

# Integration and quadrature

Riemann sums and why they are bad

The trapezoidal rule and Simpson's rule

The adaptive trapezoidal algorithm |

Richardson extrapolation and Romberg integration ←

Gaussian quadrature

## Riemann integration

This is the simplest approach, and comes from Riemann's definition of integration, as the limit of a sum over intervals of length  $\Delta x$ :

$$\int_a^b f(x) dx = \lim_{n \rightarrow \infty} \sum_{i=1}^n f(x_i) \Delta x$$

Handwritten red annotations: an arrow points from  $\Delta x$  to  $n$ , and another arrow points from  $n \rightarrow \infty$  to the limit symbol.

Riemann integration:

1. Divide the interval  $[a, b]$  into  $n$  subsegments of length

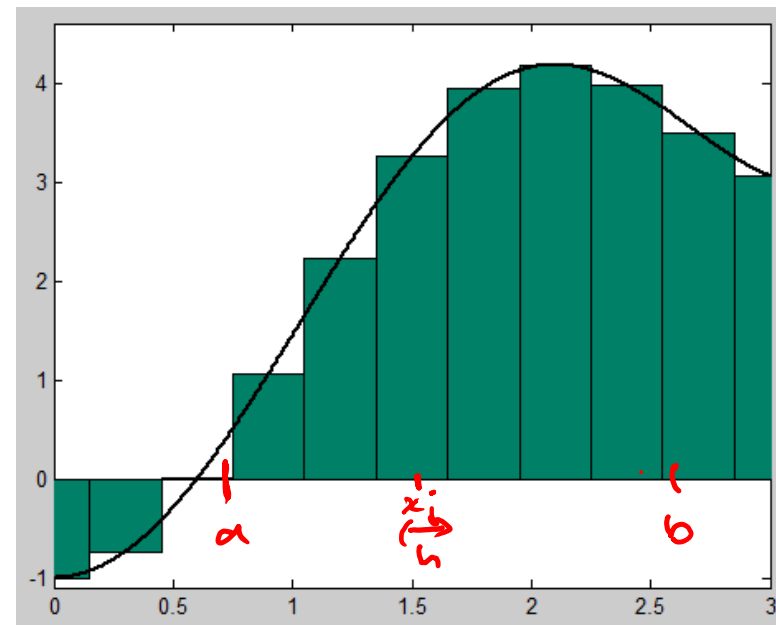
$$h = \frac{b - a}{n}$$

Handwritten red annotations: an arrow points from  $h$  to the denominator  $n$ , and another arrow points from  $n$  to the denominator.

2. Perform the sum

$$\int_a^b f(x) dx \approx \sum_{j=0}^{n-1} f(a + (j + 1/2)h)h$$

Handwritten red annotations: an arrow points from  $x_j$  to  $(j + 1/2)h$ , and another arrow points from  $x_j$  to  $h$ . A bracket underlines the entire sum expression.



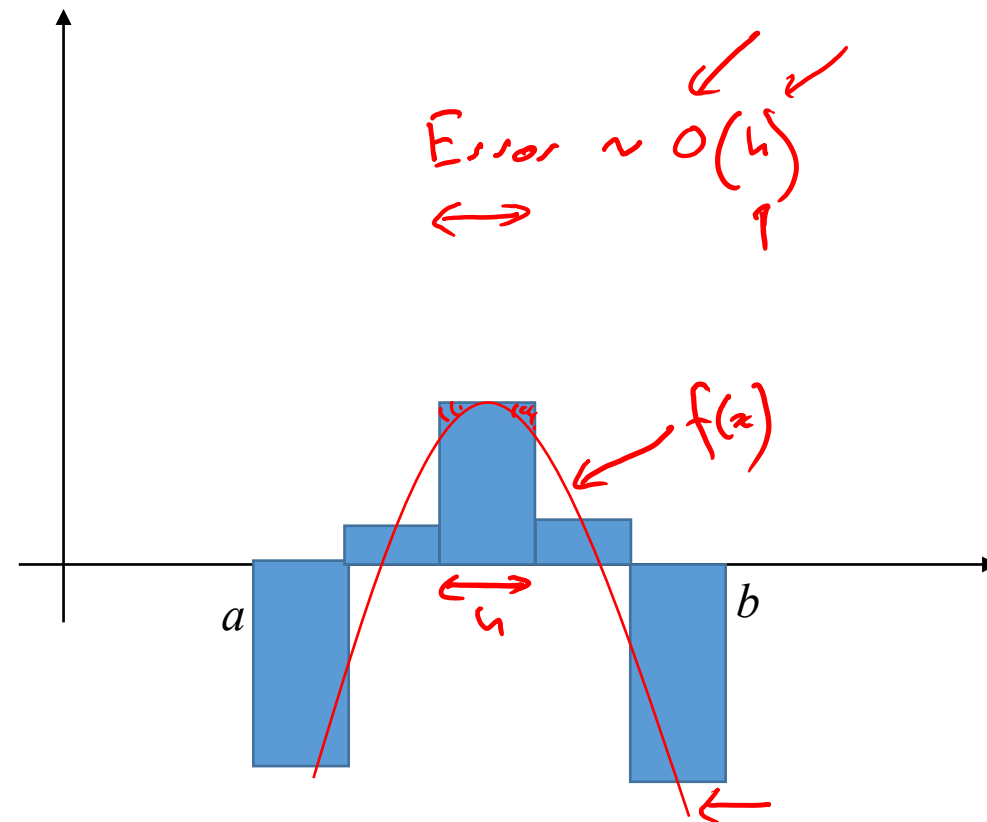
This algorithm is

- Really quick to code
- Really bad (the error is of order  $O(h^0)$ !!).

