Differentiation & Integration

IPCST
Seoul National University

Review of Taylor Series

- Taylor expansion about $x = c$,

$$
\begin{aligned}
f(x) &= f(c) + f'(c)(x-c) + \frac{1}{2!}f''(c)(x-c)^2 + \frac{1}{3!}f'''(c)(x-c)^3 + \cdots \\
&= \sum_{k=0}^{\infty} \frac{1}{k!} f^{(k)}(c)(x-c)^k.
\end{aligned}
$$

  - Maclaurin series ($c = 0$)

$$
f(x) = f(0) + f'(0)x + \frac{1}{2!}f''(0)x^2 + \frac{1}{3!}f'''(0)x^3 + \cdots = \sum_{k=0}^{\infty} \frac{1}{k!} f^{(k)}(0)x^k.
$$

Review of Taylor Series

- Error and convergence
  - Assume $f^{(k)}(x)$ $(0 \le k \le n)$ are smooth functions.
  - Partial sum: $f_n(x) = \sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(c)(x-c)^k$
  - **Taylor theorem**

$$E_{n+1} = f(x) - f_n(x) = \sum_{k=n+1}^{\infty} \frac{1}{k!} f^{(k)}(c)(x-c)^k = \frac{1}{(n+1)!} f^{(n+1)}(\xi)(x-c)^{n+1}$$

    - where ξ is some value between x and c.

Review of Taylor Series

- Geometric interpretation of the Taylor theorem in case of $n = 0$
  - Wen Shen example 1.8.

$$f(b) - f(a) = (b-a)f'(\xi),$$ for some ξ in (a,b)

$$f'(\xi) = \frac{f(b) - f(a)}{b - a}$$

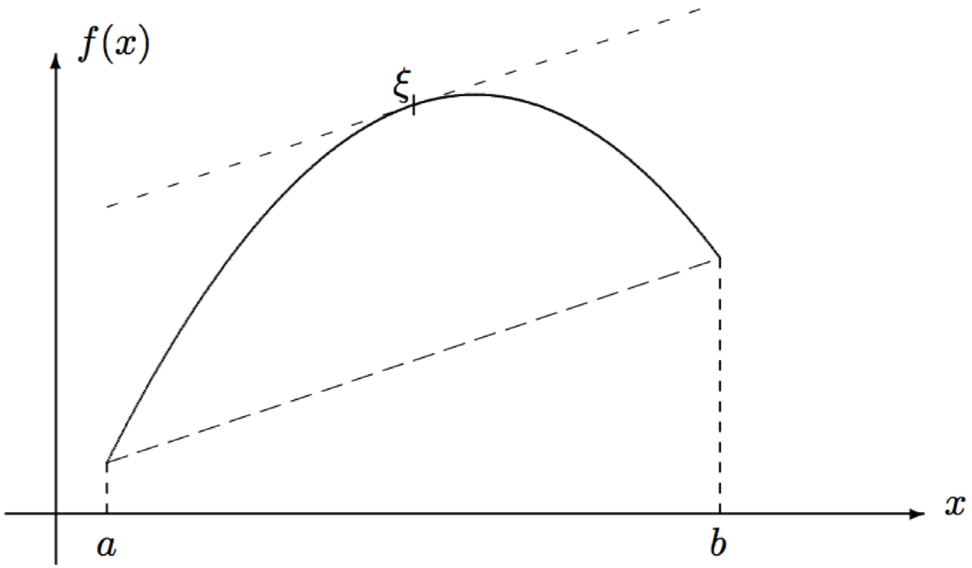


Figure from Wen Shen

Finite Differences

- 1st order derivative
  - **Forward** 2-point formula

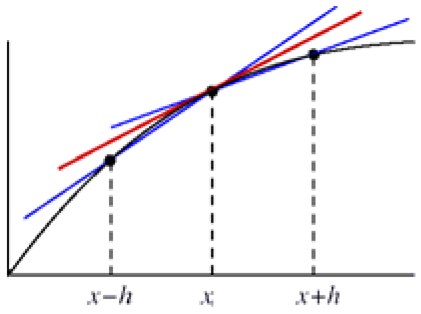
$$f'(x) = \frac{f(x+h) - f(x)}{h} + O(h)$$

