b f(x)dx \approx \int_a^b f_n(x)dx$$

$$= \int_a^b \sum_{i=0}^{n} f(x_i) l_i(x) dx = \sum_{i=0}^{n} (\int_a^b l_i(x)dx) f(x_i)$$

If we chooose $\displaystyle\int_a^b l_i(x)dx$ as A_i, then, we get Newton Cote rule

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} A_i f(x_i).$$

Speical Cases of Newton-Cotes Rule

The well-known speical cases of Newton-Cotes rule are like followings:

1. Let $n = 0, x_0 = a \rightarrow$ Riemann lower sum,

$$\int_a^b f(x)dx \approx (b-a)f(a)$$

2. Let $n = 0, x_0 = b \rightarrow$ Riemann Upper sum,

$$\int_a^b f(x)dx \approx (b-a)f(b)$$

Speical Cases of Newton-Cotes Rule

1. Let $n = 0, x_0 = \dfrac{a+b}{2}$ $\rightarrow$ Midpoints Rule,
$$\int_a^b f(x)dx \approx (b-a)f(\frac{a+b}{2})$$
2. Let $n = 1, x_0 = a, x_1 = b$ $\rightarrow$ Trapezoidal Rule,
$$\int_a^b f(x)dx \approx \frac{(b-a)}{2}(f(a)+f(b))$$
3. Let $n = 2, x_0 = a, x_1 = \dfrac{a+b}{2}, x_2 = b$ $\rightarrow$ Simpson's Rule,
$$\int_a^b f(x)dx \approx \frac{b-a}{6}(f(a)+4f(\frac{a+b}{2})+f(b))$$

If you want more accurate formula, you should take more nodes for approximating f.

Stability of Newton Cotes Rule

$$|Q_n(\tilde{f}) - Q_n(f)| = |\sum_{i=0}^{n} A_i(\tilde{f}(x_i) - f(x_i)))| \leq (\sum_{i=0}^{n} |A_i|)||\tilde{f} - f||_\infty.$$

So, $A_i \geq 0$ implies stability of algorithm.

Sensitivity and Newton-Cotes rule

Every n point Newton-Cotes rule with $n \geq 11$ has at least one negative weight, and $\sum_{i=1}^{n} |w_i| \to \infty$ as $n \to \infty$. So, Newton-Cotes rules become arbitrarily ill-conditioned.

Accuracy of Newton Cotes Rule

Midpoints Rule

$$I(f) = M(f) + E(f) + F(f) + \cdots$$

, where $m = \dfrac{a+b}{2}, M(f) = f(m)(b-a), E(f) = \dfrac{f''(m)}{24}(b-a)^3, F(f) = \dfrac{f^{(4)}(m)}{1920}(b-a)^5.$

Trapezoidal Rule

$$I(f) = T(f) - 2E(f) - 4F(f) + \cdots$$

, where $m = \dfrac{a+b}{2}, T(f) = \dfrac{b-a}{2}[f(a) + f(b)], E(f) = \dfrac{f''(m)}{24}(b-a)^3, F(f) = \dfrac{f^{(4)}(m)}{1920}(b-a)^5.$

Simpson's Rule

$$I(f) = S(f) - \frac{2}{3}F(f) + \cdots$$

, where

$m = \frac{a+b}{2}, T(f) = \frac{2}{3}M(f) + \frac{1}{3}T(f), F(f) = \frac{f^{(4)}(m)}{1920}(b-a)^5.$

Accuracy of Newton-Cotes Rule

Odd-order Newton-Cotes rule gains extra degree and n points Newton-Cotes rule is of degree $n-1$ if n is even, but of degree n if n is odd. This phenomenon is due to cancellation of positive and negative errors.

