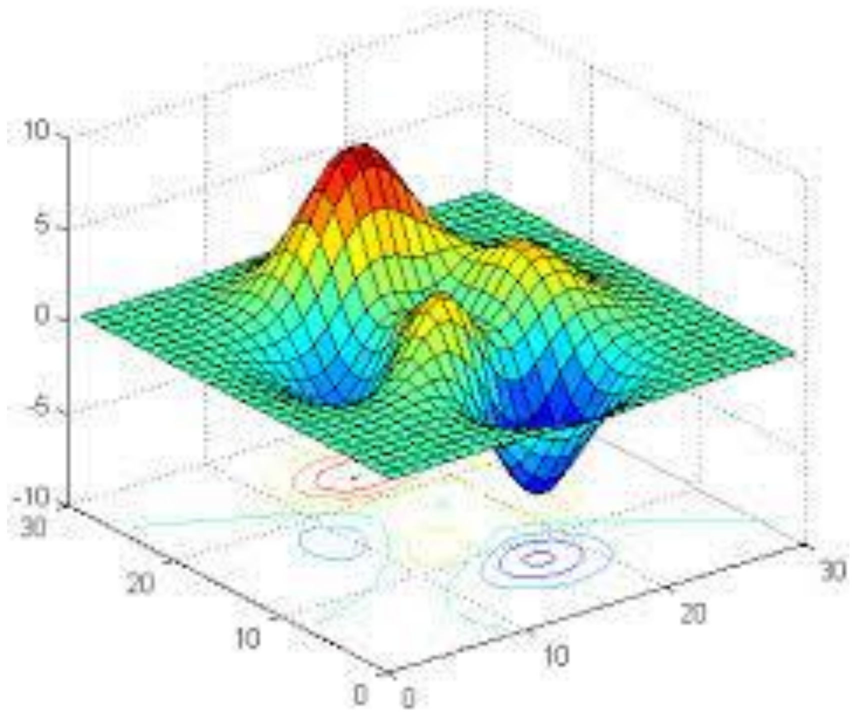
Khemis Miliana University

Faculty of Sciences of Matter and Computer Science

Department of Physics



Numerical Methods & Scientific Programming

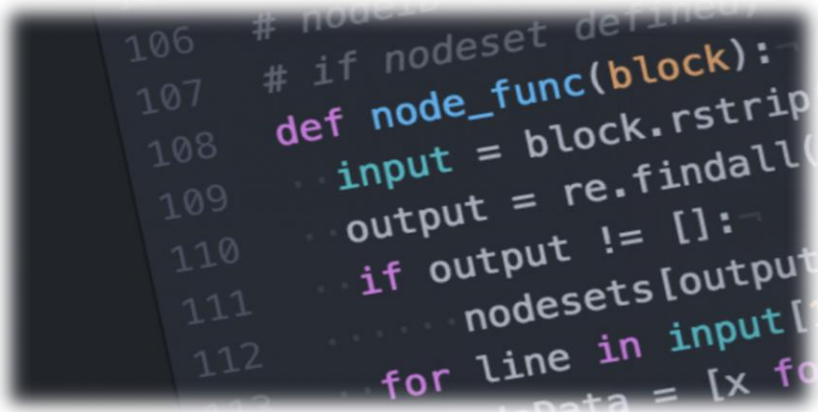
Dr. Salah-Eddine BENTRIDI

s.bentridi@univ-dbkm.dz

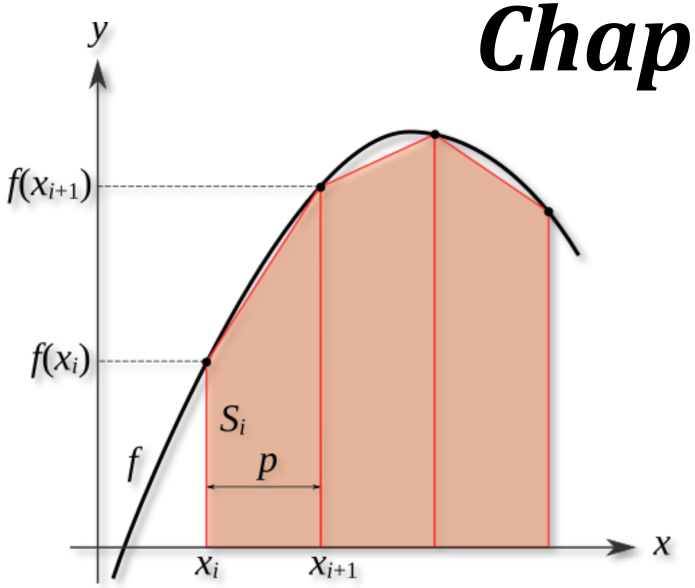
Univ. Khemis-Miliana

Content of the program

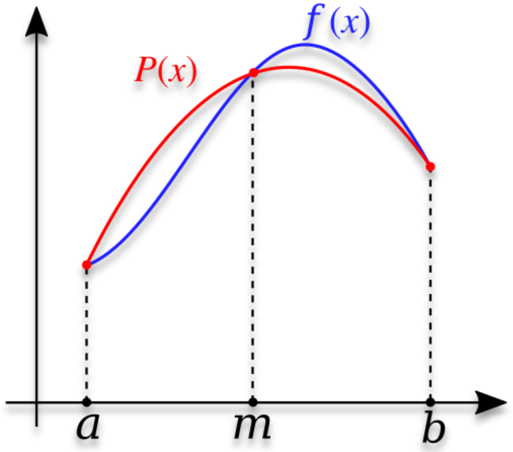
- Chapter 01: ***Initiation to a programming language (Python)***
  - ***Hands-on-Pythn; Basics of Python***
- Chapter 02: ***Numerical Integration***
  - ***Trapezoidal rule; Simpson's method***
- Chapter 03: ***Numerical Solution of equations***
  - ***Bisection method; Newton's Method***
- Chapter 04: ***Numerical resolution of differential equations***
  - ***Euler's method; Runge-Kutta method***
- Chapter 05: ***Numerical resolution of linear equations system***
  - ***Gauss method, Gauss-Seidel method***



Chapter 02: Numerical Integration



Trapezoidal rule and Simpson's rule



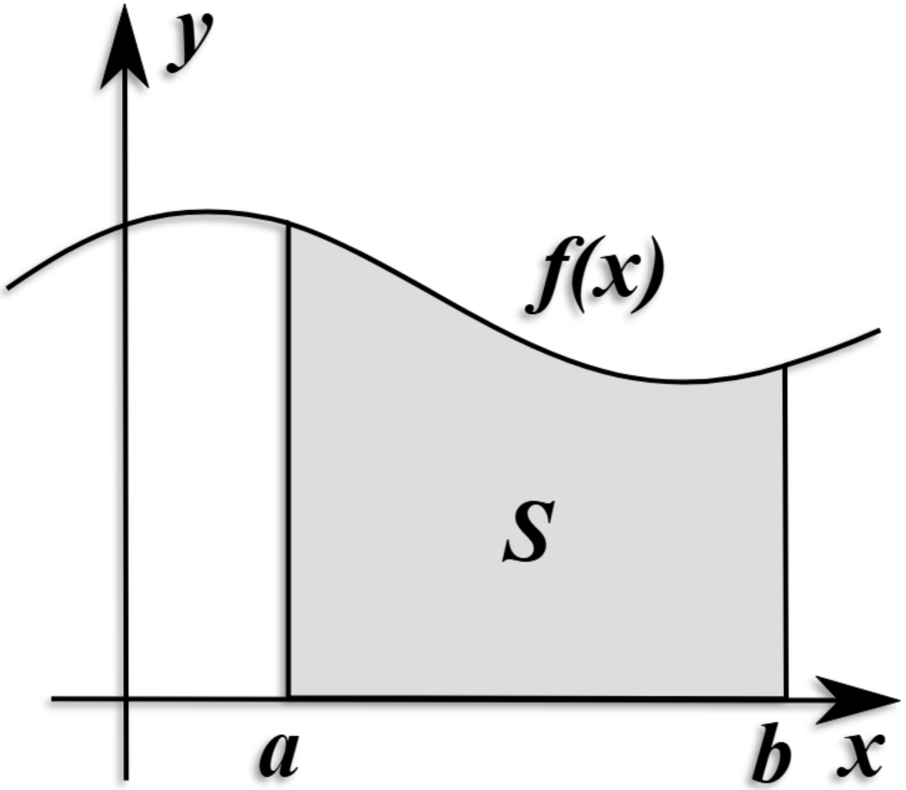
Dr. Salah-Eddine BENTRIDI

s.bentridi@univ-dbkm.dz

Khemis-Miliana University

Outline

- Numerical integration problem statement
- Principle of numerical integration
- Newton-Cotes quadrature rules
- Trapezoidal rule
- Simpson's rule



I. Numerical Integration Problem Statement

Integrals of functions often appear in applications as well as in various computations. For example, the energy of a system is usually expressed as an integral. The issue is that most integrals do not have closed form solutions. For instance, the integral

$$\boldsymbol{I[f] = \int_a^b \sqrt{1 + \cos^2(x)}.\,dx}$$

is a case where its computation is extremely hard if not impossible. In such cases we need automated methods to approximate integrals.

