

# Numerical Quadrature

---

- Consider the integral:

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

- The evaluation of the integration can be approximated through a weighted sum

$$I \approx \sum_{i=1}^N \omega_i f(x_i)$$

- Where:
  - $x_i$  are the *abscissa* points
  - $\omega_i$  are weights
- Assuming that  $f$  is finite and continuous on the interval  $[a,b]$  (not necessarily inclusive of the endpoints), the numerical quadrature can lead to a unique solution
- For positive weights, the operator is well conditioned and inherently stable
- The ultimate question is how to choose the abscissa points and weights to realize the desired error with the smallest  $N$  for a given function  $f$ .

## Types of Numerical Integration

---

- In general a quadrature integration rule works through *interpolation*
  - That is, the function is assumed to be interpolated to a given order of accuracy by a class of functions
  - The closed form integral of the known interpolation functions allow us to pre-calculate the weights
- The properties of the function can determine the type of quadrature rule that is used
  - Smooth, continuous function
    - Gauss-Legendre
    - Romberg
  - Smooth, periodic function
    - Trapezoidal
  - Singular function
    - Specialized Gauss-Quadrature

## Mid-Point Rule

- Mid-point rule

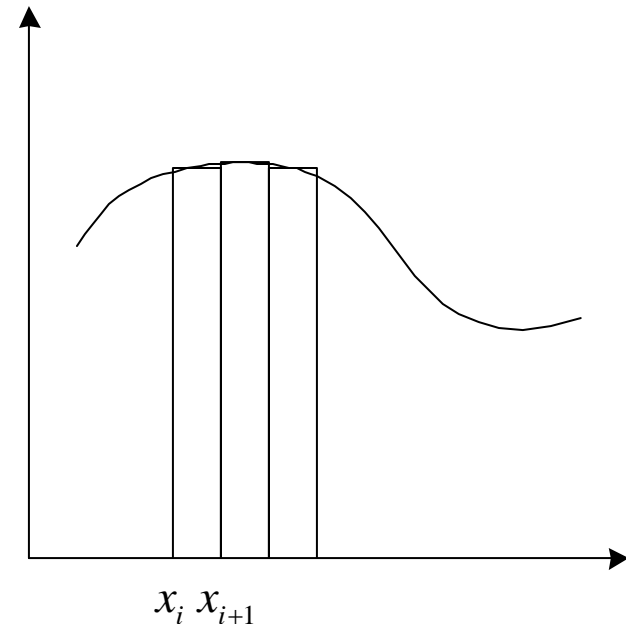
- Interpolate function via a “pulse” function with amplitude  $f((a+b)/2)$

$$I = \int_a^b f(x) dx \approx (b-a) f\left(\frac{a+b}{2}\right)$$

$$\therefore N = 1, \omega_1 = (b-a), x_1 = \frac{a+b}{2}$$

- Can break up into  $M$  sub-intervals:

$$I = \int_a^b f(x) dx \approx \sum_{i=1}^M (x_{i+1} - x_i) f\left(\frac{x_{i+1} + x_i}{2}\right)$$



- The error converges at least as  $O(h)$ , where  $h$  is the width of the sub-interval
  - (note: it will be proven later that the mid-point rule actually converges as  $O(h^2)$  when the function is sampled at the center of the interval. If sampled off-center, it converges as  $O(h)$ .)

## Trapezoidal Rule

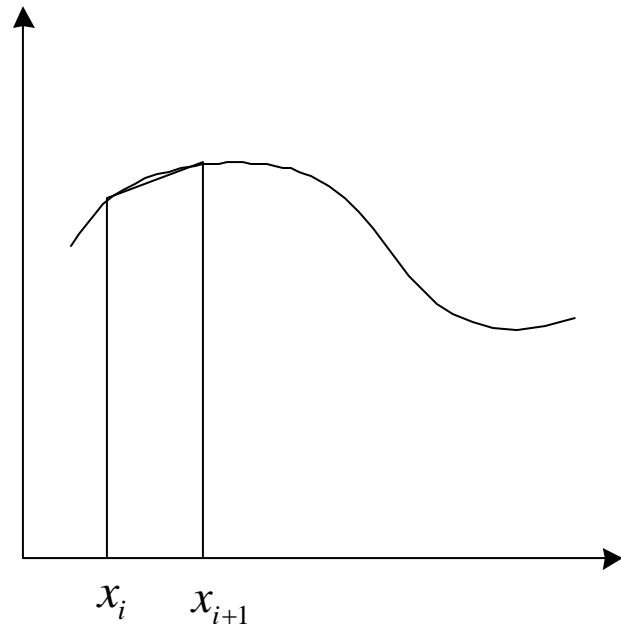
- Trapezoidal rule
  - Interpolate function via a piecewise linear function

$$I = \int_a^b f(x) dx \approx \left( \frac{b-a}{2} \right) (f(a) + f(b))$$

$$\therefore N = 2, \omega_{1,2} = \left( \frac{b-a}{2} \right), x_1 = a, x_2 = b$$

- Can break up into  $M$  sub-intervals:

$$I = \int_a^b f(x) dx \approx \sum_{i=1}^M \left( \frac{x_{i+1} - x_i}{2} \right) (f(x_i) + f(x_{i+1}))$$