Composite Newton-Cotes Rule

Consider a partition $a = x_0 < x_1 < \cdots < x_n = b$ and apply the Newton-Cote rule to each subinterval to obtain the composite Newton-Cote rule :

1. General Composite Trapezoidal Rule

$$\int_a^b f(x)dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} f(x)dx \approx \sum_{i=1}^{n} \frac{1}{2}(x_i - x_{i-1})(f(x_i) + f(x_{i-1}))$$

2. General Composite Simpson's Rule

$$\int_a^b f(x)dx = \sum_{i=1}^{\frac{n}{2}} \int_{x_{2i}}^{x_{2i-2}} f(x)dx \approx$$

$$\sum_{i=1}^{\frac{n}{2}} \frac{h_i}{3}(f(x_{2i-2} + 4f(x_{2i-1}) + f(x_{2i}))$$

Undetermined Coefficients Method

Recall $\int_a^b f(x)dx \approx \sum_{i=0}^{n} A_i f(x_i)$ when $n+1$ nodes $\{x_i\}$ is given.
Therefore, It just remains to evaluate $n+1$ weights $\{A_i\}$. So, we have $n+1$ degree of freedom. For restriction of freedom, we use degree of freedom for integral to be exact up to $n+1$ degree. namely, up to P_n.

Example

Consider numerical integration rule

$$\int_0^1 f(x)dx = A_0 f(0) + A_1 f(1).$$

Then,

$$A_0 = \frac{1}{2}, A_1 = \frac{1}{2}.$$

Error Analysis

So, we are able to gain at least order n accuracy by using $n+1$ degree of freedom for weights when a $n+1$ nodes are given.

Error Analysis Theorem

Let $f(x)$ be given function and $p(x)$ be interpolant with n points. Then,

$$\int_a^b f(x)dx - \sum_{i=1}^{n} A_i f(x_i) = \frac{1}{(n+1)!}\int_a^b f^{(n+1)}(\xi)\prod_{i=1}^{n}(x-x_i)dx.$$

Furthermore, If $|f^{(n+1)}(x)| \leq M$ for some constant M,

$$\int_a^b f(x)dx - \sum_{i=1}^{n} A_i f(x_i) \leq \frac{M}{(n+1)!}\int_a^b \prod_{i=1}^{n}(x-x_i)dx$$

The choice of $\{x_i\}$ to minimize polynomial is crucial part for **Clenshaw-Curtis Quadrature Rules**. That is

$$\text{minimize} \int_a^b \prod_{i=1}^{n} (x - x_i) dx$$

$$\Longleftrightarrow$$

$$\{x_i\} \text{ are roots of second chebyshev polynomial with order n} + 1$$

Second Chebyshev Polynomial

$$U_{n+1}(x) = \frac{\sin[\cos(n+2)\cos^{-1} x]}{\sin(\cos^{-1} x)}$$

If we use roots of second Chebyshev polynomial as nodes for quarature rule, we gain

$$\left| \int_a^b f(x)dx - \sum_{i=1}^{n} A_i f(x_i) \right| \leq \frac{M}{2^n (n+1)!}$$

Properties

1. Weights are always positive and approximate integral always converges to exact integral as $n \to \infty$.
2. Clenshaw-Curtis quadrature has many attractive features, but still does not have maximum possible degree for number of nodes used.

Now, we gonna have maximum accuracy as possible. This is motivation of Gaussian quadrature rule.

General Gauss Quadrature Rule

Again, recall quadrauture rule $\int_a^b f(x)dx \approx \sum_{i=0}^{n} A_i f(x_i)$. If any condition for nodes is not given, we have $2n+2$ degree of freedom. If we use all freedom for integration to be exact up to maximum degree as possible, we are able to obtain $2n+1$ order accuracy. This strategy is called as **Gaussian Quadrature Rule**. In general, For given weight function $w(x)$, we want to find

$$\{A_i\}_{i=0}^{n}, \{x_i\}_{i=0}^{n}$$

, where

$$\int_a^b f(x)w(x)dx \approx \sum_{i=0}^{n} A_i f(x_i)$$

is exact up to $2n+1$ degree at least.

Example

Consider numerical integration formula,

$$\int_{-1}^{1} f(x)dx \approx A_0 f(x_0) + A_1 f(x_1).$$

Find $\{A_i\}_{i=0}^{1}, \{x_i\}_{i=0}^{1}$.

This integration rule is exact up to 3 degree. So, $f = 1, f = x, f = x^2, f = x^3$ will make equations with respect to A_i, x_i. But, this equation is not linear. Therefore, we should find other strategy for computing nodes and weights.