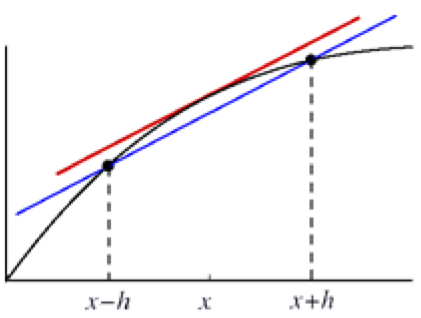
  - **Backward** 2-point formula

$$f'(x) = \frac{f(x) - f(x-h)}{h} + O(h)$$

  - **Central** 3-point formula

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





Finite Differences

- Local truncation errors

$$f(x+h) = f(x) + h\,f'(x) + \frac{h^2}{2} f''(x) + O\left(h^3\right)$$

$$f(x-h) = f(x) - h\,f'(x) + \frac{h^2}{2} f''(x) + O\left(h^3\right)$$

  - **Forward** 2-point formula

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

  - **Backward** 2-point formula

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

Finite Differences

- Local truncation errors

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + O(h^4)$$
$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2}f''(x) - \frac{h^3}{6}f'''(x) + O(h^4)$$

  - **Central** 3-point formula

$$\frac{f(x+h) - f(x-h)}{2h} = f'(x) + \frac{h^2}{6}f'''(x) + O(h^4)$$

Do It Yourself

- For $f(x) = e^x$, compute finite differences at $x = 0$ with $h = 0.1, 0.01$ and find their local truncation errors.

参考: Finite Differences

- 1st order derivative
  - **Forward** 3-point formula

$$f'(x) = \frac{-f(x+2h)+4f(x+h)-3f(x)}{2h} + O\left(h^2\right)$$

  - **Backward** 3-point formula

$$f'(x) = \frac{3f(x)-4f(x-h)+f(x-2h)}{2h} + O\left(h^2\right)$$

  - **Central** 5-point formula

$$f'(x) = \frac{-f(x+2h)+8f(x+h)-8f(x-h)+f(x-2h)}{12h} + O\left(h^4\right)$$

❖ Look for ‘Finite difference coefficients’

Finite Differences

- 2nd order derivative
  - **Forward** 3-point formula

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

  - **Backward** 3-point formula

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

  - **Central** 3-point formula

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

参考: Multivariate Finite Differences

- Partial derivate version of finite differences
  - First-order central 3-point formula

$$\frac{\partial f(x,y)}{\partial x} = \frac{f(x+h_x,y) - f(x-h_x,y)}{2h_x} + O(h^2)$$

  - Second-order central 3-point formula

$$\frac{\partial^2 f(x,y)}{\partial y^2} \approx \frac{f\left(x,y+h_y\right) - 2f(x,y) + f(x,y-h_y)}{h_y^2}$$

$$\frac{\partial^2 f(x,y)}{\partial x \partial y} \approx \frac{f\left(x+h_x,y+h_y\right) - f\left(x-h_x,y+h_y\right) - f\left(x+h_x,y-h_y\right) + f\left(x-h_x,y-h_y\right)}{4h_x h_y}$$

❖ The last formula can be reformulated as combination of other finite differences.

Numerical Integration

- a.k.a. (numerical) quadrature

- Finding $$I(f) = \int_a^b f(x)\,dx$$

  - accurately by this way
    - The interval $[a, b]$ → subintervals
    - Polynomial approximation on each subinterval
      - Interpolation
    - Integration on each subinterval → sum-up

❖ Also, there are some techniques to create a new formula of enhanced accuracy.