- The error converges as  $O(h^2)$ , where  $h^2$  is the width of the sub-interval

## Interpolatory Quadrature Rules

---

- Interpolatory Quadrature Rules (also known as Newton-Cotes) are a set of rules for which the weights are calculated such that an  $N$ -point rule integrates exactly polynomials up to order  $N-1$
- The mid-point rule and trapezoidal rule fall under the class of quadrature methods based on polynomial interpolation with equal-spaced sampling and closed intervals (that is  $x_1 = a, x_N = b$ )
- The “degree” of the rule is defined as the order  $p$  of the polynomial that the quadrature rule integrates *exactly*
  - mid-point rule:  $p = 0$  (due to error cancellation,  $p = 1$  for mid-point rule)
  - Trapezoidal rule:  $p = 1$
- Example:

$$I = \int_a^b f(x) dx \approx \sum_{i=1}^N \omega_i f(x_i)$$

$$N = 1: \text{choose } \{f(x) = 1; x = a, b\}, x_1 = \frac{a+b}{2}$$

$$\omega_1 \cdot 1 = \int_a^b 1 dx = (b-a) \rightarrow \omega_i = (b-a)$$

$$N = 2: \{f_1(x) = 1, f_2(x) = x\}, x_1 = a, x_2 = b$$

$$\begin{pmatrix} 1 & 1 \\ a & b \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix} = \begin{pmatrix} \int_a^b 1 dx \\ \int_a^b x dx \end{pmatrix} = \begin{pmatrix} b-a \\ (b^2 - a^2)/2 \end{pmatrix}$$

$$\begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix} = \frac{1}{b-a} \begin{pmatrix} b & -1 \\ -a & 1 \end{pmatrix} \begin{pmatrix} b-a \\ (b^2 - a^2)/2 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} b-a \\ b-a \end{pmatrix}$$

## Normalization

---

- For generality, it is more desirable to express the quadrature rule for a unit range (0,1) rather than a specific range (a,b). The specific range can then be treated through a scaling of the result.

– Example

$$\int_0^1 f(x) dx \approx \sum_{i=1}^N \omega_i f(x_i), \quad 0 \leq x_i \leq 1$$

$$N = 2: \{f_1(x) = 1, f_2(x) = x\}, x_1 = 0, x_2 = 1$$

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix} = \begin{pmatrix} \int_0^1 1 dx \\ \int_0^1 x dx \end{pmatrix} = \begin{pmatrix} 1 \\ 1/2 \end{pmatrix}$$

$$\begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1/2 \end{pmatrix} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$$

– Then:

$$\int_a^b g(y) dy = \int_0^1 g((b-a)x + a) \cdot (b-a) dx \approx (b-a) \sum_{i=1}^N \omega_i g((b-a)x_i + a) = (b-a) \sum_{i=1}^N \omega_i g(y_i)$$

## Example

- Integrate:

$$\int_0^1 y^3 dy = \sum_{m=1}^{M-1} \int_{y_m}^{y_{m+1}} y^3 dy \approx \sum_{m=1}^{M-1} (y_{m+1} - y_m) \sum_{i=1}^N \omega_i f_i, \text{ where } f_i = y_i^3 = \left( (y_{m+1} - y_m) x_i + y_m \right)^3$$

– Mid-point Rule (N=1):

M	I	Rel. Error
1	0.125	0.5
10	0.2487501	0.00499
20	0.2496876	0.00125
50	0.2499498	0.0002
100	0.2499871	0.000052

Trapezoidal Rule (N=2):

M	I	Rel. Error
1	0.5000000	1.00000
10	0.2525001	0.01
20	0.2506251	0.0025
50	0.2500998	0.000399
100	0.2500246	0.000098

– Both the mid-point rule and the trapezoidal rule converge with  $O(h^2)$

## Example

- Integrate:

$$\int_0^1 y^{\frac{1}{2}} dy = \sum_{m=1}^{M-1} \int_{y_m}^{y_{m+1}} y^{\frac{1}{2}} dy \approx \sum_{m=1}^{M-1} (y_{m+1} - y_m) \sum_{i=1}^N \omega_i f_i, \text{ where } f_i = y_i^{\frac{1}{2}} = \left( (y_{m+1} - y_m)x_i + y_m \right)^{\frac{1}{2}}$$

– Mid-point Rule (N=1):

M	I	Rel. Error
1	0.7071068	.06066
10	0.6683839	0.002578
20	0.6672955	0.000943
50	0.6668305	.0002458
100	0.6667253	0.0000879

Trapezoidal Rule (N=2):

M	I	Rel. Error
1	0.5	0.25
10	0.6605093	0.00924
20	0.6644467	0.00333
50	0.6660952	0.000857
100	0.6664629	0.000306