Error Analysis

Theorem

Let w be a positive weight function and let q be a nonzero polynomial of degree $n+1$ taht is w-orthogonal to n in the sense that $\forall\, p \in P_n$,

$$\int_a^b q(x)p(x)w(x)dx = 0.$$

if $x_0, \cdots, x_n$ are the zeros of q, then the quadrature formula with coefficients

$$A_i = \int_a^b w(x) \prod_{j=0 j\neq i}^{n} \frac{x-x_j}{x_i-x_j}$$

will be exact for all $f \in P_{2n+1}$.

Theorem

Let w be a positive weight function and let $q \in C[a, b]$ be nonzero and w-orthogonal to n. Then, q changes sign at least $n + 1$ times on (a, b).

Properties

1. In the Gaussian qudarture formula, the coefficients are positive and their sum is $\displaystyle\int_a^b w(x)dx$.
2. If f is continuous, then $\int_a^b f(x)w(x)dx \approx \sum_{i=0}^{n} A_i f(x_i)$ converges to the integral as $n \to \infty$.

Error Analysis for Gaussian Quadrature Rule

$$\int_a^b f(x)w(x)dx = \sum_{i=0}^{n} A_i f(x_i) + E$$

,where

$$E = \frac{f^{(2n+2)}(\xi)}{(2n)!} \int_a^b q^2(x)w(x)dx$$

$$, q(x) = \prod_{i=0}^{n}(x - x_i)$$

and $a < \xi < b$.

Gaussian Quadrature Table

Mathematicians already compute weights and nodes for Guassian quadrature rule in terms of Legendre integration. In other word,

$$\int_{-1}^{1} f(x)dx \approx A_i f(x_i)$$

$\{A_i, x_i\}$ is already known.
https://en.wikipedia.org/wiki/Gaussian_quadrature
Then, How to compute $\int_a^b f(x)dx$ by using the above table?
Change of variable can be applied.

2D Gauss Quadrature Rule

We should construct 2 dimension Gaussian Quadrautre, using 1 dimension 3 points Gaussian quadrature. Let's consider a following double integral,

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)dxdy.$$

Assume y is fixed and Let's apply 3 points Gaussian quadrature rule with respect to x. then,

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)dxdy \approx \int_{-1}^{1} \frac{5}{9}f(-\sqrt{\frac{3}{5}},y)+\frac{8}{9}f(0,y)+\frac{5}{9}f(\sqrt{\frac{3}{5}},y)dy$$

. Now, If we take 3 points Gaussian quadrature rule with respect to y, In results, we gain 2 dimension Gaussian quadrature rule like followings.

$$\int_{-1}^{1} \frac{5}{9} f(-\sqrt{\frac{3}{5}}, y) + \frac{8}{9} f(0, y) + \frac{5}{9} f(\sqrt{\frac{3}{5}}, y) dy \approx$$

$$\frac{5}{9}(\frac{5}{9} f(-\sqrt{\frac{3}{5}}, -\sqrt{\frac{3}{5}}) + \frac{8}{9} f(0, -\sqrt{\frac{3}{5}}) + \frac{5}{9} f(\sqrt{\frac{3}{5}}, -\sqrt{\frac{3}{5}}))$$

$$+\frac{8}{9}(\frac{5}{9} f(-\sqrt{\frac{3}{5}}, 0) + \frac{8}{9} f(0, 0) + \frac{5}{9} f(\sqrt{\frac{3}{5}}, 0))+$$

$$\frac{5}{9}(\frac{5}{9} f(-\sqrt{\frac{3}{5}}, \sqrt{\frac{3}{5}}) + \frac{8}{9} f(0, \sqrt{\frac{3}{5}}) + \frac{5}{9} f(\sqrt{\frac{3}{5}}, \sqrt{\frac{3}{5}})).$$

Lemma

Suppose $f(x,y) = g(x)h(y)$. Then, gauss quadrature rule is represented by

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)dxdy = \int_{-1}^{1}\int_{-1}^{1} g(x)h(y)dxdy \approx$$

$$[\frac{5}{9}g(-\sqrt{\frac{3}{5}}) + \frac{8}{9}g(0) + \frac{5}{9}g(\sqrt{\frac{3}{5}})] \times [\frac{5}{9}h(-\sqrt{\frac{3}{5}}) + \frac{8}{9}h(0) + \frac{5}{9}h(\sqrt{\frac{3}{5}})].$$

Now, suppose $f(x,y) = x^i y^j$, where $i, j \in \mathbb{N}$. Then,

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)dxdy = \int_{-1}^{1}\int_{-1}^{1} x^i y^j dxdy = \int_{-1}^{1} x^i dx \int_{-1}^{1} y^j dy.$$

By lemma and 1 dimensional Gaussian quadrature property, each terms, $\int_{-1}^{1} x^i dx, \int_{-1}^{1} y^j dy$, is exact up to 5th order at least.