Numerical Integration

- Application cases
  - Some functions could be very hard to integrate analytically - no explicit expression of their anti-derivates
    - Ex.) the error function

  - Only discrete data set
    - Ex.) experimental measurement data

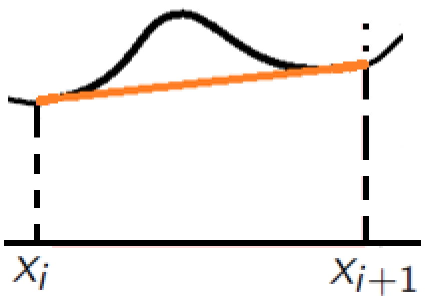
Trapezoid Rule

- ◆ Equivalent to integration of **linear splines**

- On each subinterval,

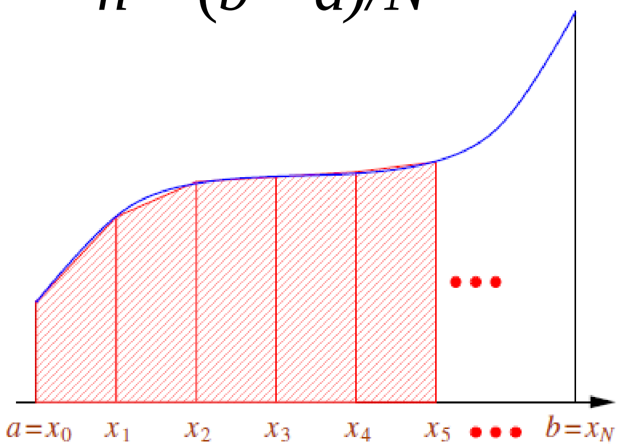
$$\int_{x_i}^{x_{i+1}} f(x)\,dx \approx \frac{h}{2}\left(f(x_{i+1}) + f(x_i)\right)$$

✓ $h = x_{i+1} - x_i$



- If equally spaced,

$$h = (b - a)/N$$



$$\frac{h}{2}[f_0 + 2f_1 + 2f_2 + \cdots + 2f_{N-1} + f_N]$$

Ex) $\int_0^1 x^2 \approx \frac{0.2}{2}[0 + 2\cdot 0.04 + 2\cdot 0.16$
$+2\cdot 0.36 + 2\cdot 0.64 + 1] = 0.34$

Trapezoid Rule

- Error estimates
  - Wen Shen p. 68
  - Considering the polynomial approximation

$$\int_{x_i}^{x_{i+1}} p_i(x)\,dx := \frac{h}{2}\left(f(x_{i+1}) + f(x_i)\right)$$

  - Enabling to define the error on subinterval

$$E_{T,i}(f;h) = \int_{x_i}^{x_{i+1}} [f(x) - p_i(x)]\,dx$$

  - Interpolation error theorem gives upper bound

$$E_{T,i}(f;h) \;=\; \frac{1}{2}f''(\xi_i)\int_{x_i}^{x_{i+1}}(x-x_i)(x-x_{i+1})\,dx \;=\; -\frac{1}{12}h^3 f''(\xi_i).$$

Trapezoid Rule

- Error estimates
  - Wen Shen p. 69
  - Total error

$$|E_T(f;h)| = \sum_{i=0}^{n-1} |E_{T,i}(f;h)| \le \sum_{i=0}^{n-1} \frac{M_i}{12} h^3 \le \frac{h^3}{12} nM = \frac{h^3}{12} \cdot \frac{b-a}{h} M .$$

- where $M_i \coloneqq \max_{\xi \in [x_i, x_{i+1}]} |f''(\xi)|$ & $M \coloneqq \max_{\xi \in [a,b]} |f''(\xi)|$

$$\therefore \ |E_T(f;h)| \le \frac{b-a}{12} h^2 \max_{x\in[a,b]} |f''(x)| .$$

- Ex) $\int_0^1 x^2 \approx 0.34$ for $h = 0.2$ → exact error = 1/150

$$|E_T(x^2; 0.2)| \le \frac{1}{12} 0.04 \cdot 2 = \frac{1}{150}$$

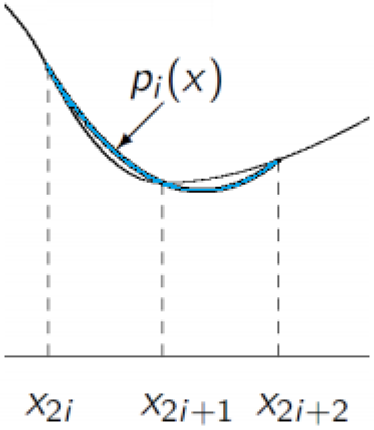
Do It Yourself

- Compute $\int_0^2 e^x dx$ (that is, $f(x) = e^x$) within the trapezoid rule and its error, increasing the number of nodes from 5 to 320. What is the minimum number of nodes to ensure an error $\leq 0.5 \times 10^{-4}$?
  - Wen Shen example 4.2