## Higher-Order Integration Rules

- We can extend the previous results to higher-order

– Assume uniform sampling and open intervals  $\int_0^1 f(x) dx \approx \sum_{i=1}^N \omega_i f(x_i), \quad 0 \leq x_i \leq 1$

$$N = 3: \{f_1(x) = 1, f_2(x) = x, f_3(x) = x^2\}$$

$$\left\{ x_1 = 0, x_2 = \frac{1}{2}, x_3 = 1 \right\}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1/2 & 1 \\ 0 & 1/4 & 1 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} = \begin{pmatrix} \int_0^1 1 dx \\ \int_0^1 x dx \\ \int_0^1 x^2 dx \end{pmatrix} = \begin{pmatrix} 1 \\ 1/2 \\ 1/3 \end{pmatrix}$$

$$\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ \frac{2}{3} \\ \frac{1}{6} \end{pmatrix}$$

$$N = 4: \{f_1(x) = 1, f_2(x) = x, f_3(x) = x^2, f_4(x) = x^3\}$$

$$\left\{ x_1 = 0, x_2 = \frac{1}{3}, x_3 = \frac{2}{3}, x_4 = 1 \right\}$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1/3 & 2/3 & 1 \\ 0 & 1/9 & 4/9 & 1 \\ 0 & 1/27 & 8/27 & 1 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{pmatrix} = \begin{pmatrix} \int_0^1 1 dx \\ \int_0^1 x dx \\ \int_0^1 x^2 dx \\ \int_0^1 x^3 dx \end{pmatrix} = \begin{pmatrix} 1 \\ 1/2 \\ 1/3 \\ 1/4 \end{pmatrix}$$

$$\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{pmatrix} = \begin{pmatrix} 1/8 \\ 3/8 \\ 3/8 \\ 1/8 \end{pmatrix}$$

## Higher-Order Integration Rules

---

- We can extend the previous results to arbitrary order

$$\int_0^1 f(x) dx \approx \sum_{i=1}^N \omega_i f(x_i), \quad 0 \leq x_i \leq 1$$

$$\{f_k(x) = x^{k-1}, x_k = (k-1)/(N-1)\}$$

$$\underbrace{\begin{pmatrix} 1 & 1 & \cdots & 1 & \cdots & 1 \\ 0 & \frac{1}{N-1} & \cdots & \frac{i-1}{N-1} & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \left(\frac{1}{N-1}\right)^{i-1} & \cdots & \left(\frac{i-1}{N-1}\right)^{i-1} & \cdots & 1 \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ 0 & \left(\frac{1}{N-1}\right)^{N-1} & \cdots & \left(\frac{i-1}{N-1}\right)^{N-1} & \cdots & 1 \end{pmatrix}}_{\text{Vandermonde Matrix}} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_i \\ \vdots \\ \omega_N \end{pmatrix} = \begin{pmatrix} \int_0^1 1 dx \\ \int_0^1 x dx \\ \vdots \\ \int_0^1 x^{i-1} dx \\ \vdots \\ \int_0^1 x^{N-1} dx \end{pmatrix}$$

- The Vandermonde matrix becomes very ill-conditioned for large  $N$ .
  - Best used with closed form solutions

## Example of Higher-Order Rules

---

- Higher-order rules will converge more rapidly. N=3 is third order (form of Simpson's Rule), N = 4 is fourth order.
- Integrate:

$$\int_0^1 y^5 dy = \sum_{m=1}^{M-1} \int_{y_m}^{y_{m+1}} y^5 dy \approx \sum_{m=1}^{M-1} (y_{m+1} - y_m) \sum_{i=1}^N \omega_i f_i, \text{ where } f_i = y_i^5$$

– Simpson's Rule (N=3):

M	I	Rel. Error
1	0.18750000	0.125
2	0.16796875	0.0078
4	0.16674805	0.00049
8	0.16667175	0.0000305
10	0.16666875	0.0000125

Fourth-Order Rule (N=4):

M	I	Rel. Error
1	0.17592593	0.0556
2	0.16724537	0.00347
4	0.16670284	0.000217
8	0.16666893	0.0000135
10	0.16666759	0.0000056

– Convergence is  $O(h^4)$  and  $O(h^4)$ , respectively

# Error

---

- Degree

- The numerical quadrature rule is of degree  $d$  if it integrates exactly every polynomial up to order  $d$ , and approximately for polynomials of order  $> d$
- We have seen that an  $N$ -point interpolatory quadrature rule is at least of degree  $N-1$ 
  - Exception was the mid-point rule

- Error Bound:

$$\text{Error} \leq \begin{cases} \frac{1}{4} h^{N+1} \|f^{(N+1)}\|_{\infty} & - N - \text{odd} \\ \frac{1}{4} h^N \|f^{(N)}\|_{\infty} & - N - \text{even} \end{cases}$$

- where

$$h = \max \{ x_{m+1} - x_m; m = 1, \dots, M \}$$

- Thus, for a smooth function  $f$ , increasing  $N$  (fixed  $M$ ) reduces the error exponentially ( $h^N$ ), and increasing  $M$  (fixed  $N$ ) reduces the error geometrically.
- Even and Odd interpolatory quadrature rules converge with the same order. However, the even will typically provide a more accurate result when integrating the same interval

# Stability

---

- Stability

- The absolute condition number of the numerical quadrature rule is obtained by the sum of the magnitudes of the weights:

$$\sigma = \sum_{i=1}^N |\omega_i|$$

- Note that in order to integrate a constant exactly:

$$\sum_{i=1}^N \omega_i \equiv 1$$

- Therefore, if the weights are all positive, the condition # = 1.
- If any weights are negative, the absolute condition number can be > 1 and the scheme can be unstable. That is, due to rounding errors, the solution can diverge for large values of  $N$  or  $M$ .