I. Numerical Integration Problem Statement

To achieve such calculation, one could use "numerical integration" instead of "analytical calculation". Numerical integration, also known as quadrature, is intrinsically a much more accurate procedure than numerical differentiation.

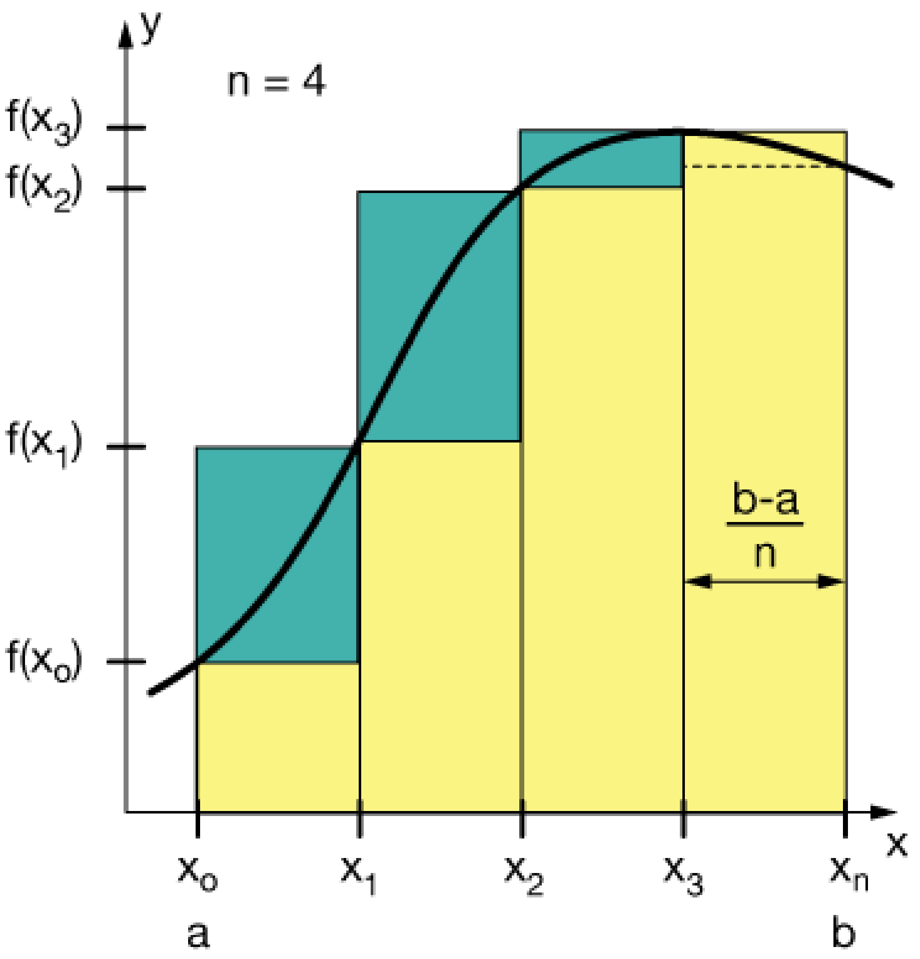
Quadrature approximates the definite integral : $I[f] = \int_a^b f(x).dx$ *By the sum:*

$$I_N[f] = \sum_{i=0}^{N} w_i . f(z_i)$$

Nodal abscissas z_i *and weights* w_i *depend on the particular rule used for the quadrature. All rules of quadrature are derived from polynomial interpolation of the integrand. Therefore, they work best if* $f(x)$ *can be approximated by a polynomial.*

II. Principle of numerical integration

- *Given a function* $f(x)$ *, we want to approximate the integral of* $f(x)$ *over the total interval* $[a, b]$ *.*
- *To do that, we assume that the interval has been discretized into a numeral grid* x*, consisting of* $n + 1$ *points* $(x_0 \to x_n)$ *with a width* $h = (b - a)/n$ *.*
- *Here, we denote each* x *by* x_i *,* $x_0 = a$ *and* $x_n = b$*.*
- *The height is defined by a function value* $f(x)$ *for some* x *in the subinterval* $[x_i, x_{i+1}]$

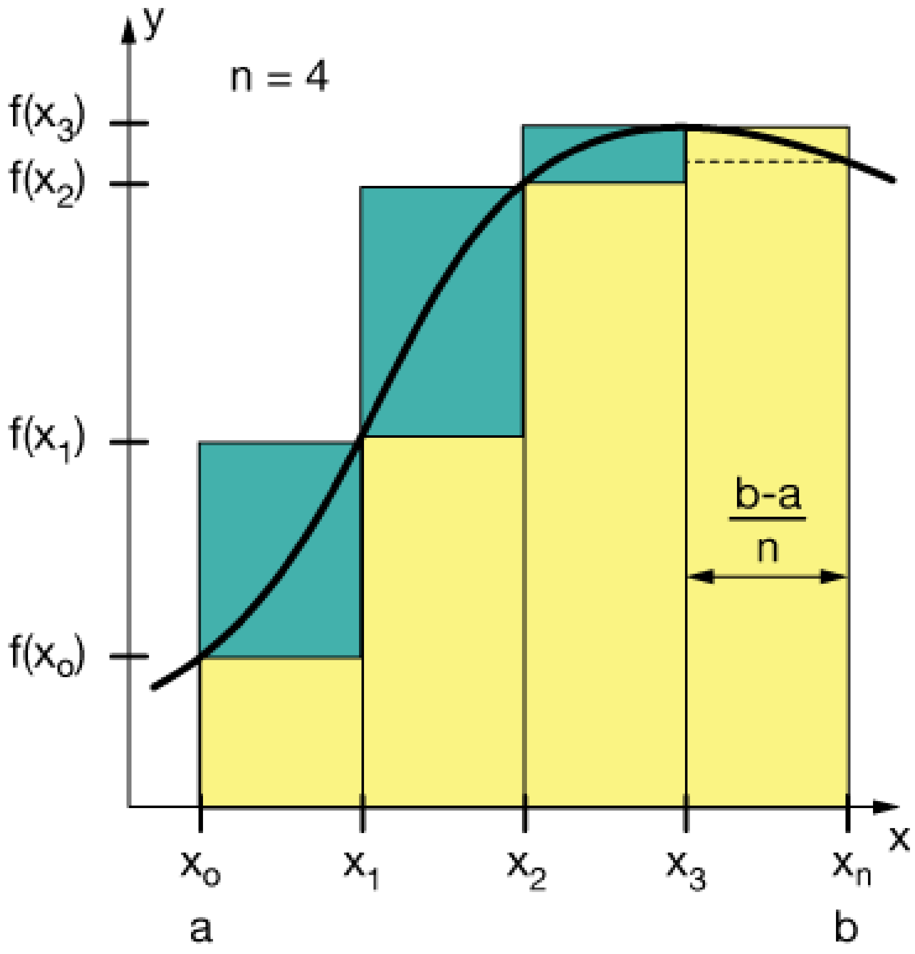


II. Principle of numerical integration

- ***An obvious choice for the height is the function value at the left endpoint, x_i (Left Integral), or the right endpoint, x_{i+1} (Right Integral).***

- ***This method is known as the "Rectangle rule":***

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n-1} h.f(x_i) \equiv \frac{b-a}{n}\sum_{i=0}^{n-1} f(x_i)$$