Simpson's Rule

◆ Equivalent to integration of **Lagrange polynomials**

- Subinterval: $[x_{2i}, x_{2i+2}]$
  - Assuming $x_{2i+2} - x_{2i+1} = x_{2i+1} - x_{2i} = h$
- Lagrange polynomial for the 3 pts $(x_{2i+2}, x_{2i+1}, x_{2i})$



$$p_i(x) = \frac{1}{2h^2} f(x_{2i})(x - x_{2i+1})(x - x_{2i+2}) - \frac{1}{h^2} f(x_{2i+1})(x - x_{2i})(x - x_{2i+2}) + \frac{1}{2h^2} f(x_{2i+2})(x - x_{2i})(x - x_{2i+1})$$

$$\int_{x_{2i}}^{x_{2i+2}} p_i(x)\, dx = \frac{h}{3}\left[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})\right].$$

Figure from Wen Shen

Simpson's Rule

- If equally spaced,
  - Wen Shen pp. 71~72

$$\begin{aligned}
\int_a^b f(x)\,dx &\approx S(f;h) \\
&\doteq \sum_{i=0}^{n-1} \int_{x_{2i}}^{x_{2i+2}} p_i(x)\,dx \\
&= \frac{h}{3} \sum_{i=0}^{n-1} \left[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})\right] \\
&= \frac{h}{3} \left[f(x_0) + 4\sum_{i=1}^{n} f(x_{2i-1}) + 2\sum_{i=1}^{n-1} f(x_{2i}) + f(x_{2n})\right].
\end{aligned}$$

Simpson's Rule

- Error estimates
  - Method 1 (Wen Shen p. 73)
    - Like in the trapezoid rule

$$|E_{S,i}(f;h)| = \left|\int_{x_{2i}}^{x_{2i+2}} [f(x) - p_i(x)]\,dx\right| \le \frac{h^5}{90} M_i\,, \qquad M_i = \max_{\xi\in[x_{2i},x_{2i+2}]} \left|f^{(4)}(\xi)\right|.$$

$$\Rightarrow \quad |E_S(f;h)| \le \frac{h^5}{90}\sum_{i=0}^{n-1} M_i \le \frac{h^5}{90}\, n \max_{\xi\in[a,b]}\left|f^{(4)}(\xi)\right| = \frac{b-a}{180} h^4 \max_{\xi\in[a,b]}\left|f^{(4)}(\xi)\right|.$$

  - Method 2
    - Halving subintervals again

$$|S(f,h) - S(f,h/2)| \le \frac{b-a}{180} h^4 \left[\max_{\xi\in[a,b]}\left|f^{(4)}(\xi)\right| - \frac{1}{16}\min_{\xi\in[a,b]}\left|f^{(4)}(\xi)\right|\right] \le \frac{15}{16}\frac{b-a}{180} h^4 \max_{\xi\in[a,b]}\left|f^{(4)}(\xi)\right|$$

$$\rightarrow \; |S(f,h) - S(f,h/2\,)| \approx \frac{15}{16}|E_S(f;h)| = 15|E_S(f;h/2)|$$