Number of boxes:  $n = \frac{b-a}{h}$

Total error  $\sim \left(\frac{b-a}{h}\right) \times O(h)$

$\sim O(h^0) \sim O(1)$



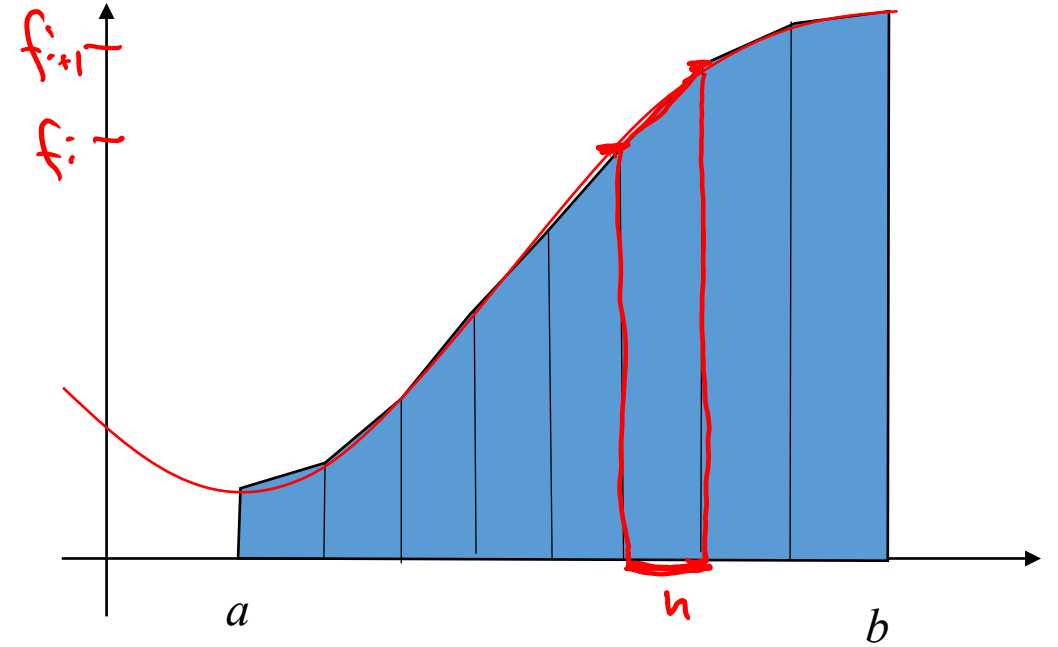
## The trapezoidal rule

This is the “rule of choice” for anything you need done quickly.

The idea: Break the interval up  $n$  into equal intervals (“panels”), then sum the areas of the trapezoids.

Area of each trapezoid:

$$\text{Area} = \frac{h}{2}(f_i + f_{i+1}) \leftarrow$$



$$\begin{aligned} \text{So Total Area} &= \sum_{i=0}^{n-1} \frac{h}{2}(f_i + f_{i+1}) = \frac{h}{2} [f_0 + f_1 + f_1 + f_2 + f_2 + f_3 + \dots + f_n] \\ &= \frac{h}{2} [f_0 + 2f_1 + 2f_2 + \dots + 2f_{n-1} + f_n] \end{aligned}$$

The trapezoidal rule:

1. Break the interval into  $n$  panels of length

$$h = \frac{b-a}{n}$$

with nodes at  $x_j = a + jh$ , and function values  $f_j = f(x_j)$

2. Form the sum

$$I = \frac{h}{2} (f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n)$$

The error in this method is *quadratic*:  
that is,

$$I = I_{\text{exact}} + O(h^2)$$

Simpson's rule: (we will derive this later)

This rule involves interpolating each set of three points by a *parabola*:

Simpson's rule:

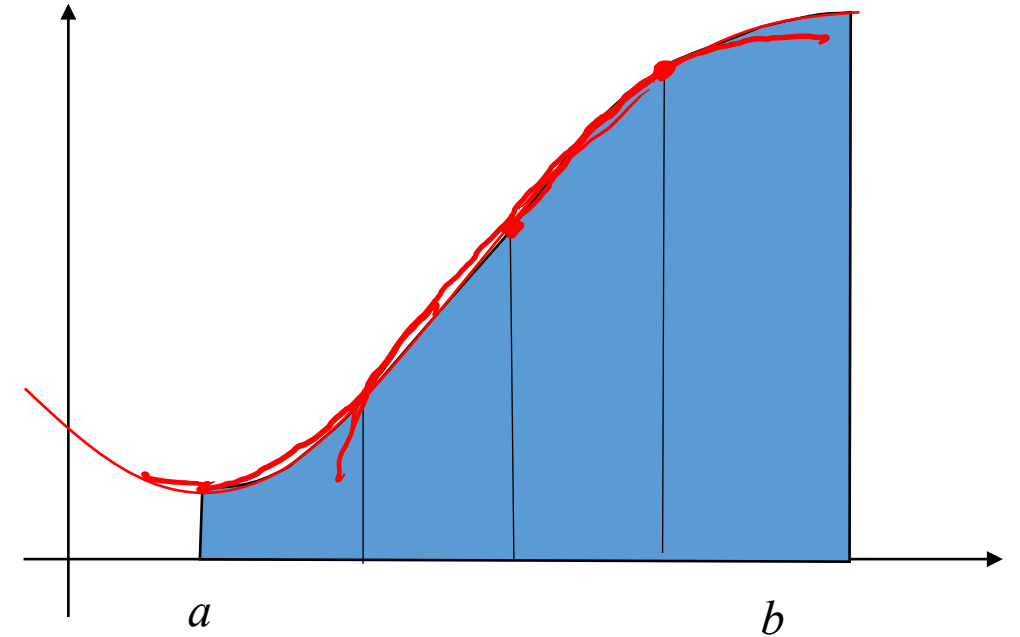
1. Break the interval into  $n$  panels of length

$$h = \frac{b - a}{n}$$

with nodes at  $x_j = a + jh$ , and function values  $f_j = f(x_j)$

2. Form the sum

$$I = \frac{h}{3} (f_0 + 4f_1 + 2f_2 + 4f_3 + \cdots + f_n)$$



The error in this method is quartic:  
that is,

$$I = I_{\text{exact}} + O(h^4)$$

The trapezoidal rule:

1. Break the interval into  $n$  panels of length

$$h = \frac{b - a}{n}$$

with nodes at  $x_j = a + jh$ , and function values  $f_j = f(x_j)$

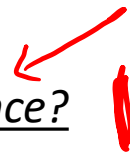
2. Form the sum

$$I = \frac{h}{2} (f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n)$$

The error in this method is *quadratic*:  
*that is,*

$$I = I_{\text{exact}} + O(h^2)$$

What if we want to work out the integral *to within a given tolerance*?



## The refined trapezoidal rule

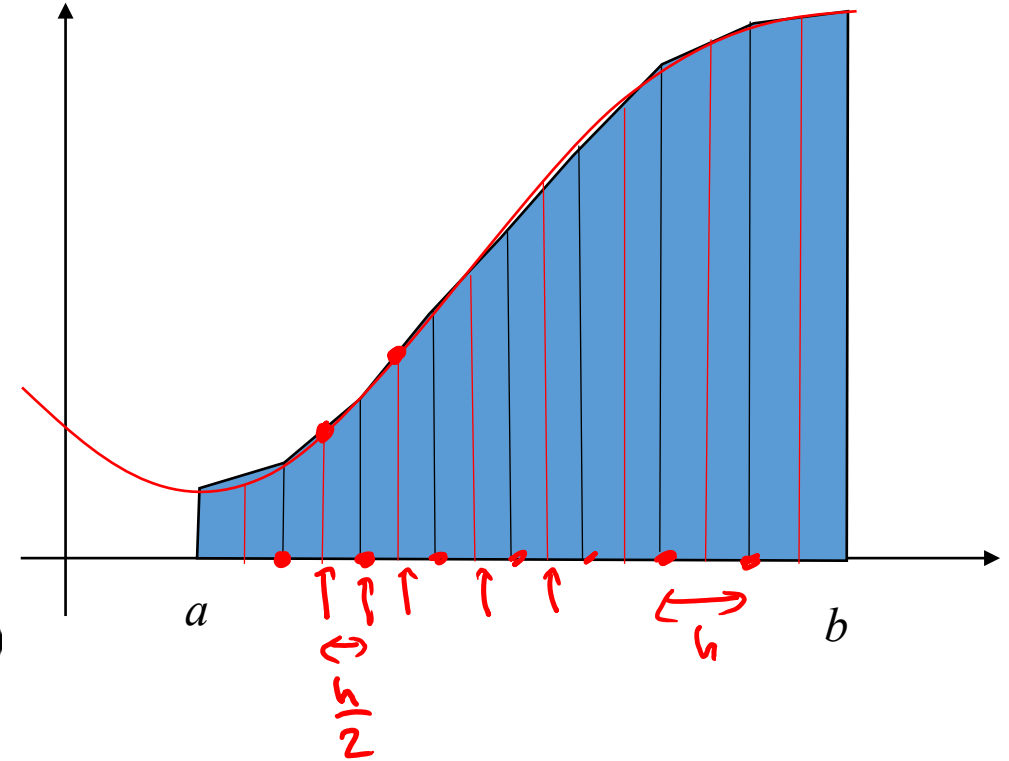
This is a great “workhorse” numerical method. The idea is that you start with a quadrature interpolation, then keep inserting points until the required tolerance is reached.