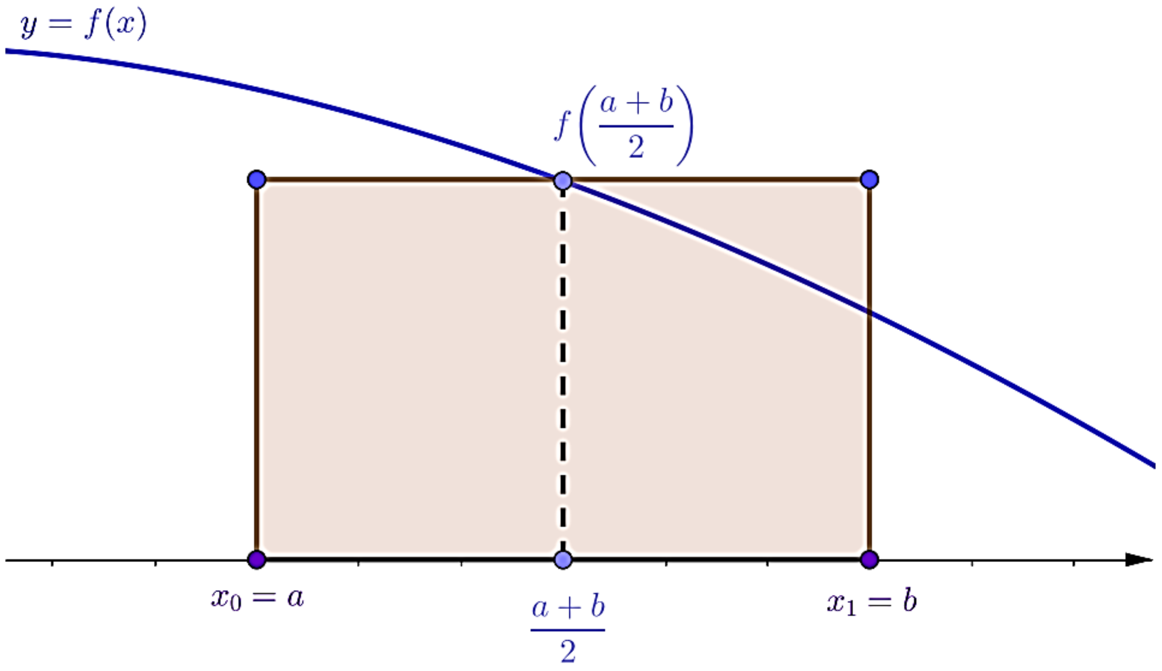


II. Principle of numerical integration

- ***If the height is taken at the middle of the subinterval*** $[x_i, x_{i+1}]$ ***:***

$$z_i = \frac{x_{i+1}+x_i}{2} \rightarrow f(z_i) = f\left(\frac{x_{i+1}+x_i}{2}\right).$$

- ***This is a specific case of Rectangle rule approximation, known as "Midpoint rule" given by:***



$$\int_a^b f(x)dx \approx \frac{b-a}{n}\sum_{i=0}^{n-1} f\left(\frac{x_{i+1}+x_i}{2}\right) \equiv \sum_{i=0}^{n-1} h.f(z_i)$$

II. Principle of numerical integration

- ***Simple rule (one simple width):***

$$\int_a^b f(x)dx \approx (b-a).f(z_i)\,;z_i \in [a,b]$$

- ***Composite rule (several uniformly distributed widths):***

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n-1} h.f(z_i) \equiv \frac{b-a}{n}\sum_{i=0}^{n-1} f(z_i)\,;$$

$$z_i \in [x_i, x_{i+1}];\ h = x_{i+1} - x_i = (b-a)/n$$

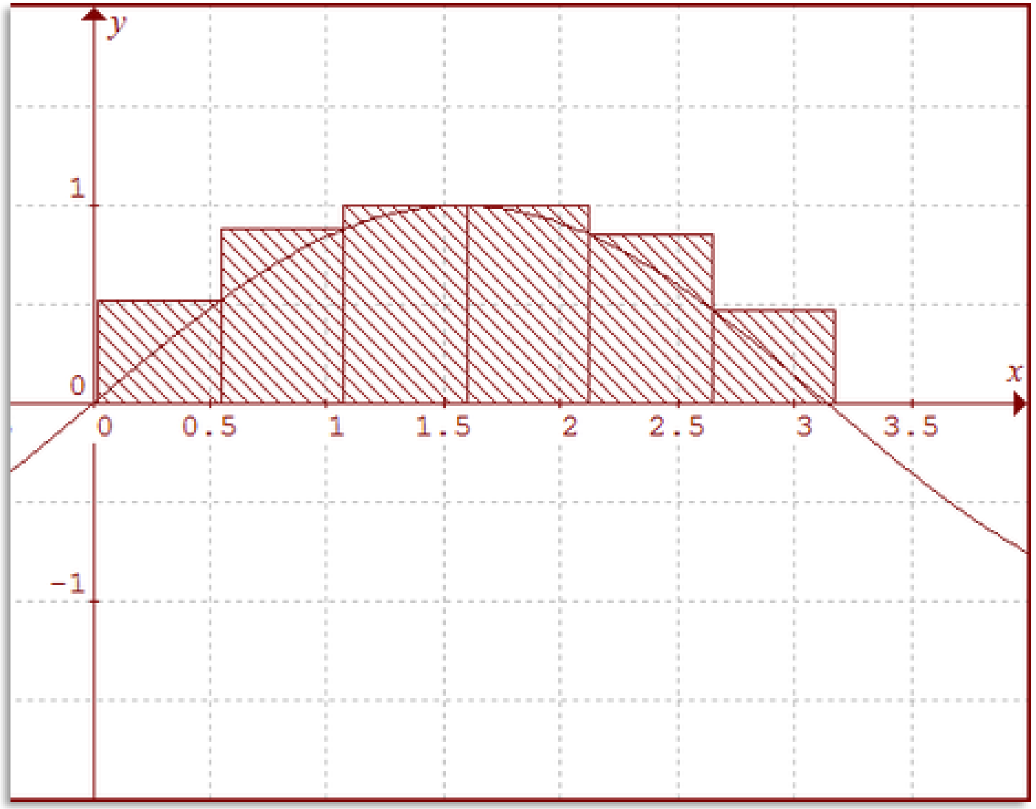
II. Principle of numerical integration

Application 01:

Write a Python code to use the left Rectangle rule, right Rectangle rule, and Midpoint Rule to approximate :

$$\int_0^{\pi} sin(x)dx$$

with 11 evenly spaced grid points over the whole interval.

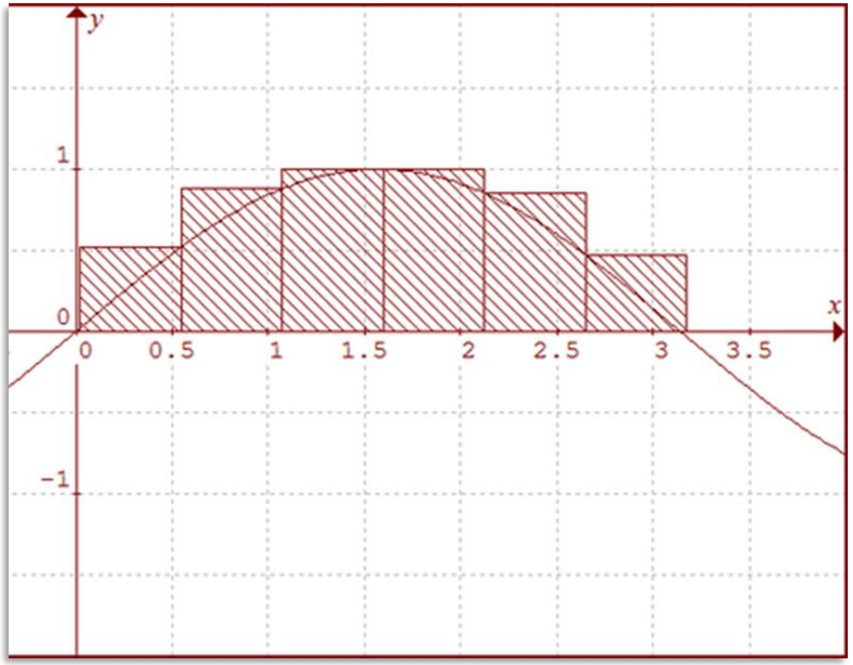


Try other values of n, and compare each time the obtained results to the exact value of 2. Comment the quality of results according to errors.