# Accuracy

---

- Error Control
  - Typically when calculating an integral numerically, we need to evaluate it to a desired precision
  - If the integral is computed beyond this precision, we've exhausted unnecessary computational time
  - If integral is computed to less than the desired precision, undesirable error results
  - It is desirable to automate the estimation of the error and to adaptively refine the number of sub-intervals or the order of integration
- Calculating Error
  - Since the exact solution is not known, the error must be estimated based on the current *best* estimate for the solution
    - Have to be careful since estimates can appear to be near a converged solution if the intervals are drastically over sampled
- Refinement
  - When refining for a more accurate answer, can we re-use information to estimate a better solution using a technique known as *Richardson Extrapolation*

## Richardson Extrapolation

---

- Richardson Extrapolation is a very useful technique that allows one to refine the discretization and cancel out errors to effectively obtain a higher-order result

- Allows for adaptivity
- Let  $F(h)$  be the integral evaluated with uniform spacing  $h$
- We can estimate that:

$$F(h) = a_0 + a_1 h^p + O(h^r) \quad (r > p)$$

- Where,  $p$  and  $r$  are known based on error bounds, but  $a_0$  and  $a_1$  are not
- Next, assume that we now evaluate  $F(h/n)$  using the same quadrature rule, but using smaller intervals

$$F(h/n) = a_0 + a_1 n^{-p} h^p + O(h^r)$$

- We can easily solve:

$$a_0 = F(h) + \frac{F(h) - F(h/n)}{n^{-p} - 1} + O(h^r)$$

- Now the error is improved to  $O(h^r)$

## Example of Richardson Extrapolation

---

- Compute

$$\int_0^{\pi/2} \sin(x) dx = 1$$

- Using trapezoidal integration and Richardson Extrapolation ( $p = 2$ ):

$$a_0 = F(h) + \frac{F(h) - F(h/2)}{2^{-2} - 1} = \frac{4F(h/2) - F(h)}{3}$$

M	F(h)	F(h/2)	a0	Rel_Err(a0)
1	0.7853982	0.9480594	1.002280	0.00228
2	0.9480594	0.9871159	1.000135	0.000135
4	0.9871159	0.9967852	1.000008	8.22E-06

- Error converges with  $O(h^4)$

## Example of Richardson Extrapolation

- Compute

$$\int_0^1 \sqrt{x} dx = \frac{2}{3}$$

- Using trapezoidal integration and Richardson Extrapolation ( $p = 2$ ):

$$a_0 = F(h) + \frac{F(h) - F(h/2)}{2^{-2} - 1} = \frac{4F(h/2) - F(h)}{3}$$

M	F(h)	F(h/2)	a0	Rel_Err(a0)
1	0.5000000	0.6035534	0.6380712	0.04289
2	0.6035534	0.6432831	0.6565263	0.0152
4	0.6432831	0.6581302	0.6630793	0.00538
8	0.6581302	0.6635812	0.6653982	0.00190
16	0.6635812	0.6655588	0.6662180	0.000673

- Error converges sub-optimally due to the fact that the derivative of  $f(x)$  is undefined at the origin. Nevertheless, Richardson Extrapolation improves result

## Rhomberg Integration

---

- Successive Richardson Extrapolations based on the trapezoidal rule with halved step sizes is known as *Rhomberg Integration*
  - Domains are successively reduced until the relative error of two successive results is less than the desired precision
  - Easily automated
  - Using uniform spacing, information can be re-used.
    - Original integral of  $N$  points
    - Refined integration only requires calculation of  $N-1$  new points

## Periodic Functions

---

- The integral of a periodic function can be performed with exponential convergence and uniform spacing using a trapezoidal rule
  - Consider, which is computed using the trapezoidal rule

$$I = \int_{x=0}^{2\pi} \cos(x) dx = 0$$

M	Result	Absolute Error
1	6.283185307179586	6.283185307179586
2	0.0000000000000000	0.0000000000000000
3	-4.44089209850062E-16	4.44089209850062E-16

- Integration is performed to machine precision (here in double precision)

## Gauss-Quadrature Rules

---

- Gauss-Quadrature rules are a set of numerical quadratures that provide *optimal* convergence rates.
  - $N$ -point rule is expected to yield  $O(h^{2N-1})$  convergence
  - This can be realized by letting both the abscissa *and* weights be unknowns
  - Thus, with  $2N$  degrees of freedom one can hope to obtain a degree of  $2N-1$
  - Gauss-Quadrature rules can be determined using the method of undetermined coefficients – but requires a non-linear solution
  - More conveniently computed using “orthogonal” polynomials
  - Gauss-Quadrature rules are generally more efficient than interpolatory (Newton-Cotes) methods even when accelerated with Richardson Extrapolation
  - Specialized Gauss-quadrature rules can be developed for integrating singular functions and unbounded intervals.
  - Trapezoidal rule is more efficient when integrating periodic functions