$$I_n = \frac{h}{2} (f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n) \quad \leftarrow \text{red arrow pointing to } \frac{h}{2}$$

$$I_{2n} = \frac{h}{4} (f_0 + \underbrace{2f_{i,1}}_{\uparrow} + 2f_1 + \underbrace{2f_{i,2}}_{\uparrow} + 2f_2 + \cdots + 2f_{n-1} + \underbrace{2f_{i,n}}_{\downarrow} + f_n)$$

$$= \frac{h}{4} (f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n) + \frac{h}{4} \times 2 (f_{i,1} + f_{i,2} + \cdots + f_{i,n})$$

$$= \underbrace{\frac{1}{2} I_n}_{\uparrow} + \underbrace{\frac{h}{2} \sum_{j=1}^n f_{i,j}}_{\text{correction,}}$$





## Refining trapezoid algorithm

1. Compute the trapezoidal rule with  $n$  panels and interval size  $h_n$

$$I_n = \frac{h_n}{2} (f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n) \quad \leftarrow$$

2. Compute the nodes of the intermediate points

$$x_{i,j} = \frac{1}{2}(x_j + x_{j+1}) \quad , j = \{1 \dots n\} \quad \leftarrow$$

3. Form the new integral, stop if converged

$$I_{2n} = \frac{1}{2}I_n + \frac{h_n}{2} \sum_{j=1}^n f(x_{i,j}) \quad \leftarrow$$

4. Merge the intermediate points  $x_{i,j}$  with the set of nodes  $x_j$ ,

5. Set  $h_{2n} = h_n/2$  and repeat from Step 2.  $\leftarrow$

## Richardson extrapolation

The trapezoidal rule is an *approximation* to the exact value of the integral. That is,

$$I_T = I_{\text{exact}} + \text{Error}(h)$$

↑      ↑  
Trapezoidal      Error

We can show that the error is quadratic in  $h$ , that is

$$I_T = \underline{I_{\text{exact}}} + c_1 \underline{h^2} + \underline{O(h^4)}$$

We can derive *new rules of integration* by trying to cancel out this error.

To see how this works, let's see what happens when we halve the stepsize  $h$ :

$$I_T(h) = I_{\text{exact}} + c_1 h^2 \quad \leftarrow (1)$$

$$I_T\left(\frac{h}{2}\right) = I_{\text{exact}} + c_1 \left(\frac{h}{2}\right)^2 \quad \leftarrow$$

So

$$I_T\left(\frac{h}{2}\right) = I_{\text{exact}} + \frac{1}{4} c_1 h^2$$

$$\Rightarrow 4 I_T\left(\frac{h}{2}\right) = 4 I_{\text{exact}} + c_1 h^2 \quad \leftarrow (2)$$

Subtract (2) - (1):

$$4 I_T\left(\frac{h}{2}\right) - I_T(h) = 4 I_{\text{exact}} + \cancel{c_1 h^2} - (I_{\text{exact}} + \cancel{c_1 h^2})$$

So

$$4 I_T\left(\frac{h}{2}\right) - I_T(h) = 3 I_{\text{exact}} + O(h^4)$$

Then

$$\underline{I_{\text{exact}}} = \frac{1}{3} \left[ 4 I_T\left(\frac{h}{2}\right) - I_T(h) \right] + O(h^4)$$

↑  
 $I_{\text{new}}$

We have found a new rule:

$$I_{\text{new}} = \frac{1}{3} [4I_T(h/2) - I_T(h)] + O(h^4)$$

What does this look like?