II. Principle of numerical integration

Partial Solution (complete for Right and midpoint rule)

```python
import numpy as np
import matplotlib.pyplot as plt
```




```python
a = 0
b = np.pi
n = 10
h = (b - a) / n
x = np.linspace(a, b, n+1)
f = np.sin(x)
```

```python
I_rectL = h * sum(f[:n])
err_rectL = 2 - I_rectL
```

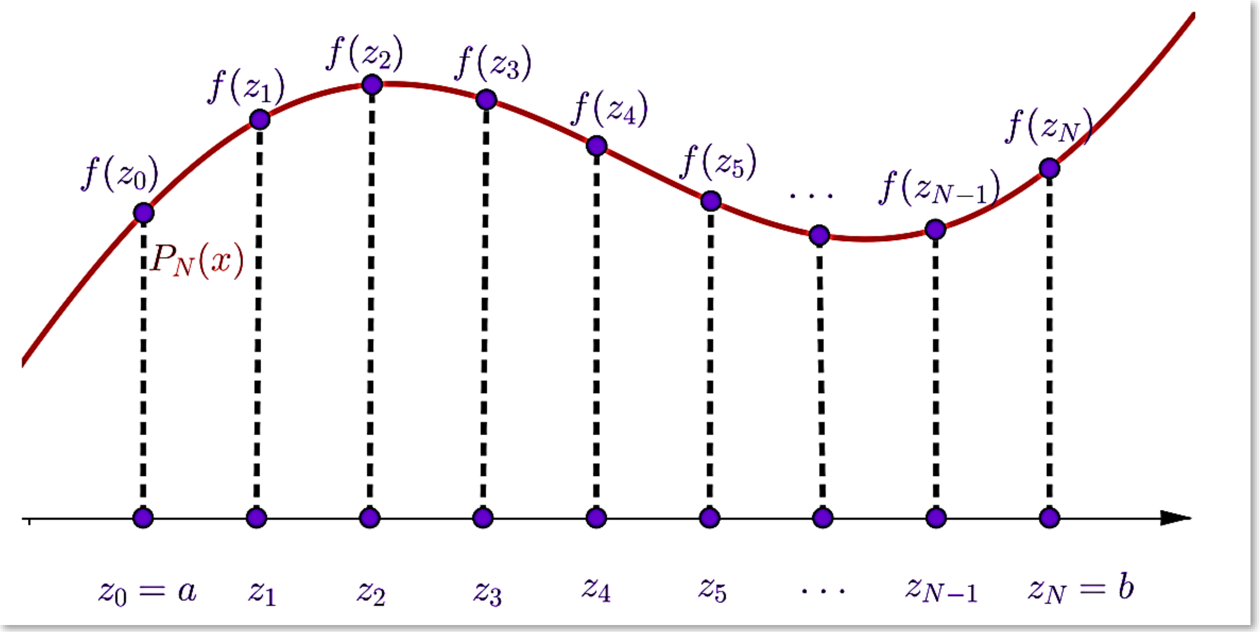
```python
print('Left Rectangle rule gives : ', I_rectL)
print('with error : ', err_rectL, '\n')
```

III. Newton-Cotes quadrature rules

Numerical quadrature:

- *Newton-Cotes rules are based on the approximation of the integrant $f(x)$ by an interpolating polynomial. The integral of f is then approximated by the integral of the interpolating polynomial, which we can compute exactly.*

- *Using $N+1$ equally-spaced points : $a = x_0 < x_1 < \ldots < x_N = b$, the quadrature nodes are defined as $z_i \equiv x_i$ for $i = 0, 1, \ldots, N$.*



- *We evaluate $f(x)$ at these points and take the interpolating polynomial:*

$$P_N(x) = \sum_{i=0}^{N} f(z_i).\, l_i(x)$$

III. Newton-Cotes quadrature rules

Numerical quadrature:

- The $l_i(x)$ *are known as cardinal functions used to obtain the most suitable polynomial according to the* *"Lagrange formula"* *given previously:* $P_N(x) = \sum_{i=0}^{N} f(z_i).\,l_i(x)$*. They are defined as:*

$$l_i(x) = \frac{x - x_0}{x_i - x_0}\cdot\frac{x - x_1}{x_i - x_1}\cdots\frac{x - x_{i-1}}{x_i - x_{i-1}}\cdot\frac{x - x_{i+1}}{x_i - x_{i+1}}\cdots\frac{x - x_n}{x_i - x_n} = \prod_{\substack{j=0\\ j\neq i}}^{N}\frac{x - x_i}{x_j - x_i};\, i = 0, 1, \ldots N$$

For example, if $N = 1$*, the interpolant is the straight line* $P_1(x) = y_0.\,l_0(x) + y_1.\,l_1(x)$*, where:*

$$l_0(x) = \frac{x - x_1}{x_0 - x_1};\, l_1(x) = \frac{x - x_0}{x_1 - x_0}$$

III. Newton-Cotes quadrature rules

Numerical quadrature:

- ***The quadrature formula for $I_N[f]$, which approximates $I[f]$, is defined as:***

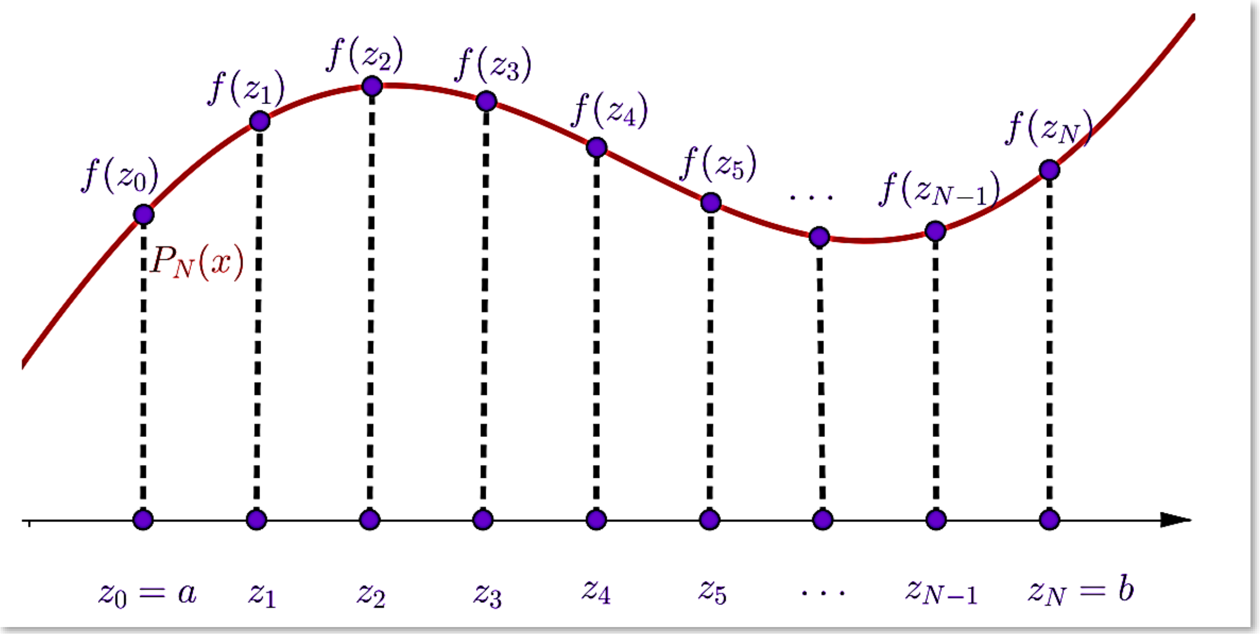
$$I_N[f] = \int_a^b P_N(x)dx = \int_a^b \sum_{i=0}^{N} f(z_i).l_i(x)\,dx = \sum_{i=0}^{N}\left[f(z_i)\int_a^b l_i(x)dx\right] = \sum_{i=0}^{N} w_i.f(z_i)$$

- ***Where the weights w_i are given as:***

$$w_i = \int_a^b l_i(x)dx$$

The general Newton-Cotes quadrature rule has the form:

$$I_N[f] = \sum_{i=0}^{N} w_i.f(z_i)$$



IV. Trapezoidal rule

Simple trapezoidal rule: *it uses linear polynomials $P_1(x)$ to approximate the function $f(x)$:*

$$l_0(x) = \frac{x - x_1}{x_0 - x_1} = -\frac{x-b}{h}; \text{ and } l_1(x) = \frac{x - x_0}{x_1 - x_0} = \frac{x-a}{h}; \text{ since } x_0 \equiv a \text{ and } x_1 \equiv b$$

Where $h = x_1 - x_0 = b - a$ is the length of the interval $[a, b]$. In this case, we have:

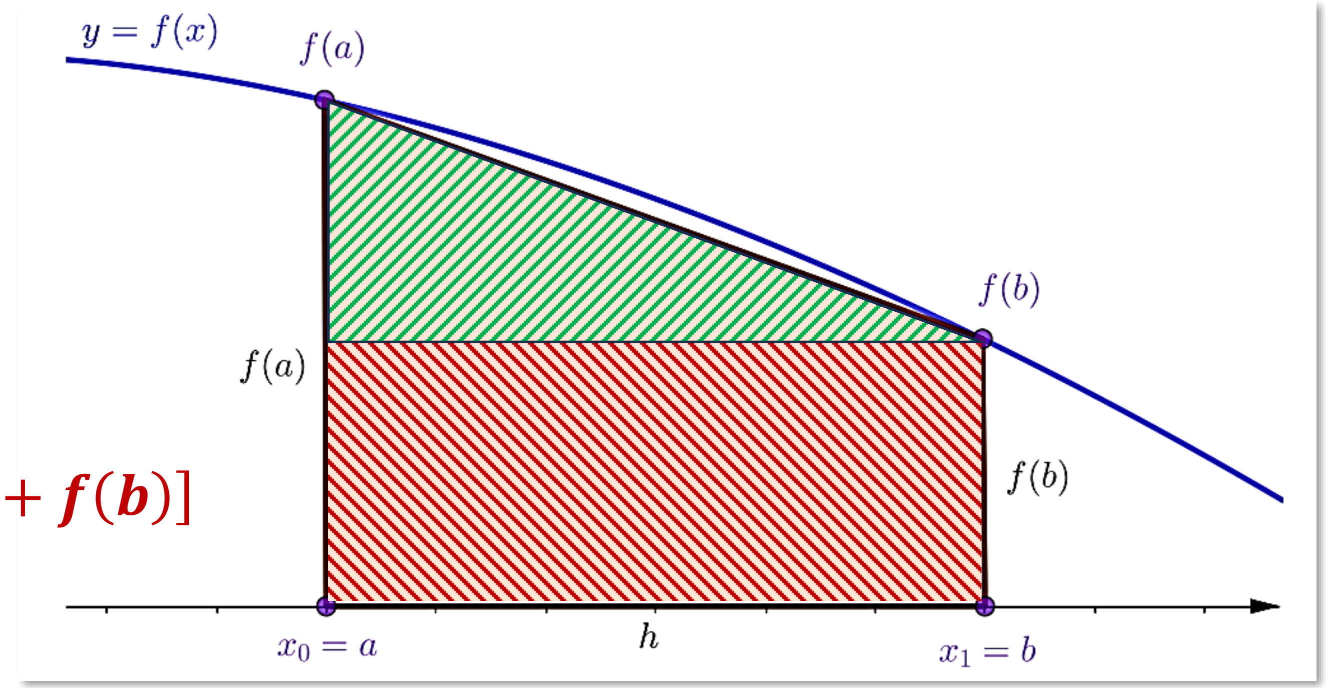
$$w_0 = \int_a^b l_0(x)dx = \frac{1}{h}\int_a^b (b-x)dx = \frac{h}{2}$$

$$w_1 = \int_a^b l_1(x)dx = \frac{1}{h}\int_a^b (x-a)dx = \frac{h}{2}$$

Substitution of the weights w_i, yields to:

$$I_1[f] = \frac{h}{2}[f(a) + f(b)] = \frac{(b-a)}{2}[f(a) + f(b)]$$

$$A_{trapz} = h.f(b) + \frac{(f(a) - f(b))}{2} \equiv I_1[f]$$



This is consistent with the geometrical representation of the integral shown in opposite figure

IV. Trapezoidal rule

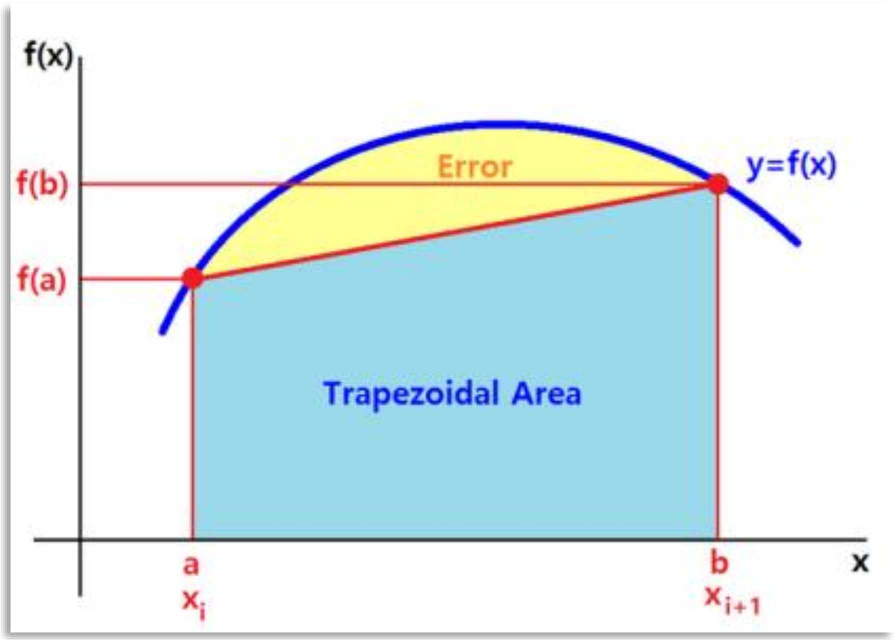
Truncation error estimation: *What about the error made when using trapezoidal rule to estimate the integration $I_1[f]$ instead of the exact one $I[f] = \int_a^b f(x)dx$? (yellow zone)*

In fact, this error could be written as: $E = I[f] - I_1[f]$, if we know the exact value of $I[f]$ on the interval $[a, b]$. In other word, what is the accuracy of this numerical integration method?

The main idea to estimate E is to use Taylor series of $f(x)$ and evaluate both $I[f]$ and $I_1[f]$, then obtain the difference $E = I[f] - I_1[f]$:

For given function $f(x)$ around a given point x_0, the Taylor series is given by:

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \frac{(x - x_0)^2}{2!}f''(x_0) + \cdots$$



IV. Trapezoidal rule

Truncation error estimation: ***Let's write and develop both*** $I[f]$ and $I_1[f]$, ***using Taylor series***

- ***A simple choice is to take*** $x_0 \equiv a$, ***thus:***

$$I[f] = \int_a^b f(x)dx \equiv \int_a^b \left[f(a) + (x-a)f'(a) + \frac{(x-a)^2}{2!} f''(a) + \cdots \right] dx$$

- ***After performing integration, one gets:***

$$I[f] = (b-a)f(a) + \frac{(b-a)^2}{2} f'(a) + \frac{(b-a)^3}{3!} f''(a) + \cdots$$

- ***On the other hand, using the fact that is possible to write*** $f(b)$ ***using Taylor series:***

$$f(b) = f(a) + (b-a)f'(a) + \frac{(b-a)^2}{2!} f''(a) + \cdots$$

- ***We can rewrite Trapezoidal rule*** $I_1[f] = \frac{(b-a)}{2}[f(a) + f(b)]$ ***as:***

$$I_1[f] = \frac{(b-a)}{2} \left[f(a) + f(a) + (b-a)f'(a) + \frac{(b-a)^2}{2!} f''(a) + \cdots \right]$$

IV. Trapezoidal rule

Truncation error estimation: *The last step consists to make the difference and eliminate similar terms:*

- *A simple choice is to take $x_0 \equiv a$, thus:*

$$
\begin{aligned}
&I[f] - I_1[f] \\
&= \left[(b-a)f(a) + \frac{(b-a)^2}{2}f'(a) + \frac{(b-a)^3}{3!}f''^{(a)} + \cdots\right] \\
&- \left[(b-a)f(a) + \frac{(b-a)^2}{2}f'^{(a)} + \frac{(b-a)^3}{4}f''(a) + \cdots\right]
\end{aligned}
$$