Simpson's Rule

- Error estimates
  - Ex.) $\int_0^2 e^x dx$, $h = 0.1$
    - Method 1

$$|E_S(f;h)| \leq \frac{2}{180} h^4 e^2 = 8.2 \times 10^{-6}$$

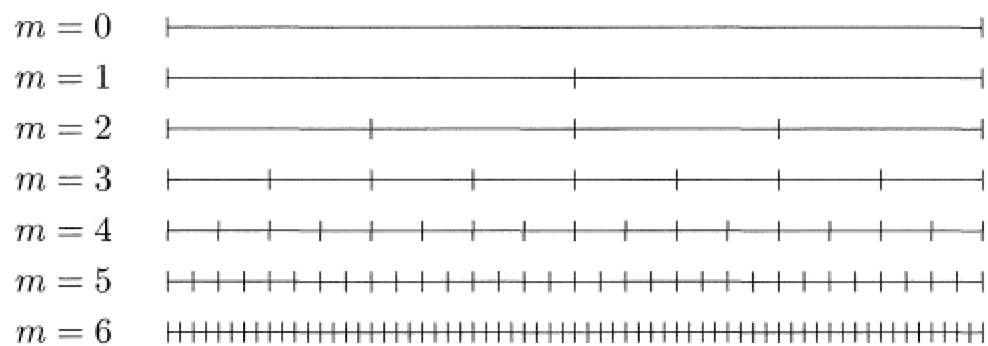
    - Method 2

$$|E_S(f;h/2)| \approx \frac{|S(f,h) - S(f,h/2)|}{15} = 2.2 \times 10^{-7}$$

Recursive Trapezoid Rule

– Wen Shen pp. 75 ~76

$$h_m = \frac{b-a}{2^m}, \qquad h_{m+1} = \frac{1}{2} h_m.$$



$$\begin{cases} T(f; h_m) = h_m \cdot \left[\frac{1}{2} f(a) + \frac{1}{2} f(b) + \sum_{i=1}^{2^m - 1} f(a + i h_m) \right], \\ T(f; h_{m+1}) = h_{m+1} \cdot \left[\frac{1}{2} f(a) + \frac{1}{2} f(b) + \sum_{i=1}^{2^{m+1} - 1} f(a + i h_{m+1}) \right]. \end{cases}$$

- ✓ Flexible
- ✓ More efficient than repetitive use of the trapezoid rule

$$T(f; h_{m+1}) = \frac{1}{2} T(f; h_m) + h_{m+1} \sum_{j=0}^{2^m - 1} f(a + (2j+1) h_{m+1}).$$

Richardson Extrapolation

- Error formula for trapezoid rule

$$E(f;h) = I(f) - T(f;h) = a_2 h^2 + a_4 h^4 + \cdots + a_n h^n$$

  - if f can be expanded into a Taylor series

$$E\left(f;\frac{h}{2}\right) = I(f) - T\left(f;\frac{h}{2}\right) = a_2\left(\frac{h}{2}\right)^2 + a_4\left(\frac{h}{2}\right)^4 + \cdots + a_n\left(\frac{h}{2}\right)^n$$

➢ Eliminating the 2nd order term,

$$(2^2-1)I(f) - 2^2 T\left(f;\frac{h}{2}\right) + T(f;h) = O(h^4)$$

$$\Rightarrow \quad U(h) = \frac{2^2 T(f;h/2\,) - T(f;h)}{2^2-1} = I(f) + O(h^4)$$

✓ Likewise, you can remove the 4th order term, the 6th order, …… → **Romberg Algorithm**

Romberg Algorithm

- Defining $R(i,k)$:
  - ◆ $R(i,0)$ = recursive trapezoid rule results

    $R(0,0) = T(f;L)$ where $L = b - a$

    $R(1,0) = T(f;L/2), R(2,0) = T(f;L/4), \dots \dots$

    $\dots \dots, R(n,0) = T(f;L/2^n)$
  - ◆ $k \neq 0$: $R(i,k)$ is defined by a recursion formula