Rewrite

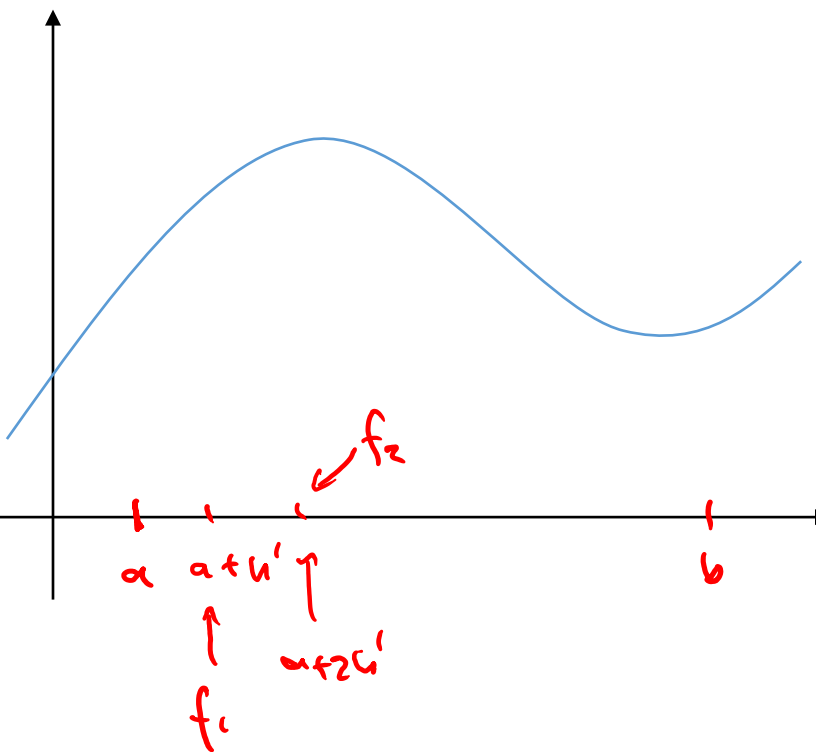
$$h' = \frac{h}{2}$$

$$I_{\text{new}} = \frac{1}{3} [4 \underline{I_T(h')} - \underline{I_T(2h')}]$$

$$= \frac{1}{3} \left[ 4 \frac{h'}{2} (f_a + 2f_1 + 2f_2 + \dots + f_b) \right.$$

$$\left. - \frac{(2h')}{2} (f_a + 2f_2 + 2f_4 + \dots + f_b) \right]$$

$$= \frac{h'}{3} [f_a + 4f_1 + 2f_2 + 4f_3 + \dots + f_b]$$



← Simpson's rule.

Can we design higher order schemes? The answer is yes -  
This is called Romberg integration

$$\underline{I_1(h)} = I_T(h)$$

$$\underline{I_2(h)} = \frac{1}{3} \left[ I_1\left(\frac{h}{2}\right) - I_1(h) \right]$$

Now,

$$I_2(h) = I_{\text{exact}} + c_2 h^4$$

$$16 I_2\left(\frac{h}{2}\right) = 16 I_{\text{exact}} + c_2 \left(\frac{h}{2}\right)^4 \cdot 16$$

$$16 I_2\left(\frac{h}{2}\right) - I_2(h) = 15 I_{\text{exact}} + O(h^6)$$

$$I_{\text{exact}} = \frac{1}{15} \left[ 16 I_2\left(\frac{h}{2}\right) - I_2(h) \right]$$

In general

$$I_j(h) = I_{\text{exact}} + c_j h^{2j}$$

$$I_j\left(\frac{h}{2}\right) = I_{\text{exact}} + c_j \left(\frac{h}{2}\right)^{2j}$$

$$\Rightarrow I_{j+1}(h) = \frac{1}{2^{2j} - 1} \left[ 2^{2j} I_j\left(\frac{h}{2}\right) - I_j(h) \right]$$

Higher-order quadrature schemes can be derived from the trapezoidal rule. Each iteration pushes the error out to a higher order in  $h$ .

## Romberg integration

1. Compute the integral according to the trapezoidal rule with stepsize  $h$ , and 2 panels

$$R(n, 0) = I_T(h)$$

*number of intervals* (pointing to  $n$ )  
*Order of method.* (pointing to  $0$ )

2. While the integral is not converged:

- Set new panel size  $h_n = \frac{b-a}{2^n}$
- Compute the integral according to the trapezoidal rule with stepsize  $h_{n+1}$ , (2n panels):

$$R(n+1, 0) = I_T(h_{n+1})$$

- Perform the extrapolation to get the integral according to the higher-order rules:

$$R(n+1, j) = \frac{1}{2^{2j}} \left[ 2^{2j} R(n+1, j-1) - R(n, j-1) \right] \quad j = 1 \dots n+1$$

- Update  $n = n+1$ . The final term  $I = R(n, n)$  is the value of the integral, Repeat Step 2.