- *We obtain finally:*

$$E = I[f] - I_1[f] = \left[\frac{(b-a)^3}{6}f''^{(a)} + \cdots\right] - \left[\frac{(b-a)^3}{4}f''(a) + \cdots\right] = -\frac{(b-a)^3}{12}f''(a) + \cdots$$

$$E = I[f] - I_1[f] = -\frac{h^3}{12}f''(a) + O(h^4)$$

IV. Trapezoidal rule

Truncation error estimation: ***As shown, the next term in the expansion has a*** $(\boldsymbol{b}-\boldsymbol{a})^4 = h^4$ ***factor, which could be assumed negligeable. Even more, to bound the truncation error we can take some value*** $\boldsymbol{f}(\boldsymbol{\theta})$***, where:*** $\boldsymbol{a} < \boldsymbol{\theta} < \boldsymbol{b}$ ***such that:***

$$\boldsymbol{E} = \boldsymbol{I}[\boldsymbol{f}] - \boldsymbol{I}_1[\boldsymbol{f}] = -\frac{\boldsymbol{h}^3}{\mathbf{12}}\boldsymbol{f}''(\theta) + \boldsymbol{O}(\boldsymbol{h}^4)$$

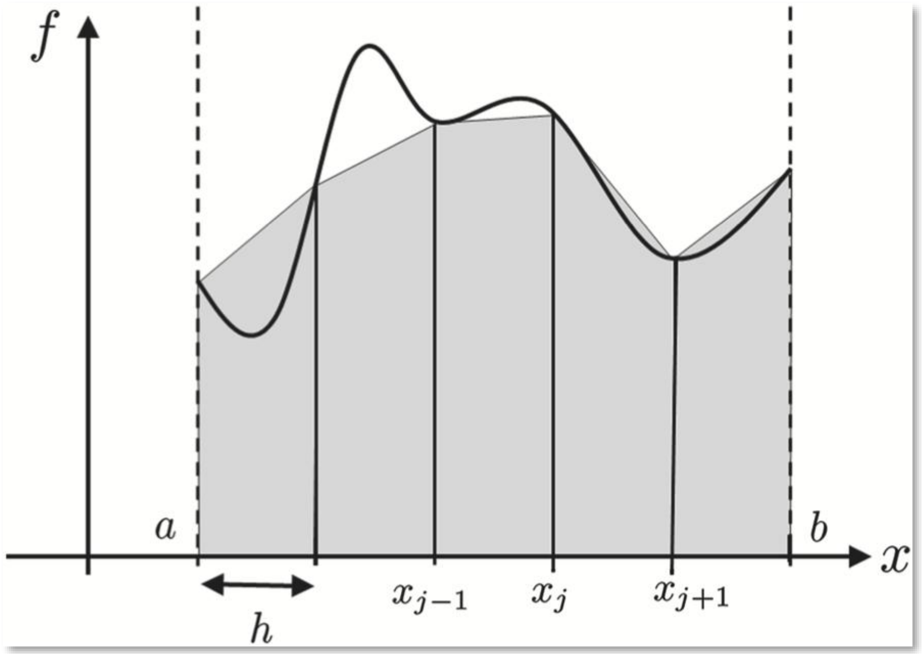
And by maximizing, the truncation error could be bounded:

$$-\frac{\boldsymbol{h}^3}{\mathbf{12}} max|\boldsymbol{f}''(\theta)| \le \boldsymbol{E} = \boldsymbol{I}[\boldsymbol{f}] - \boldsymbol{I}_1[\boldsymbol{f}] \le \frac{\boldsymbol{h}^3}{\mathbf{12}} max|\boldsymbol{f}''(\theta)|$$

IV. Trapezoidal rule

Composite trapezoidal rule: *The simple trapezoidal rule could be generalized easily when dividing the interval* $[a, b]$ *into equally spaced subintervals* $[x_i, x_{i+1}] : h = (x_{i+1} - x_i)$.

$$\int_a^b f(x)dx = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)dx$$



When applied in each subinterval, trapezoidal rule gives:

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

Where:$h = (x_{i+1} - x_i)$ *for* $i = 0, 1, \dots N - 1$

Hence, the composite trapezoidal rule can be written as:

$$I_N[f] = \sum_{i=0}^{N-1} \frac{h}{2}[\mathrm{f}(x_i) + f(x_{i+1})] = \frac{h}{2}[\mathrm{f}(x_0) + f(x_1) + \mathrm{f}(x_1) + f(x_2) + \mathrm{f}(x_2) + \cdots + f(x_N)]$$

IV. Trapezoidal rule

Composite trapezoidal rule: *It is easy to see that the numerical integration $I_N[f]$ could be simplified to:*

$$I_N[f] = \sum_{i=0}^{N-1} \frac{h}{2}[f(x_i) + f(x_{i+1})] = \frac{h}{2}[f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{N-1}) + f(x_N)]$$

$$I_N[f] = \frac{h}{2}\left[f(a) + 2\sum_{i=1}^{N-1} f(x_i) + f(b)\right]$$

Truncation error estimation of composite trapezoidal rule: *Similarly, the truncation error estimation in this case, could be generalized from the simple case to n subintervals:*

$$E_1[f] = I[f] - I_N[f] = \int_a^b f(x)dx - \frac{h}{2}\left[f(a) + 2\sum_{i=1}^{N-1} f(x_i) + f(b)\right] = -\frac{h^3}{12}\sum_{i=0}^{N-1} f''(\theta_i)$$

IV. Trapezoidal rule

Truncation error estimation of composite trapezoidal rule: *for uniform function, it is possible to find θ such that:*

$$\sum_{i=0}^{N-1} c_i . f''(\theta_i) \equiv f(\theta)\left[\sum_{i=0}^{N-1} c_i\right] = N . f''(\theta);\ since\ c_i = 1$$

This will allow us to write the estimation of error truncation as follows:

$$E_1[f] = I[f] - I_N[f] = -\frac{h^3}{12}\sum_{i=0}^{N-1} f''^{(\theta_i)} = -\frac{h^3}{12} N . f''^{(\theta)} = -\frac{(b-a)h^2}{12} f''^{(\theta)};$$

$$using\ N . h = b - a$$

Finally, it is possible to bound the truncation error by:

$$-\frac{(b-a)h^2}{12} max|f''(\theta)| \leq E_1[f] = I[f] - I_1[f] \leq \frac{(b-a)h^2}{12} max|f''(\theta)|$$

IV. Trapezoidal rule

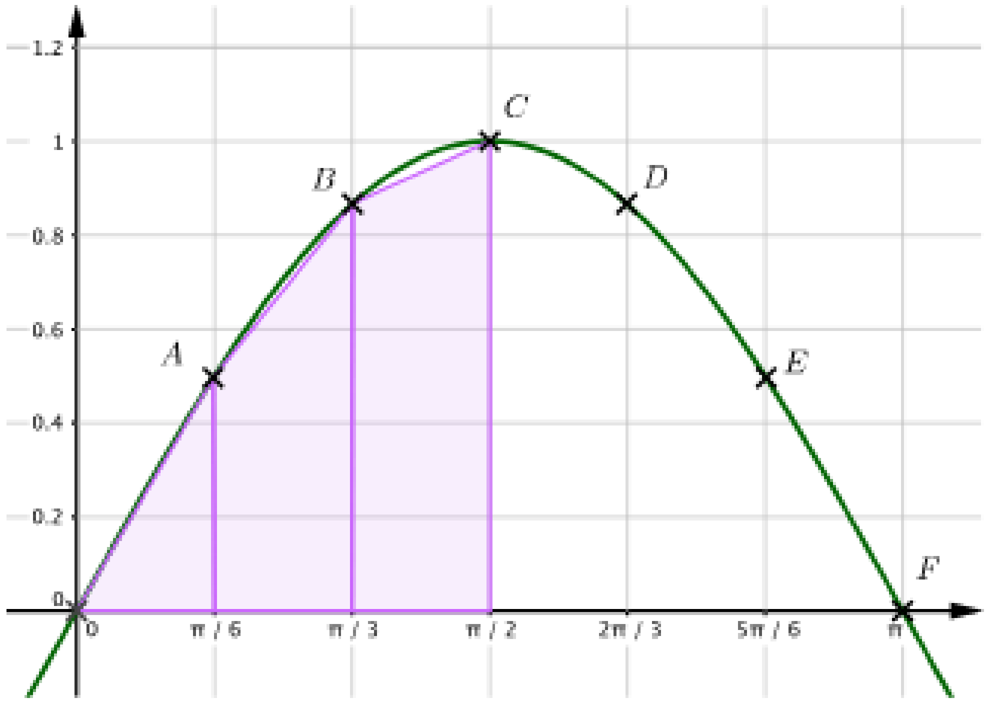
Application 02:

Using the previous code, add the Trapezoid Rule to same integration:

$$\int_0^{\pi} sin(x)dx$$

Compare this value to the exact value of 2.