Let assume $R(k,0)$ be Trapezoidal Rule with 2^k subinterval.

$$R(0,0) = \frac{1}{2}(b-a)[f(a)+f(b)]$$

$$R(n,0) = \frac{1}{2}R(n-1,0) + h\sum_{i=1}^{2^n-1} f(a+(2i-1)h)$$

Euler-Maclaurin formula

$$\int_0^1 f(t)dt = \frac{1}{2}[f(0)+f(1)] + \sum_{k=1}^{m-1} A_{2k}[f^{(2k-1)}(0) - f^{(2k-1)}(1)]$$

$$-A_{2m}f^{(2m)}(xi_0)$$

where $\xi_0 \in (0,1)$ and $k!A_k$ are Bernoulli numbers.

If we apply Euler-Maclaurin formula, we have a following equations.

$$I = R(n,0) + c_2h^2 + c_4h^4 + \cdots + c_{2m}h^{2m}f^{(2m)}(\xi)$$

where $f \in \mathcal{C}^{2m}[a,b]$ and $\xi \in (a,b)$. and coefficients c_i are independent on h. Now Let's apply Romberg algorithm.

$$I = R(n,0) + c_2h^2 + c_4h^4 + \cdots$$

$$I = R(n,0) + c_2(\frac{h}{2})^2 + c_4(\frac{h}{2})^4 + \cdots$$

This relation yields equation

$$I = R(n,0) + \frac{1}{3}(R(n,0) - R(n-1,0))$$

In general,

$$R(n,m) = R(n,m-1) + \frac{1}{4^m - 1}[R(n,m-1) - R(n-1,m-1)]$$

Error Estimator

1. Error $E(f)$ can be estimated to

$$E(f) \approx \frac{T(f) - M(f)}{3}.$$

2. Let $h = b - a$, $S^{(1)} = S(a,b)$ and $E^{(1)} = E(a,b)$.

$$I = S^{(1)} + E^{(1)}$$

Now, apply Simpson's rule on two subintervals.

$$I = S(a,c) + S(c,b) + E(a,c) + E(c,b) =: S^{(2)} + E^{(2)}$$

If we assume $f^{(4)}$ is constant, then

$$S^{(2)} - S^{(1)} = E^{(1)} - E^{(2)} = 15E^{(2)}$$

Therefore, $I = S^{(2)} + E^{(2)} = S^{(2)} + \frac{1}{5}(S^{(2)} - S^{(1)})$

Algorithm

1. Compute $S^{(1)}$ and $S^{(2)}$
2. If $|S^{(2)} - S^{(1)}| < 15\epsilon$, then $I = \frac{16}{15}S^{(2)} - \frac{1}{15}S^{(1)}$
3. else repeat the process on $[a, c]$ and $[c, b]$ with tolerance $\epsilon/2$.

Multi-D Integral

Example

Compute a following definite integrals.

1. $\displaystyle\iint_D x^2 y dx dy,\ D = [0,3] \times [1,2].$
2. $\displaystyle\iint_D 1 - x^2 - y^2 dA,\ D = \{(x,y) | x^2 + y^2 \leq 1\}.$

Improper Integration and Singular Integration

Example

Compute a following definite integrals.

1. $\displaystyle\int_0^\infty e^{-x^2} dx$
2. $\displaystyle\int_0^1 \frac{e^{-x}}{\sqrt{x}} dx$
3. $\displaystyle\int_0^1 \frac{\sin x}{x} dx$

Monte-Carlo Approach

Example

Use Monte Carlo approximation to Compute a following definite integrals.

$$\iint_D x^2 y dx dy$$

, $D = [0, 3] \times [1, 2]$.

Properties

1. Approximation error is dependent to $\frac{1}{\sqrt{n}}$. Furthermore, the Error is not dependent to dimension.
2. Monte Carlo simulation in high dimension computation is high cost computation.