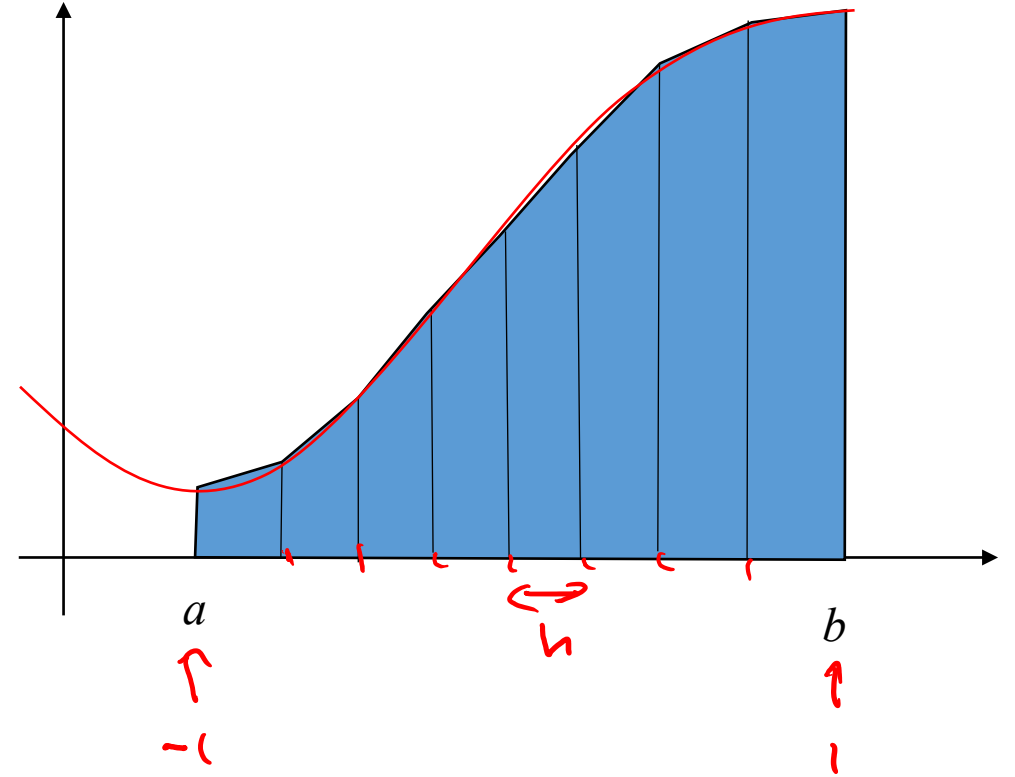
## Gaussian Quadrature

All our quadrature schemes thus far have involved picking Equally-space points. In each case we have ended up with a formula like

$$I = \frac{h}{2} (f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n)$$
$$= \frac{h}{2} f_0 + 2\frac{h}{2} f_1 + 2\frac{h}{2} f_2 + \cdots + 2\frac{h}{2} f_{n-1} + \frac{h}{2} f_n$$

We can get faster, more accurate integrals if we are *free to pick our points as we like*. This idea leads to Gaussian quadrature.

Important point: Most Gaussian quadrature schemes are derived for the interval [-1,1].  
We'll do the change of variables later.



Imagine that we pick a set of points  $x_j$ . We can approximate our function by Lagrange interpolation:

$$f(x) \approx \sum_j^n \ell_j(x) f_j$$

*Handwritten notes:* "approximately equal" with an arrow pointing to the approximation symbol  $\approx$ . A red underline is under the summation  $\sum_j^n$ . Blue arrows point from  $\ell_j(x)$  and  $f_j$  to the right.

Note that when  $f$  is a polynomial of order  $\leq n$ , this approximation is exact.

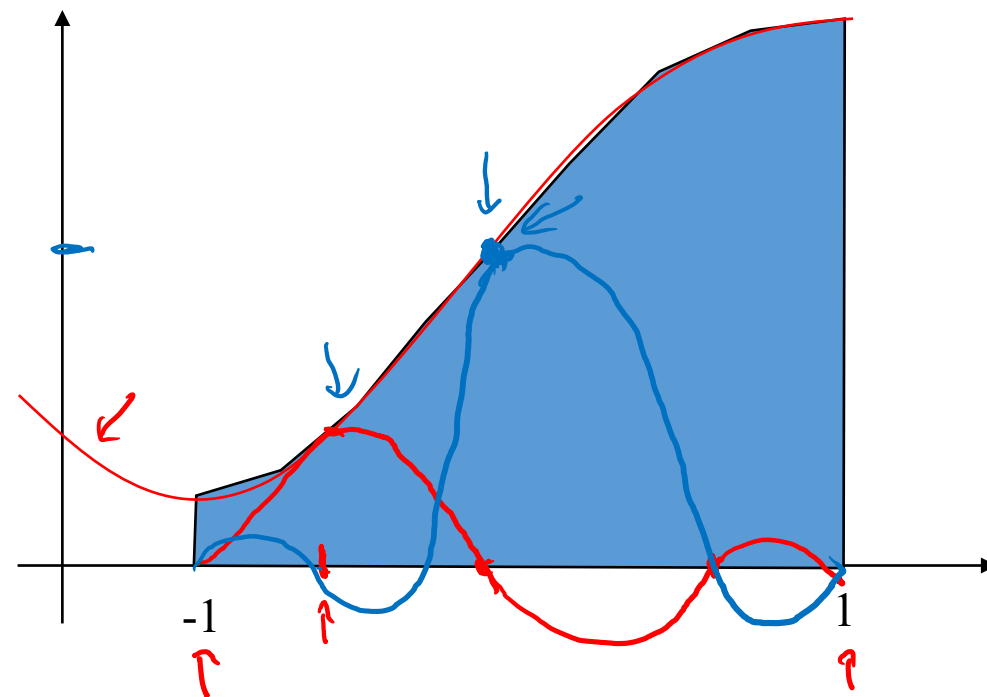
Integrating, we find

$$\int_{-1}^1 f(x) dx \approx \int_{-1}^1 \sum_{j=1}^n \ell_j(x) f_j dx$$

$$= \sum_{j=1}^n f_j \underbrace{\int_{-1}^1 \ell_j(x) dx}_{w_j} = \sum_{j=1}^n f_j w_j$$

*Handwritten notes:* The first integral is underlined in blue. The second integral is underlined in blue and labeled  $w_j$ . The  $f_j$  term has an arrow pointing to it from the  $f_j$  in the sum above.