Compare the found truncation error with the maximal bound




```
I_trap = (h/2)*(f[0] + 2 * sum(f[1:n]) + f[n])
```

IV. Trapezoidal rule

Application 03:

Duplicate the same code to evaluate:

$$\int_0^{\pi/2} \cos(x)dx$$

With different methods (Rectangle, midpoint, and Trapezoidal rules).

Compare this value to the exact value of 1, and comment the findings upon truncation errors for each rule.

Is there any differences with the previous case of $\int_0^{\pi} \sin(x)dx$***? Why ?***

Miscellaneous:

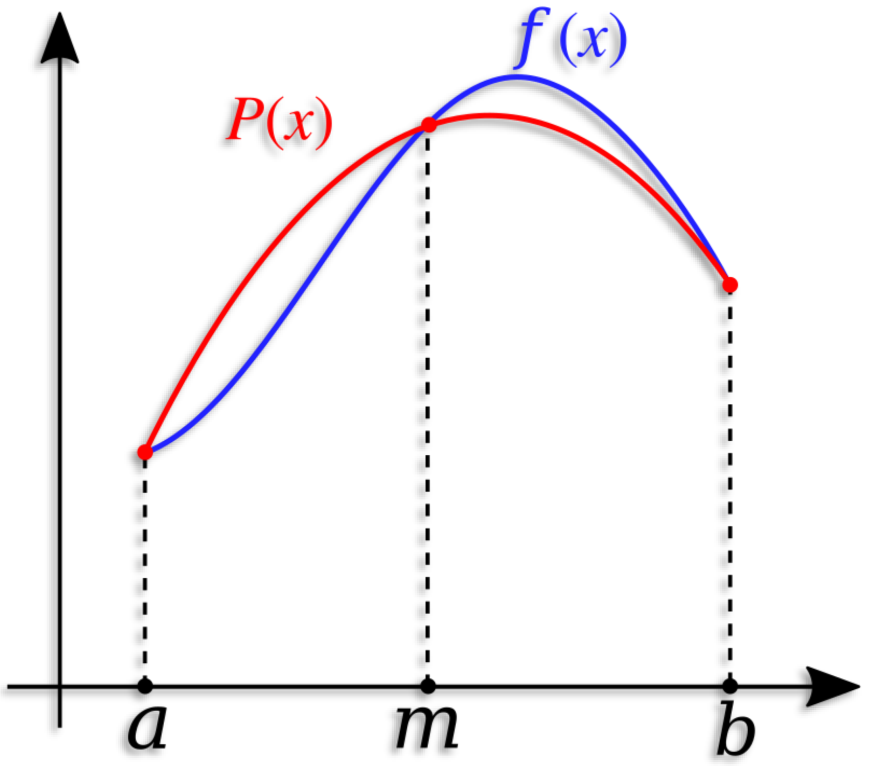
- *Any numerical rule with large width h will provide inaccurate results, that's why composite rules are necessary with N large enough.*
- *For symmetrical functions, it could be seen that Riemann integration and Trapezoidal rule are equivalent;*
- *The difference could be seen in non symmetrical functions, where trapezoidal rule gives better results*
- *The mid point rule provide lower truncation error estimation, which an absolute value being the half of the trapezoidal one (by using the same procedure) :*

$$E_0[f] = \mathrm{I}[f] - I_0[f] = \frac{h^3}{24}\sum_{i=0}^{N-1} f''^{(\theta_i)} = \frac{h^3}{24}N.f''^{(\theta)} = \frac{(b-a)h^2}{24}f''^{(\theta)};$$

V. Simpson's rule

Simpson's rule (named after Thomas Simpson (1710–1761) and also known as the $1/3$ rule) is a Newton-Cotes rule for the quadratic interpolation polynomial $P_2(x)$. For Simpson's rule we have $N = 2$ using 3 nodes. Consider two consecutive subintervals, $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$.

Simpson's Rule approximates the area under $f(x)$ over these two subintervals by fitting a quadratic polynomial through the points $(x_{i-1}, f(x_{i-1})), (x_i, f(x_i))$, and $(x_{i+1}, f(x_{i+1}))$, which is a unique polynomial, and then integrating the quadratic exactly.



V. Simpson's rule

Simple Simpson's rule: *it uses nonlinear polynomials $P_2(x)$ to approximate the function $f(x)$.*

- *We take $x_0 = a, x_1 = (a+b)/2$ and $x_2 = b$, and also $z_i = x_i$ for $i = 0, 1, 2$.*
- *The length of each interval $[x_i, x_{i+1}]$ is $h = (b - a)/2$.*
- *Accordingly, The quadratic Lagrange cardinal functions are given as follows:*

$$l_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)};\ l_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)};\ l_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

- *It is possible to perform integration of $l_i(x)$ by introducing new variable $\theta = x - x_1$ such as:*

$$\theta_0 = 0 - h = -h;\ \theta_1 = h - h = 0;\ \theta_2 = 2h - h = h$$

- *Consequently, the quadrature weights become ($d\theta = d(x - x_1) = dx$):*

$$w_i = \int_a^b l_i(x)dx = \int_{-h}^{h} l_i(\theta)d\theta\,;\ for\ i = 0, 1, 2$$

$$l_0(\theta) = \frac{(\theta-0)(\theta-h)}{-h(-2h)} = \frac{\theta(\theta-h)}{2h^2};\ l_1(\theta) = \frac{(\theta+h)(\theta-h)}{-h^2};\ l_1(\theta) = \frac{\theta(\theta+h)}{2h^2}$$

V. Simpson's rule

Simple Simpson's rule:

- ***Finally, the obtained quadrature weights are given by:***

$$w_0 = \int_{-h}^{h} \frac{\theta(\theta-h)}{2h^2} d\xi = \frac{1}{2h^2}\int_{-h}^{h}(\theta^2-\theta h)d\theta = \frac{h}{3}$$

$$w_1 = \int_{-h}^{h} \frac{(\theta+h)(\theta-h)}{-h^2} d\theta = -\frac{1}{h^2}\int_{-h}^{h}(\theta^2-h^2)d\theta = \frac{4h}{3}$$

$$w_2 = \int_{-h}^{h} \frac{\theta(\theta+h)}{2h^2} d\theta = \frac{1}{2h^2}\int_{-h}^{h}(\theta^2+\theta h)d\theta = \frac{h}{3}$$

Thus, Simpson's rule for the approximation of an integral is:

$$I_2[f] = \sum_{i=0}^{2} w_i . f(z_i) = \frac{h}{3}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right]$$

V. Simpson's rule

Error of Simple Simpson's rule:

- *Similar to the trapezoidal rule error, the error of Simpson's rule is given by:*

$$E_2[f] = I[f] - I_2[f] = \int_a^b [f(x) - P_2(x)]dx$$

Where: P_2 is the unique interpolating polynomial of the function $f(x)$ at the nodes $x_0 = a, x_1$ and x_2 =b. Thus, we can obtain the following expression:

$$E_2[f] = \frac{-1}{24}\int_a^b (x-a)\left(x - \frac{a+b}{2}\right)^2 (b-x).f^{(4)}(\theta)dx$$

Reduced to the following expression after integrating:

$$E_2[f] = -\frac{(b-a)^5}{2^4.180}f^{(4)}(\theta) = -\frac{(b-a)h^4}{180}f^{(4)}(\theta)$$