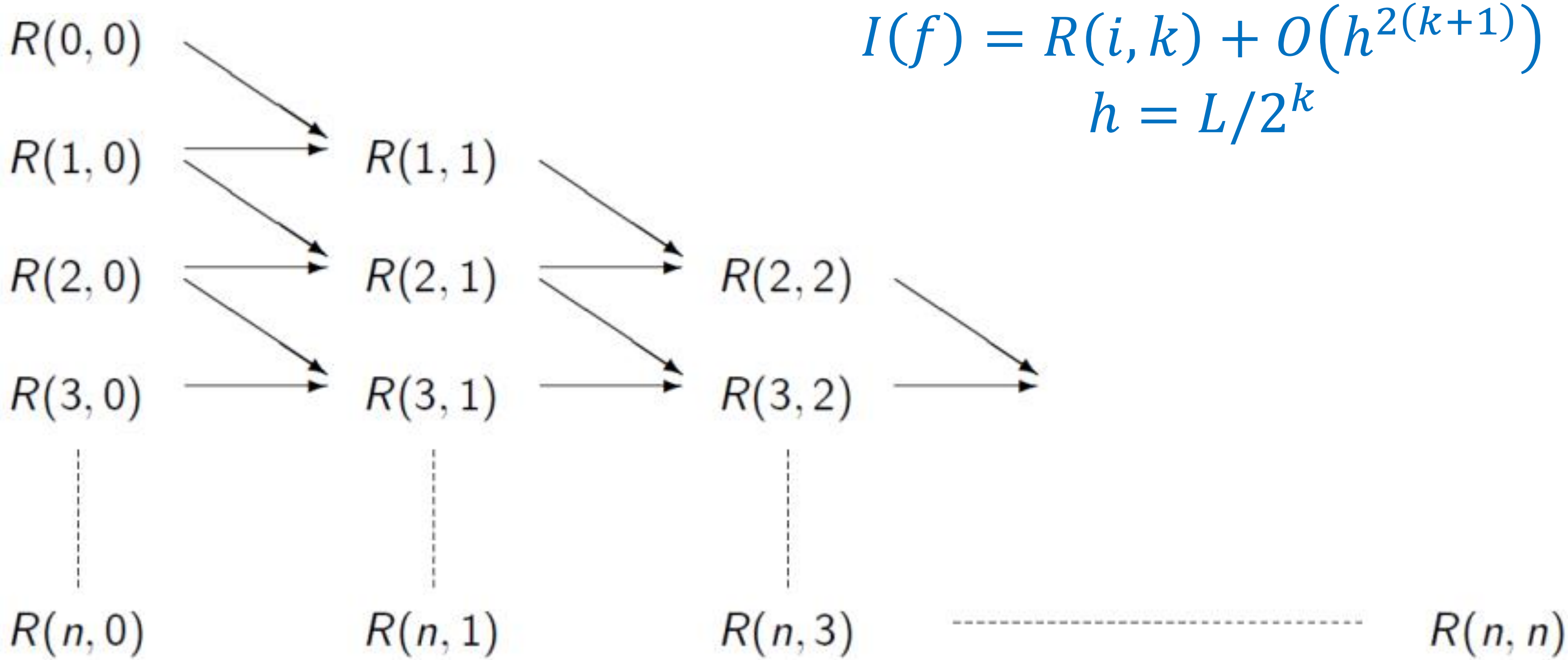
$$R(i,k) = R(i,k-1) + \frac{R(i,k-1) - R(i-1,k-1)}{2^{2k}-1}$$

❖ Cf.) $U(h) = T(f;h/2) + \frac{T(f;h/2) - T(f;h)}{2^2-1}$

Romberg Algorithm

- Romberg triangle

$$I(f) = R(i,k) + O\left(h^{2(k+1)}\right)$$
$$h = L/2^k$$



Romberg Algorithm

- Algorithm summary
  ① Initialization: set $h = L$ & compute $R(0,0)$
  ② Loop of i: compute $R(i,0)$ by using the recursive trapezoid rule formula

$$T(f;h_{m+1}) = \frac{1}{2}T(f;h_m) + h_{m+1}\sum_{j=0}^{2^m-1} f(a+(2j+1)h_{m+1}).$$

    - Set $h = h/2$ every iteration
    - Inner loop for the summation (from 0 to $2^m - 1$)

  ③ Loop of i & k: compute $R(i,k)$
    - Outer loop: k (from 1 to n)
    - Inner loop: i (from k to n)

Do It Yourself

- [After this class]: Romberg integration
  - Wen Shen pp. 87
  - Use Romberg algorithm to compute $\int_0^{\pi/2} \cos(2x)\, e^{-x} dx$ (= 0.2415759) and get the results below