How do we choose the points  $x_j$ ? It turns out that a really, really good choice is that we choose them to be the zeros of orthogonal polynomials.



$$\ell_j(x) = \frac{\prod_{i \neq j} (x - x_i)}{\prod_{i \neq j} (x_j - x_i)}$$

where  $w_j = \int_{-1}^1 \ell_j(x) dx$

*Handwritten notes:* The integral is underlined in blue.

## Orthogonal polynomials

Consider a set of polynomials  $p_j(x)$  of degree  $1 \dots n$ . We say that This set is *orthogonal over an interval*  $[-1,1]$  if

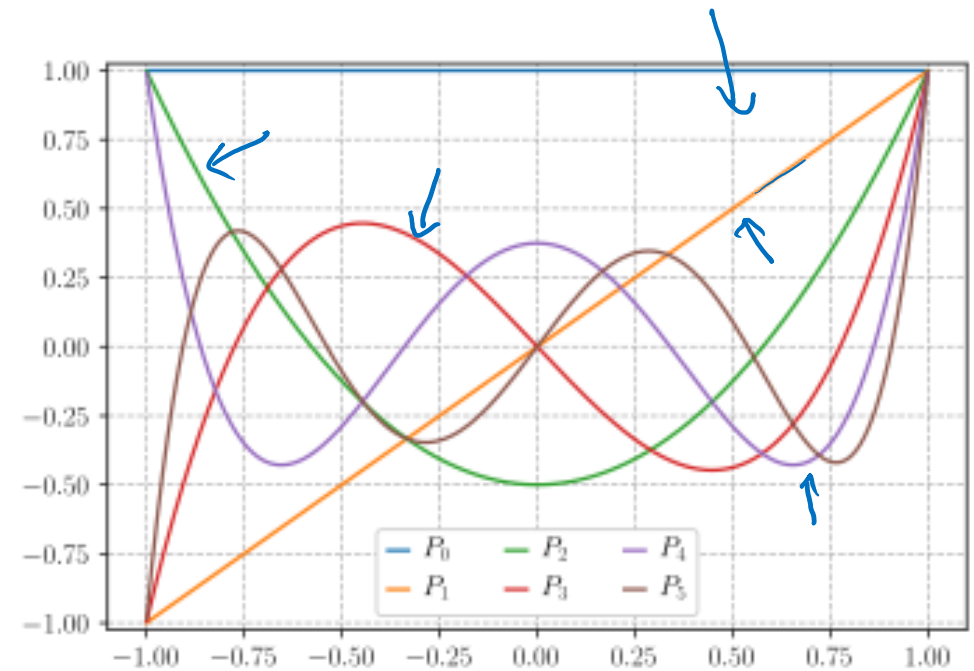
$$\int_{-1}^1 p_i(x) p_j(x) w(x) dx = 0 \quad \text{for } i \neq j$$

← weight function

The type of polynomial depends on the weight function  $w(x)$ :

Weight $w(x)$	Type of Polynomial
1	Legendre
$\frac{1}{\sqrt{1-x^2}}$	Chebyshev
$e^{-x^2}$	Hermite
$x^\alpha e^{-x}$	Laguerre

$$\int_{-1}^1 P_0(x) P_1(x) dx = \int_{-1}^1 1 \cdot x dx = \left[ \frac{x^2}{2} \right]_{-1}^1 = 0$$



$n$	$P_n(x)$
0	1
1	$x$
2	$\frac{1}{2} (3x^2 - 1)$
3	$\frac{1}{2} (5x^3 - 3x)$
4	$\frac{1}{8} (35x^4 - 30x^2 + 3)$
5	$\frac{1}{8} (63x^5 - 70x^3 + 15x)$
6	$\frac{1}{16} (231x^6 - 315x^4 + 105x^2 - 5)$
7	$\frac{1}{16} (429x^7 - 693x^5 + 315x^3 - 35x)$
8	$\frac{1}{128} (6435x^8 - 12012x^6 + 6930x^4 - 1260x^2 + 35)$
9	$\frac{1}{128} (12155x^9 - 25740x^7 + 18018x^5 - 4620x^3 + 315x)$
10	$\frac{1}{256} (46189x^{10} - 109395x^8 + 90090x^6 - 30030x^4 + 3465x^2 - 63)$

All these have been computed and their zeros tabulated.