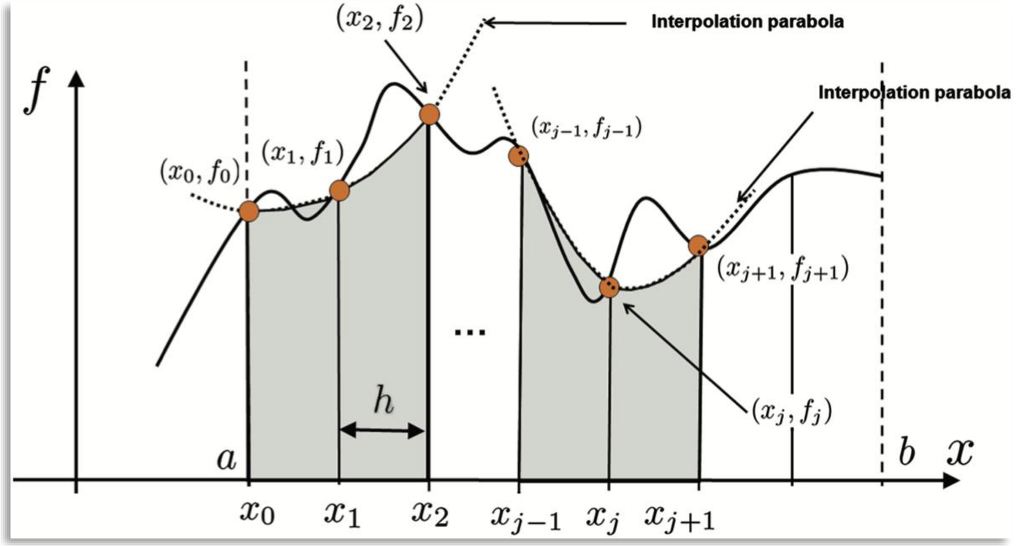
$$-\frac{(b-a)h^4}{180}max\left|f^{(4)}(\theta)\right| \leq E_2[f] \leq \frac{(b-a)h^4}{180}max\left|f^{(4)}(\theta)\right|$$

V. Simpson's rule

Composite Simpson's rule: *Let* $N = 2m$ *be an even number, and* $h = (b-a)/N$. *The nodes could be defined as* $x_i = a + ih; i = 0, 1, \dots N$ *since in Simpson's rule we have* **3** *nodes on each subinterval. That's why we consider intervals* $[x_{2k}, x_{2k+2}]; k = 0, 1, \dots, \frac{N}{2} - 1$.

Consequently, the integral $I[f]$ *could be written as:*

$$\int_a^b f(x)dx = \sum_{k=2}^{\frac{N}{2}-1} \int_{x_{2k}}^{x_{2k+2}} f(x)dx$$



Applied on subintervals $[x_0, x_2], [x_2, x_4], \dots, [x_{N-2}, x_N]$ *to obtain:*

$$\int_{x_{2K}}^{x_{2k+2}} f(x)dx \approx \frac{h}{3}[f(x_{2K}) + 4f(x_{2K+1}) + f(x_{2K+2})]$$

V. Simpson's rule

Composite Simpson's rule: *Then, the composite Simpson's rule is given by the sum:*

$$I[f]=\int_a^b f(x)dx=\sum_{k=2}^{\frac{N}{2}-1}\int_{x_{2k}}^{x_{2k+2}} f(x)dx \approx \sum_{k=2}^{\frac{N}{2}-1}\frac{h}{3}[f(x_{2K})+4f(x_{2K+1})+f(x_{2K+2})]$$

$$I_N[f]=\frac{h}{3}[f(x_0)+4f(x_1)+2f(x_2)++4f(x_3)+\cdots+2f(x_{N-2})+4f(x_{N-1})+f(x_N)]$$

*One can notice that term with even indices are multiplied by **2**, while odd indices are multiplied by **4**:*

$$\int_a^b f(x)dx \approx I_N[f]=\frac{h}{3}\left[f(x_0)+4\sum_{i=1}^{N-1}f(x_{2i-1})+2\sum_{i=1}^{N-1}f(x_{2i})+f(x_N)\right]$$

$$\int_a^b f(x)dx \approx I_N[f]=\frac{h}{3}\left[f(x_0)+4\sum_{odd}^{N-1}f(x_i)+2\sum_{even}^{N-1}f(x_i)+f(x_N)\right]$$

V. Simpson’s rule

Error of composite Simpson’s rule: ***In similar way, we can estimate the error on the composite Simpson’s rule by adding each term contribution:***

$$E_N[f] = \int_a^b f(x)dx - I_N[f] = -\frac{(b-a)^5}{N^4.180} f^{(4)}(\theta) = -\frac{(b-a)h^4}{180} f^{(4)}(\theta);\ \theta \in [a,b]$$

$$-\frac{(b-a)h^4}{180} max\left|f^{(4)}(\theta)\right| \leq E_N[f] \leq \frac{(b-a)h^4}{180} max\left|f^{(4)}(\theta)\right|$$

V. Simpson's rule

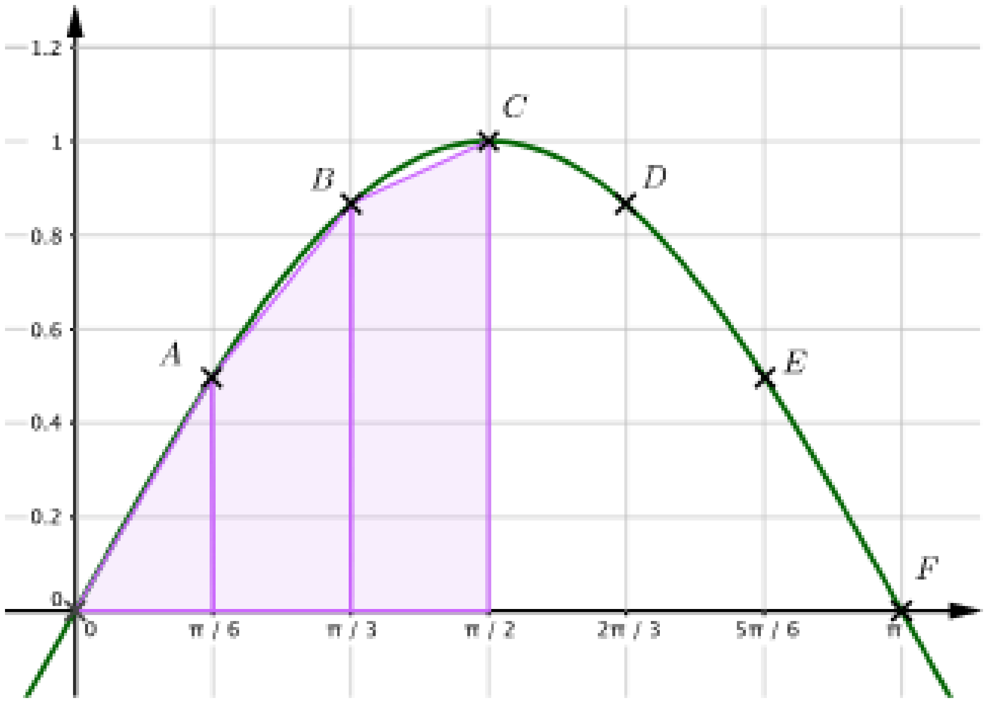
Application 04:

Using the previous code, add the Simpson's rule to same integration:

$$\int_0^{\pi} sin(x)dx$$

Compare this value to the exact value of 2.

Compare the found truncation error with the maximal bound




```
I_simp = (h/3) * (f[0] + + 4*sum(f[1:n:2]) + 2*sum(f[2:n:2]) + f[n])
```

V. Simpson's rule

Application 05:

Duplicate the same code to evaluate:

$$\int_0^{\pi/2} cos(x)dx$$

With different methods (Riemann, midpoint, Trapezoidal rule and Simpson's rule).

Compare this value to the exact value of 1, and comment the findings upon truncation errors for each rule.

Synthesis table

Designation	*Expression*	*Error*	*Composite Expression*	*Error*
Rectangle rule	$h.f(\alpha);\ \alpha \in [a,b]$	$\frac{(b-a)^2}{2}.f'(\theta)$	$h\sum_{i=1}^{N-1} f(\alpha_i)$	$\frac{h(b-a)}{2}.f''^{(\theta)}$
Midpoint rule	$h.f\left(\frac{a+b}{2}\right)$	$\frac{(b-a)^3}{24}.f''(\theta)$	$h\sum_{i=1}^{N-1} f\left(\frac{x_i+x_{i+1}}{2}\right)$	$\frac{h^2(b-a)}{24}.f''^{(\theta)}$
Trapezoidal rule	$\frac{h}{2}[f(a)+f(b)]$	$-\frac{(b-a)^3}{12}f''(\theta)$	$\frac{h}{2}\left[f(a)+2\sum_{i=1}^{N-1} f(x_i)+f(b)\right]$	$-\frac{(b-a)h^2}{12}f''^{(\theta)}$
Simpson's rule	$\frac{h}{3}\left[f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right]$	$-\frac{(b-a)^5}{2^4.180}f^{(4)}(\theta)$	$\frac{h}{3}\left[f(x_0)+4\sum_{odd}^{N-1} f(x_i) + 2\sum_{even}^{N-1} f(x_i)+f(x_N)\right]$	$-\frac{(b-a)h^4}{180}f^{(4)}(\theta)$