Quadrature $\equiv$ Numerical Integration

Approximate a definite integral $I(f) = \int_a^b f(x)\,dx$

Integration is "continuous summation", $I(f)$ = total amount of f in $[a,b]$,

Average of f on $[a,b]$ $= \frac{1}{b-a}\int_a^b f(x)dx$

In Calculus we learn to compute $I(f)$ via the Fundamental Thm of Calculus:

$\int_a^b f(x)dx = F(b) - F(a)$ where F is an antiderivative of f: $F' = f$

so it relies on figuring out F (via tricks...). Only certain elementary integrals can be evaluated this way, most cannot, because finding antiderivatives is non-trivial and may not even exist:

e.g. $\int e^{-x^2}dx$ is not an elementary function! $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-s^2}ds$ = area under bell curve

However, $\int_a^b f(x)dx$ = area under the graph of $y=f(x)$ over $[a,b]$

which suggests various methods, and definition of Riemann integral

Basic approach (for Newton-Cotes rules): Given f on $[a,b]$ (by formula or discrete values)*

1. discretize $[a,b]$ into M subintervals: $a=x_0, x_1, \dots, x_i, \dots, x_M = b$
2. approximate $f(x)$ by an interpolating polynomial on $[x_i, x_{i+1}]$, $P_N(x)$
3. integrate the polynomial $P_N(x)$ exactly (by Calculus)
4. add up the pieces

Basic fact: Quadrature is a well-conditioned process, so can be done accurately in contrast to num. differentiation!

	exact	numerical
differentiation	easy (thanks to Chain Rule)	hard (ill-conditioned)
integration	hard / impossible	easy (well-conditioned)

General form of Quadrature Rules: $I \approx I_N(f) = \sum_{i=0}^{N} w_i f(x_i)$, x_i nodes, w_i weights

Characterizations: order of accuracy is k if $|I(f) - I_N(f)| \leq C \cdot h^k$ as $h \to 0$ $h = \Delta x$

error $= \mathcal{O}(h^k)$

precision is p if rule is exact for polynomials of degree $\leq p$

i.e. if $I_N(x^n) = I(x^n)$ for $n = 0, 1, \ldots, p$ but not for $p+1$

Most important quadrature rules:

A. Newton-Cotes type rules: ~~predetermined~~/specified nodes $\{x_i\}$
weights chosen for certain precision

Rectangle Rule, Trapezoidal Rule, Simpson Rule, Romberg

A'. Open Newton-Cotes rules: versions that avoid evaluation at end points a, b

Midpoint Rule, ...

B. Adaptive methods: choose nodes for max accuracy

C. Gaussian rules: choose both node and weights for max precision

Gauss-Legendre, Gauss-Chebyshev, Gauss-Laguerre, Gauss-Hermite, Gauss-Kronrod

Basic Quadrature schemes for $I(f)=\int_a^b f(x)\,dx \approx Q_N(f)=\sum_{i=0}^{N} w_i f(x_i)$

Newton-Cotes type: we choos nodes x_i, rule specifies the weights w_i.

Knowing $f(x)$ at the nodes $a=x_0<x_1<\dots<x_i<\dots<x_N=b$ (N subintervals)

1. interpolate $f(x)$ over $[x_i, x_{i+k}]$ by an interp. polynomial $P_k(x)$ for some k
2. integrate $P_k(x)$ exactly to obtain quadrature rule $Q_N(f)$

so $$\int_{x_i}^{x_{i+k}} f(x)\,dx = \underbrace{\int_{x_i}^{x_{i+k}} P_k(x)\,dx}_{\text{quad. rule}} + \underbrace{\int_{x_i}^{x_{i+k}} \frac{f^{(k+1)}(\xi)}{(k+1)!}\prod_{j=0}^{k}(x-x_j)\,dx}_{\text{error}} = Q(f) + \text{local error}$$

3. Composite rule: add them up:

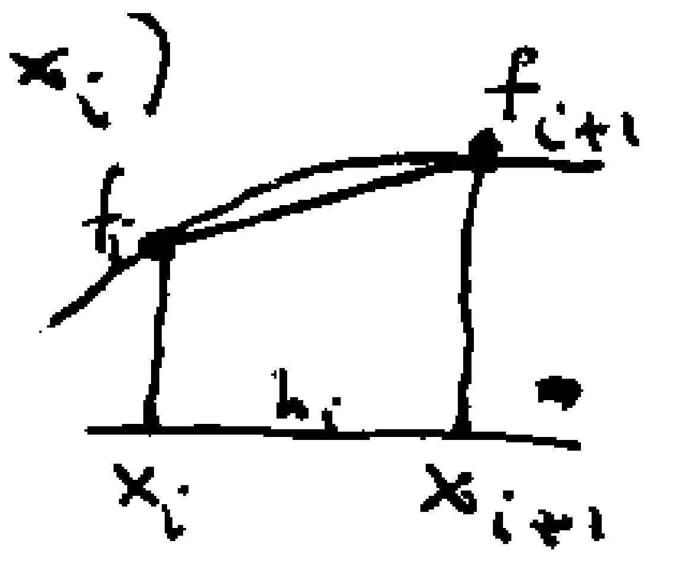
$$I(f) \approx Q_N(f) \quad \text{with error } \mathcal{O}(h^k)$$

Basic Rules:

1. Rectangle Rule: piecewise const. interpolant: $\int_{x_i}^{x_{i+1}} f(x)\,dx \approx f_i\cdot\Delta x_i$, $\text{local error} = \mathcal{O}(h^2)$

composite: $I(f) \approx \sum_{i=0}^{N-1} f_i\cdot h_i$, error $=\mathcal{O}(h)$; 1st order accurate, precision 0

2. Trapezoidal Rule: piecewise linear interpolant $P_1(x) = f_i + \frac{f_{i+1}-f_i}{x_{i+1}-x_i}(x-x_i)$



$$\Rightarrow \int_{x_i}^{x_{i+1}} f(x)\,dx \approx \int_{x_i}^{x_{i+1}} P_1(x)\,dx = f_i\cdot h_i + \frac{f_{i+1}-f_i}{h_i}\,\frac{(x-x_i)^2}{2}\Big|_{x_i}^{x_{i+1}}$$

$$= f_i\cdot h_i + \frac{f_{i+1}-f_i}{2}\cdot h_i = \frac{h_i}{2}\cdot(f_i+f_{i+1}) = \text{area of trapezoid}$$

$$\text{local error} = \int_{x_i}^{x_{i+1}} \frac{f''(\xi)}{2!}(x-x_i)(x-x_{i+1})\,dx \underset{=}{\overset{MVT}{=}} \frac{f''(\eta_i)}{2}\int_{x_i}^{x_{i+1}}(x-x_i)(x-x_{i+1})\,dx$$

$$= -\frac{f''(\eta_i)}{12}\cdot h_i^3$$

Composite: $$I(f) = \int_a^b f(x)\,dx = \sum_{i=0}^{N-1}\int_{x_i}^{x_{i+1}} f(x)\,dx \approx \sum_{i=0}^{N-1}\frac{h_i}{2}(f_i+f_{i+1}) = T_N(f)$$

For equispaced nodes: $h_i \equiv h = \Delta x = \frac{b-a}{N}$, $x_i = a + i\cdot h$, $i=0:N$

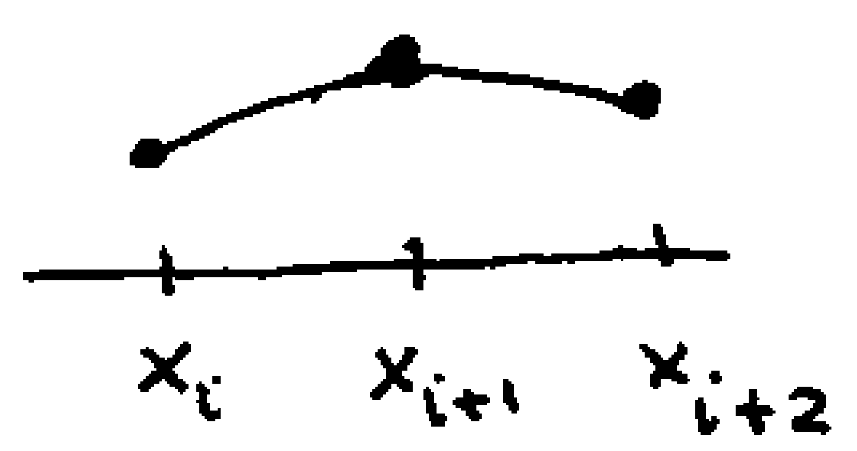
$$T_N(f) := \frac{h}{2}\Big[f(a) + 2\cdot\sum_{i=1}^{N-1} f_i + f(b)\Big]$$

$$|\text{error}| \le \frac{\|f''\|_\infty}{12}\cdot h^3\cdot N = \frac{\|f''\|_\infty}{12}(b-a)\cdot h^2 = \mathcal{O}(h^2)$$

Trapezoidal Rule is 2nd order accurate and of precision 1 (exact for 1st deg polys)

Extremely useful. Not very accurate on polynomials, but "exponentially" accurate on trig. polynomials! (on periodic functions)

3. <u>Simpon Rule</u>: quadratic interpolant over



$\int_{x_i}^{x_{i+2}} f(x)dx \approx \int_{x_i}^{x_{i+2}} P_2(x)dx$ with local error $\int_{x_i}^{x_{i+2}}$ (error of P_2 interpolation) dx

For equispaced nodes: $S_2(f) = \frac{h}{3}\left[f_i + 4f_{i+1} + f_{i+2}\right]$

expected to be exact for quadratics, but a miracle happens and turns out to be exact also for cubics! $S_2(x^3) \equiv I(x^3)$

so Simpson has precision 3 (instead of the expected precision 2)

so we could use cubic interpolant and thus obtain local error $O(h^5)$
global error $O(h^4)$

<u>Composite Simpson</u> Rule for even number of subintervals $N = 2M$

$$I(f) \approx \frac{h}{3}\left[f_0 + 4f_1 + 2f_2 + 4f_3 + \dots + 4f_{N-1} + f_N\right]$$

$$\text{with } |\text{error}| \leq \frac{\|f^{(4)}\|_\infty}{180} \cdot (b-a) \cdot h^4 = O(h^4)$$

Simpson Rule has precision 3, order 4 (one better than expected)
so it achieve better accuracy with smaller N.
provided f is C^4-smooth (degrades if not).
But it requires N = even, inconvenient some times...

Example (Epperson p.260): $I = \int_0^1 e^x dx$, for |error| $\leq 10^{-6}$
Trapezoidal needs $h \approx 0.002 \Rightarrow N \approx 500$ subintervals
but Simpson needs $h \approx 0.09 \Rightarrow N \approx 12$