## Orthogonal Polynomials

---

- Define the inner product between two functions  $f$  and  $g$  as

$$\langle f, g \rangle = \int_a^b W(x) f(x) g(x) dx$$

- where  $W$  is a weight function. The two functions are orthogonal if  $\langle f, g \rangle = 0$
- We want to find a set of polynomials  $p_n(x)$  that are mutually orthogonal over the specified weight function. This can be derived using the recurrence relation:

$$p_{-1}(x) = 0, \quad p_0(x) = 1$$
$$p_{j+1}(x) = (x - a_j) p_j(x) - b_j p_{j-1}(x), \quad j = 0, 1, 2, \dots$$

- where,

$$a_j = \frac{\langle xp_j, p_j \rangle}{\langle p_j, p_j \rangle}, \{j = 0, 1, 2, \dots\}, \quad b_j = \frac{\langle p_j, p_j \rangle}{\langle p_{j-1}, p_{j-1} \rangle}, \{j = 1, 2, 3, \dots\}, (b_0 = 0)$$

## Orthogonal Polynomials, continued

---

- Example: Legendre Polynomials  $P_n(x)$

$$W(x) = 1, \quad -1 < x < 1$$

$$(j+1)P_{j+1}(x) = (2j+1)xP_j(x) - jP_{j-1}(x), \quad j = 0, 1, 2, \dots$$

$$\therefore P_0(x) = 1, \quad P_1(x) = x, \quad P_2(x) = \frac{1}{2}(3x^2 - 1), \quad P_3(x) = \frac{1}{2}(5x^3 - 3x),$$

$$P_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3)$$

- Jacobi Polynomials  $P_n^{\alpha, \beta}(x)$

$$W(x) = (1-x)^\alpha (1+x)^\beta, \quad -1 < x < 1, \quad \alpha, \beta \geq -\frac{1}{2}$$

$$c_j P_{j+1}^{\alpha, \beta}(x) = (d_j + e_j x) P_j^{\alpha, \beta}(x) - f_j P_{j-1}^{\alpha, \beta}(x), \quad j = 0, 1, 2, \dots$$

where,

$$c_j = 2(j+1)(j+\alpha+\beta+1)(2j+\alpha+\beta), \quad d_j = (2j+\alpha+\beta+1)(\alpha^2 - \beta^2)$$

$$e_j = (2j+\alpha+\beta)(2j+\alpha+\beta+1)(2j+\alpha+\beta+2)$$

$$f_j = 2(j+\alpha)(j+\beta)(2j+\alpha+\beta+2)$$

## Gauss-Quadrature Integration

---

- The orthogonal polynomials have  $j$  distinct roots over the interval  $(-1,1)$ . The abscissa of an  $N$  point quadrature rule can be chosen as the roots of the orthogonal polynomial. Solving for the weights of the quadrature rule leads to a quadrature rule of degree  $2N-1$ .

- Define:

$$\int_{-1}^1 f(x)W(x)dx \approx \sum_{i=1}^N f(x_i)w_i$$

- Constrain the weights such that the quadrature rule is satisfied for polynomials up to order  $N-1$

$$\begin{bmatrix} p_0(x_1) & \cdots & p_0(x_N) \\ p_1(x_1) & \cdots & p_1(x_N) \\ \vdots & & \vdots \\ p_{N-1}(x_1) & \cdots & p_{N-1}(x_N) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} \int_{-1}^1 W(x)p_0(x)dx \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

## Gauss-Quadrature Integration

---

- The linear system of equations:

$$\begin{bmatrix} p_0(x_1) & \cdots & p_0(x_N) \\ p_1(x_1) & \cdots & p_1(x_N) \\ \vdots & & \vdots \\ p_{N-1}(x_1) & \cdots & p_{N-1}(x_N) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} \int_{-1}^1 W(x) p_0(x) dx \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- Has a closed form solution:

$$w_j = \frac{\langle p_{N-1}, p_{N-1} \rangle}{p_{N-1}(x_j) p'_N(x_j)}, \quad j = 1, N$$

- For Legendre Polynomials:

$$w_j = \frac{2}{(1-x_j^2)(P'_N(x_j))^2}, \quad j = 1, N$$

## Gauss-Legendre Quadrature Integration

- Defined on the interval (0,1)

N = 1:

$$x(1) = .50000000000000000000000000000000$$

$$w(1) = 1.00000000000000000000000000000000$$

N = 2:

$$x(1) = .2113248654051871177450$$

$$x(2) = .7886751345948128822500$$

$$w(1) = .50000000000000000000000000000000$$

$$w(2) = .50000000000000000000000000000000$$

N = 3:

$$x(1) = .1127016653792583114820$$

$$x(2) = .50000000000000000000000000000000$$

$$x(3) = .8872983346207416885200$$

$$w(1) = .27777777777777777777777777780$$

$$w(2) = .4444444444444444444444444400$$

$$w(3) = .277777777777777777777777780$$

N=4:

$$x(1) = .0694318442029737123880$$

$$x(2) = .3300094782075718676000$$

$$x(3) = .6699905217924281324000$$

$$x(4) = .9305681557970262876100$$

$$w(1) = .1739274225687269286870$$

$$w(2) = .3260725774312730713100$$

$$w(3) = .3260725774312730713100$$

$$w(4) = .1739274225687269286870$$

N=5:

$$x(1) = .0469100770306680036010$$

$$x(2) = .2307653449471584544820$$

$$x(3) = .50000000000000000000000000000000$$

$$x(4) = .7692346550528415455200$$

$$x(5) = .9530899229693319964000$$

$$w(1) = .1184634425280945437570$$

$$w(2) = .2393143352496832340210$$

$$w(3) = .28444444444444444444444400$$

$$w(4) = .2393143352496832340210$$

$$w(5) = .1184634425280945437570$$

## Example

- Example  $I = \int_{x=0}^1 x^5 dx \approx \sum_{i=1}^N \omega_i x_i^5$     Exact I = 1/6

N	Res	Relative Error
1	0.03125	0.8125
2	0.152777777777777	0.0833
3	0.166666666666666	3.33E-16

$$I = \int_{x=0}^{\pi/2} \sin(x) dx \approx \frac{\pi}{2} \sum_{i=1}^N \omega_i \sin\left(\frac{\pi}{2} x_i\right) \quad \text{Exact I} = 1$$

N	Res	Relative Error
1	1.110720734539	0.1107
2	0.998472613404	0.001527
3	1.000008121555	8.12E-06
4	0.999999977197	2.28E-8

## Properties

---

- It can be shown that the derived quadrature rule will compute the integration of polynomials up to order  $2N-1$  *exactly*.
- The abscissas never lie on the end points (closed interval)

- Error:

$$\int_a^b f(x) dx \approx \frac{(b-a)}{2} \sum_{i=1}^N f(x_i) w_i + R_n$$

- for Gauss-Legendre Rules:

$$R_n = \frac{(b-a)^{2N+1} (N!)^4}{(2N+1)[(2N)!]^3} f^{(2N)}(\xi), \quad (-1 < \xi < 1)$$

- One can say that for arbitrary smooth functions the error converges as  $O(h^{2N-1})$ .
- Gauss-Quadrature cannot integrate to high-order discontinuous functions - unless the discontinuity is placed at an endpoint, and the integration is broken up into smaller intervals over smooth parts of the function
- Gauss-Legendre Quadrature cannot integrate to high-order singular functions - even if the singularity is integrable and at an end point. With adaptive refinement, it can converge - albeit slowly.

## Integrating Singular Functions

---

Functions with integrable singularities can be integrated to high order using specialized quadrature rules. A good example is the Gauss-Jacobi quadrature defined earlier based on Jacobi polynomials  $P_n^{\alpha,\beta}(x)$  with weights:

$$W(x) = (1-x)^\alpha (1+x)^\beta, \quad -1 < x < 1, \quad \alpha, \beta > -\frac{1}{2}$$

Other arbitrary singularities  $s(x)$  can be computed based on the moments:

$$1, s(x), x, xs(x), x^2, x^2s(x), \dots$$

defining specialized rules.

One specific rule used earlier:  $s(x) = \ln(x)$ . This was defined as the lin-log rule.

We can also manipulate existing quadrature rules to integrate known singularities.

**Example:** Consider:  $\int_0^1 y^{\frac{n-1}{\alpha}} dy \approx \sum_{i=1}^N w_i x_i^{\frac{n-1}{\alpha}}$  (singular at  $y=0$ )

$$\text{Let: } y = x^\alpha, \therefore dy = \alpha x^{\alpha-1} dx$$

$$\int_0^1 (x^\alpha)^{\frac{n-1}{\alpha}} \alpha x^{\alpha-1} dx \approx \sum_{i=1}^N (w_i \alpha x_i^{\alpha-1}) (x_i^\alpha)^{\frac{n-1}{\alpha}}$$

Therefore, define a new quadrature rule with  $\tilde{w}_i = \alpha x_i^{\alpha-1} w_i$ ,  $\tilde{x}_i = x_i^\alpha$ , which can now integrate the function  $y^{(n-1)/\alpha}$  exactly.

## Singular Integration, continued

### Example:

Let:  $\alpha = 2$ . Then,  $\tilde{w}_i = 2x_i w_i$ ,  $\tilde{x}_i = x_i^2$ . Thus, effectively, the quadrature rule will integrate exactly the moments:

$$x^{-\frac{1}{2}}, 1, x^{\frac{1}{2}}, x, x^{\frac{3}{2}}, \dots$$

An  $N$ -point rule will then integrate up to order  $(2N-3)/2$  exactly.