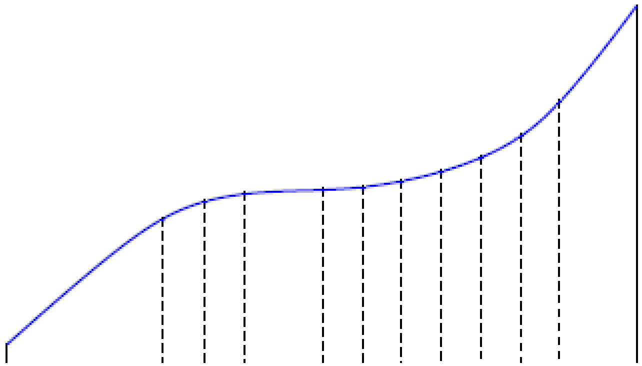
0.6221			
0.3111	0.2074		
0.2575	0.2397	0.2419	
0.2455	0.2415	0.2416	0.2416

Adaptive Quadrature

- The integrand varying rapidly → fine spacing
- The integrand varying slowly → coarse spacing

- **Algorithm**
  1. Integration for the given subinterval
  2. Error estimation
  3. If (error > tolerance)
     I. Halve the given subinterval
     II. Recursive calls for each subinterval (Repeat 1~3 for the new intervals)

Adaptive Quadrature



- rapidly → fine spacing
- slowly → coarse spacing

- **Algorithm**
  1. Integration for the given subinterval
  2. Error estimation
  3. If (error > tolerance)
     I. Halve the given subinterval
     II. Recursive calls for each subinterval (Repeat 1~3 for the new intervals)

Adaptive Simpson's Quadrature

- Use $|S_i(f,h) - S'_i(f,h/2\,)| \approx 15\left|E_{S',i}(f;h/2)\right|$ for each subinterval
  - where $S'_i(f,h/2\,) = S_{2i}(f,h/2) + S_{2i+1}(f,h/2)$
- Algorithm summary
  ① Compute $S(f,h)$
  ② Halve each subinterval and compute $S(f,h/2)$
  ③ Check $|S_i(f,h) - S'_i(f,h/2\,)| \leq 15 \cdot (\text{tolerance})$
    - Yes: stop halving
    - No: recursive calls

  - You can easily find example codes via internet.

Gaussian Quadrature

- So far, all numerical integration formulas have the form of

$$\int_a^b f(x) \approx A_1 f(x_0) + A_2 f(x_2) + \cdots + A_n f(x_n)$$

◆ Goal: Find the accurate integration values for polynomials by using the least non-uniform nodes.

❖ Some sets of polynomials have orthogonality and completeness.
  - Smooth functions in some interval can be approximated as a series of such polynomials.

Gaussian Quadrature

- Legendre polynomials
  - Interval: [-1,1]

    $P_0(x) = 1, P_1(x) = x, P_2(x) = \frac{1}{2}(3x^2 - 1), P_3(x) = \frac{1}{2}(5x^3 - 3x)$
  - Degree $m = 1$: $f(x) = a_0 P_0(x) + a_1 P_1(x)$

    $$\int_{-1}^{1} f(x) = 2a_0 = 2f(0)$$
  - Degree $m = 2$: $f(x) = a_0 P_0(x) + a_1 P_1(x) + a_2 P_2(x)$
    - One node is insufficient. Add one more.

    $$\int_{-1}^{1} f(x) = 2a_0 = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$$

Gaussian Quadrature

- Legendre polynomials
  - Interval: [-1,1]

    $P_0(x) = 1, P_1(x) = x, P_2(x) = \frac{1}{2}(3x^2 - 1), P_3(x)$
    $= \frac{1}{2}(5x^3 - 3x)$
  - Degree $m = 3$: $f(x) = a_0P_0(x) + a_1P_1(x) + a_2P_2(x) + a_3P_3(x)$

    $$\int_{-1}^{1} f(x) = 2a_0 = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$$
  - Thanks to the properties of Legendre polynomials, we need only a few nodes.