In python, the Legendre polynomials are found by calling `p_roots(n)`



Once we have the polynomials, we can look up the zeros and the weights, and evaluate

$$w_j = \int_{-1}^1 l_j(x) dx$$

$$\int_{-1}^1 f(x)w(x)dx \approx \sum_{j=1}^n w_j f(x_j)$$

We usually have to *change the interval* of integration. This involves the change of coordinates

$$\int_a^b f(x)dx = \int_{-1}^1 f\left(\frac{b-a}{2}u + \frac{a+b}{2}\right) \frac{dx}{du} du$$

Number of points, $n$	Points, $x_i$		Weights, $w_i$	
1	0		2	
2	$\pm \frac{1}{\sqrt{3}}$	$\pm 0.57735\dots$	1	
3	0		$\frac{8}{9}$	0.888889...
	$\pm \sqrt{\frac{3}{5}}$	$\pm 0.774597\dots$	$\frac{5}{9}$	0.555556...
4	$\pm \sqrt{\frac{3}{7} - \frac{2}{7}\sqrt{\frac{6}{5}}}$	$\pm 0.339981\dots$	$\frac{18 + \sqrt{30}}{36}$	0.652145...
	$\pm \sqrt{\frac{3}{7} + \frac{2}{7}\sqrt{\frac{6}{5}}}$	$\pm 0.861136\dots$	$\frac{18 - \sqrt{30}}{36}$	0.347855...
5	0		$\frac{128}{225}$	0.568889...
	$\pm \frac{1}{3}\sqrt{5 - 2\sqrt{\frac{10}{7}}}$	$\pm 0.538469\dots$	$\frac{322 + 13\sqrt{70}}{900}$	0.478629...
	$\pm \frac{1}{3}\sqrt{5 + 2\sqrt{\frac{10}{7}}}$	$\pm 0.90618\dots$	$\frac{322 - 13\sqrt{70}}{900}$	0.236927...




$$\int_a^b f(x)w(x)dx \approx \frac{b-a}{2} \sum_{j=1}^n w_j f\left(\frac{b-a}{2}u_j + \frac{a+b}{2}\right)$$

## Gaussian quadrature

To integrate functions of the type

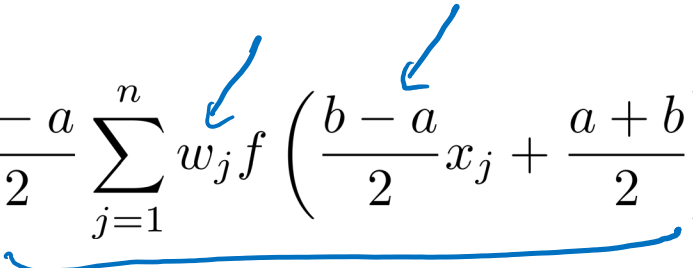
$$\int_a^b f(x)w(x)dx$$

  $w(x)=1$  is often the case.

1. Choose an appropriate set of polynomials for the weighting, and the number of points  $n$  at which you are using to integrate the function

2. Evaluate the sum

$$\int_a^b f(x)w(x)dx \approx \frac{b-a}{2} \sum_{j=1}^n w_j f\left(\frac{b-a}{2}x_j + \frac{a+b}{2}\right)$$



where the  $w_j$  and  $x_j$  are the tabulated weights and zeros for the expansion.

Gaussian quadrature is

1. Simple to implement
2. Extremely fast
3. More accurate (by far) than almost any other method

## Why Gaussian quadrature works so well

A Gaussian quadrature with  $n$  points integrates polynomials of *degree less than  $2n$  exactly*. That is, it is unreasonably effective in “capturing the bumps” of a function.

To see this, consider integrating a function  $f(x)$  with polynomial order  $< 2n$ . We can write

$< 2n$ . We can write

$$f(x) = q(x)p_n(x) + r(x)$$

Diagram illustrating the division of  $f(x)$  by  $p_n(x)$ :

- $q(x)$  is the quotient (polynomial of order  $< n$ ).
- $p_n(x)$  is the polynomial of order  $n$ .
- $r(x)$  is the remainder.

Then:

$$\int f(x) dx = \int q(x) p_n(x) dx + \int r(x) dx$$

We expand the quotient in terms of the orthogonal polynomials

$$q(x) = \sum_{i=0}^{n-1} q_i p_i(x)$$

Then, because of orthogonality

$$\int q(x) p_n(x) dx = \sum_{i=0}^{n-1} \int p_i(x) p_n(x) dx$$

$= 0$  because  $i \neq n$

So we have shown that

$$\int f(x)dx = \int r(x)dx$$

Note that  
this is  
exact, not  
an approximation.

We now interpolate  $r(x)$  at the nodes  $x_j$ :

$$r(x) = \sum_{j=1}^n r(x_j) l_j(x)$$

So

$$\begin{aligned} \int r(x)dx &= \int \sum_{j=1}^n r(x_j) l_j(x) dx \\ &= \sum_{j=1}^n r(x_j) \int l_j(x) dx = \sum_{j=1}^n r(x_j) w_j \end{aligned}$$

But

$$\begin{aligned} f(x_j) &= p_n(x_j) q(x_j) + r(x_j) \\ &= 0 + r(x_j) \end{aligned}$$

because  $x_j$  is a zero of  $p_n(x)$ .

Therefore

$$\int f(x)dx = \sum_{j=1}^n f(x_j) w_j$$

where this is an exact relation,  
not an approximation.

$\Rightarrow$  any integral of polynomial order  
less than  $2n$  is exactly  
integrated by Gaussian quadrature.