### Log Singularity:

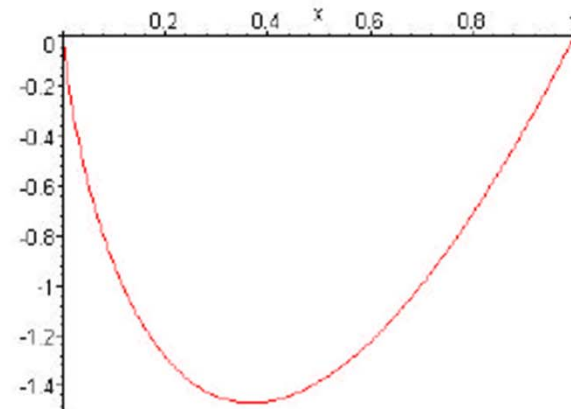
Consider:  $\int_0^1 \ln(y) dy \approx \sum_{i=1}^N w_i \ln(x_i)$  (singular at  $y=0$ )

It has been suggested to Let:  $y = x^2, \therefore dy = 2x dx$

$$\int_0^1 \ln(x^2) 2x dx \approx \sum_{i=1}^N (w_i 4x_i) \ln(x_i)$$

Which now is non-singular.

However, the derivative is singular at  $x = 0$ ! Therefore, the error can still be large. Consequently, adaptive schemes will converge slowly. On the other hand, the lin-log rule will integrate this function exactly with 1 term!



## Lin-Log Rule

---

**Lin-Log Rule:**

$$\int_0^1 f(x) dx \approx \sum_{t=1}^N w_t f(x_t), \text{ for functions: } 1, \ln(x), x, x \ln(x), x^2, x^2 \ln(x), \dots$$

**1-point Rule:** Let:  $f(x) = 1, \ln(x)$

$$f(x) = 1: \quad w_1 = \int_0^1 dx = 1$$

$$f(x) = \ln(x): \quad w_1 \ln(x_1) = \int_0^1 \ln(x) dx = -1 \Rightarrow x_1 = e^{-1} = 0.3678749441$$

$$\text{Check: } \int_0^1 \ln(x) dx \approx \sum_{t=1}^1 w_t \ln(x_t) = 1 \cdot \ln(e^{-1}) = -1$$

## Lin-Log Rule Continued

---

**2-point Rule:** Let:  $f(x) = 1, \ln(x), x, x \ln(x)$

$$f(x) = 1: \quad w_1 + w_2 = \int_0^1 dx = 1$$

$$f(x) = \ln(x): \quad w_1 \ln(x_1) + w_2 \ln(x_2) = \int_0^1 \ln(x) dx = -1$$

$$f(x) = x: \quad w_1 x_1 + w_2 x_2 = \int_0^1 x dx = \frac{1}{2}$$

$$f(x) = x \ln(x): \quad w_1 x_1 \ln(x_1) + w_2 x_2 \ln(x_2) = \int_0^1 x \ln(x) dx = -\frac{1}{4}$$

Solution (from Maple):

$$\{x1 = .6751864909, w2 = .2984998937, w1 = .7015001063, x2 = .08829686514\}$$

## Some Tabulated Lin-Log Quadrature Rules

$$I = \int_{x=0}^1 f(x) dx \approx \sum_{i=1}^N \omega_i f(x_i)$$

N = 1

x( 1) = .3678794411714422400E+00  
w( 1) = .10000000000000000000E+01

N = 2

x( 1) = .8829686513765301500E-01  
x( 2) = .6751864909098872900E+00  
w( 1) = .2984998937055248900E+00  
w( 2) = .7015001062944751000E+00

x( 1) = .2881166253095182700E-01  
x( 2) = .3040637296121376200E+00  
x( 3) = .8116692253440781200E+00  
w( 1) = .1033307079649286500E+00  
w( 2) = .4546365259700986200E+00  
w( 3) = .4420327660649726600E+00

N = 4

x( 1) = .1180259099784491700E-01  
x( 2) = .1428256799774836900E+00  
x( 3) = .4892015226545744200E+00  
x( 4) = .8786799740691836700E+00  
w( 1) = .4339102877841439800E-01  
w( 2) = .2404520976594606700E+00  
w( 3) = .4214034522597759500E+00  
w( 4) = .2947534213023489200E+00

N = 5

x( 1) = .5652228205080097200E-02  
x( 2) = .7343037174265228100E-01  
x( 3) = .2849574044625581000E+00  
x( 4) = .6194822640847783600E+00  
x( 5) = .9157580830046983800E+00  
w( 1) = .2104694579185462700E-01  
w( 2) = .1307055407444467000E+00  
w( 3) = .2897023016713141000E+00  
w( 4) = .3502203701203987700E+00  
w( 5) = .2083248416719857900E+00

## Example

---

$$I = \int_{x=0}^{0.5} Y_0(x) dx = -0.5617954591943976$$

$$\approx 0.5 \cdot \sum_{i=1}^N \omega_i Y_0(0.5x_i), \quad \text{where } Y_0(x) \text{ is the } 0^{\text{th}} \text{ - order Bessel Function of the 2nd kind}$$

Note that  $Y_0(x)$  has a log singularity at  $x = 0$

N	I	Relative Error
1	- 0.5683032670660	0.01158
2	-0.56176503071178	5.416E-05
3	-0.56179392745245	2.7265E-06
4	-0.56179547537619	2.8804E-08
5	-0.56179545927356	1.409E-10