Gaussian Quadrature

- Legendre polynomials
  - Interval: [-1,1]

  $P_0(x) = 1, P_1(x) = x, P_2(x) = \frac{1}{2}(3x^2 - 1), P_3(x) = \frac{1}{2}(5x^3 - 3x),\ P_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3),\ P_5(x) = \frac{1}{8}(63x^5 - 70x^3 + 15x)$

  - Degree $m = 4$: $f(x) = a_0P_0(x) + a_1P_1(x) + a_2P_2(x) + a_3P_3(x) + a_4P_4(x)$
    - Two nodes are insufficient. Add one more.

$$\int_{-1}^{1} f(x) = 2a_0 = \frac{5}{9} f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9} f(0) + \frac{5}{9} f\left(\sqrt{\frac{3}{5}}\right)$$

Gaussian Quadrature

- Legendre polynomials
  - Interval: [-1,1]

  $P_0(x) = 1, P_1(x) = x, P_2(x) = \frac{1}{2}(3x^2 - 1), P_3(x) = \frac{1}{2}(5x^3 - 3x),\ P_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3),\ P_5(x) = \frac{1}{8}(63x^5 - 70x^3 + 15x)$
  - Degree $m = 5$: $f(x) = a_0P_0(x) + a_1P_1(x) + a_2P_2(x) + a_3P_3(x) + a_4P_4(x) + a_5P_5(x)$

$$\int_{-1}^{1} f(x) = 2a_0 = \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right)$$

Gaussian Quadrature

- Legendre polynomials
  - We need n nodes for a polynomial of degree $m = 2n - 1$.
    - Without using the properties of Legendre polynomials, we can prove it by using $\int_a^b f(x) \approx A_1 f(x_0) + A_2 f(x_2) + \cdots + A_n f(x_n)$.

    See Wen Shen p.83.

  - Node points x_i: roots of P_n

參考: Gaussian Quadrature

- Legendre polynomials

Weights $A_i = \frac{2}{(1-x_i^2)[P_n'(x_i)]^2}$

#points	Nodes x_i	Weights A_i
1	0	2
2	$\pm\sqrt{1/3}$	1
3	$\pm\sqrt{0.6}$ 0	$5/9$ $8/9$
4	$\pm\sqrt{(3-\sqrt{4.8})/7}$ $\pm\sqrt{(3+\sqrt{4.8})/7}$	$(18+\sqrt{30})/36$ $(18-\sqrt{30})/36$
5	$\pm\sqrt{(5-\sqrt{40/7})/9}$ 0 $\pm\sqrt{(5+\sqrt{40/7})/9}$	$(322+13\sqrt{70})/900$ $128/225$ $(322-13\sqrt{70})/900$

Gaussian Quadrature

- Legendre polynomials
  - Use this transformation for general interval $[a, b]$

$$t = \frac{1}{2}(b-a)x + \frac{1}{2}(a+b)$$

where $x \in [-1,1]$ & $t \in [a, b]$. Weights should be

$$A_i \rightarrow \overline{A_i} = \frac{b-a}{2} A_i$$

  - Inverse transformation

$$x = \frac{2t-(a+b)}{b-a}$$

Gaussian Quadrature

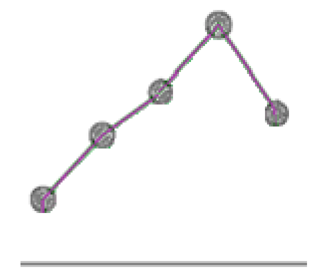
- Advantages
  - Higher order accuracy with small number of nodes
  - These formulas can handle integrands singular at the end of intervals.

Do It Yourself

- Use Gaussian quadrature to get approximate values of $\int_0^1 1/\sqrt{x}\, dx$.
  - Starting from 1 node, increase the node number up to 3 nodes (or 5 nodes if you can).

参考: Integrating Discrete Data

- (x_k, y_k), $k = 1,\ldots,n$ $\qquad (x_k < x_{k+1})$
- **Trapezoid rule**



$$T = \sum_{k=1}^{n-1} h_k \frac{y_{k+1} + y_k}{2}$$

$h_k = x_{k+1} - x_k$

Equivalent to '(piecewise) linear (spline) interpolation' and integration

✓ Other methods (Simpson rule, Romberg algorithm, ……) cannot be applied if only discrete data are given.

✓ You can give higher order corrections, but they do not guarantee higher accuracy.

References

- Wen Shen,
  “An Introduction to Numerical Computation”

- Wikipedia

- C. Moler,
  “Numerical Computing with MATLAB”