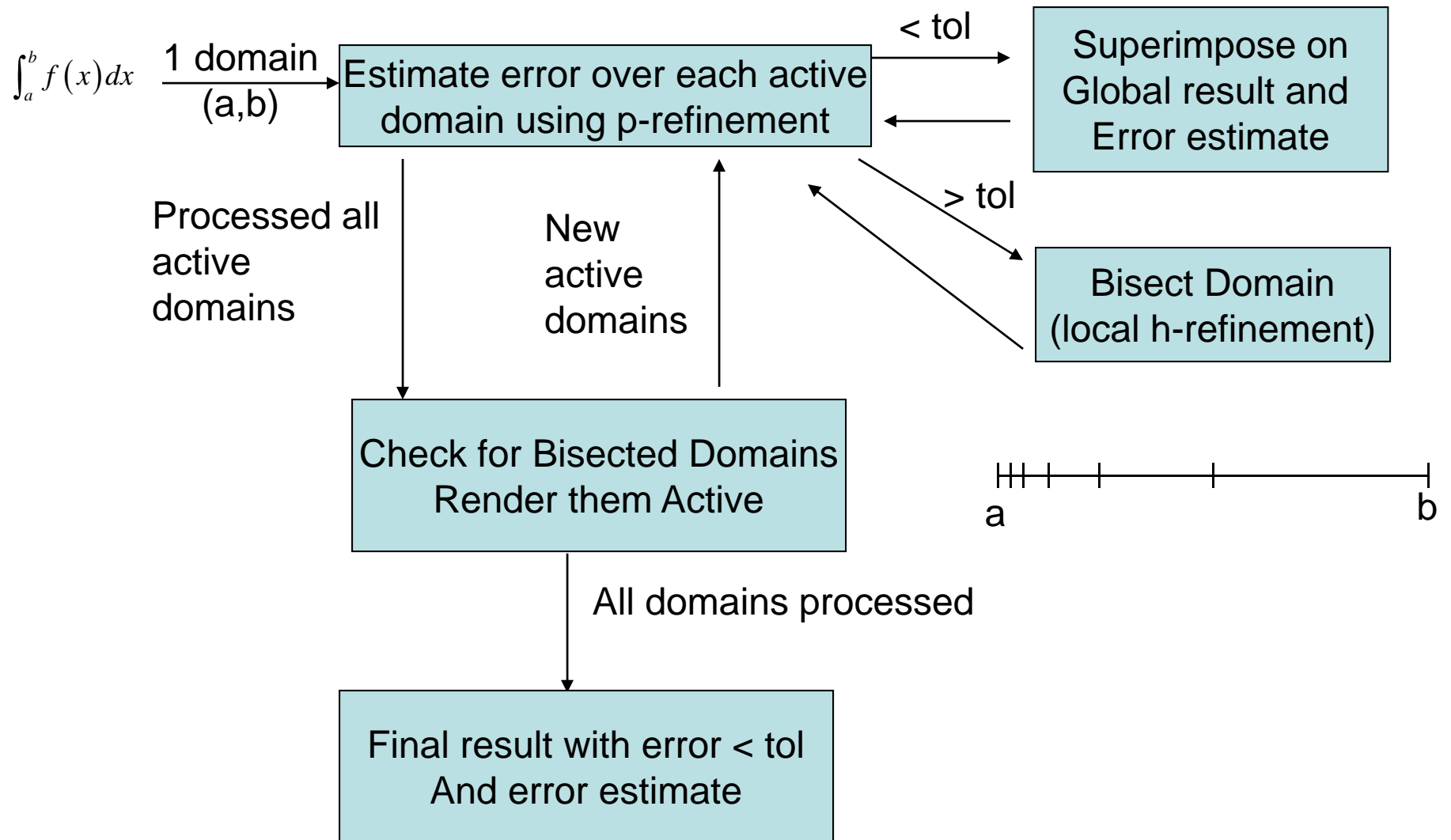
## Adaptive Quadrature

---

- Adaptive Gauss-quadrature rules typically utilize both  $p$  and  $h$  refinement
  - $p$ -refinement implies increasing the order
    - Typically do  $p$ -refinement over the same region to check error
  - $h$ -refinement implies decreasing the region
    - Typically do  $h$ -refinement if error is not within tolerance
- Adaptive algorithm implies that  $h$  or  $p$  refinement is only performed in regions where the integral is slowly convergent
  - When integrating singular integrands, integral may be slowly convergent near the singularity, and converge rapidly away from the singularity where the function is smooth
- Due to the non-uniformity of abscissa separation, reuse of information can not be done with general quadrature rules
  - There are specialized rules that allow for  $p$ -refinement
    - Gauss-Konrod Rules
      - Applicable for a limited number of quadrature orders
    - Used in the package: “Quadpack”

## Adaptive Quadrature - continued

- Typical top-level design of adaptive quadrature:



## Adaptive Gaussian Quadrature

---

- Adaptive Gaussian quadrature is typically more efficient than Romberg integration in terms of CPU time
  - It is not efficient to apply Richardson extrapolation to Gauss-quadrature rules
    - Generally cannot re-use points
    - Greater increase in the order can be achieved with a small increase in the number of quadrature points due to exponential convergence
- We can combine specialized Quadrature rules (e.g., lin-log) with Gauss-Legendre quadrature rules to rapidly accelerate the integral of functions with known singularities (such as the Hankel function)