# Lecture 1: Numerical Integration (I)

(Adapted from slides by Uriel Morzan)

Gerald E. Fux
(gfux@ictp.it)

11. Oct. 2023

# Numerical Integration - Introduction

**Numerical integration is ...**

- ... about calculating the numerical value of a definite Integral of some function $f(x)$, such as (in 1 dimension):

$$\int_a^b f(x)\,\mathrm{d}x.$$

**Numerical integration is not necessary if ...**

- ... we know an analytical expression for $f(x)$ and a primitive $F(x) = \int f(x)\mathrm{d}x$. Then

$$\int_a^b f(x)\,\mathrm{d}x = F(b) - F(a).$$

For example:

$$\int_0^t x^2 \sin(x)\,\mathrm{d}x = -2 + (2 - t^2)\cos(t) + 2t\sin(t).$$

# Numerical Integration - Introduction

**We need numerical integration because ...**

- ... for many functions $f(x)$ the primitive function $F(x)$ is either
  - unknown, or
  - not “analytical” (i.e. expressible as an elementary function), such as $\int e^{-x^2}\,\mathrm{d}x$.
- ... sometimes we don’t even know an expression for $f(x)$, but it is itself the result of some numerical computation.

For example:

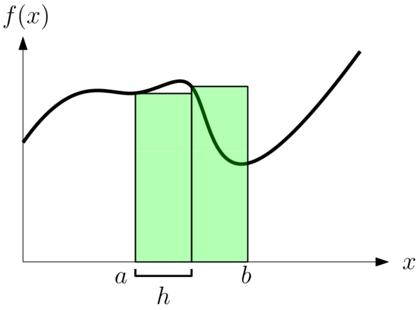
$$\int_{-\infty}^{1} e^{-x^2}\,\mathrm{d}x \simeq 1.63305$$

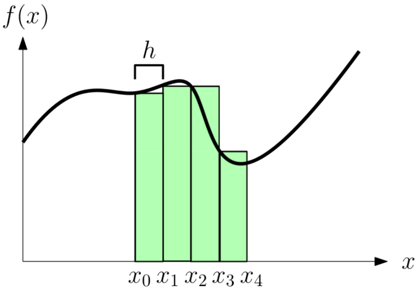
# Numerical Integration Through the Integral Sum

Consider the definition of integrals via **the integral sum**:

$$\int_a^b f(x)\,\mathrm{d}x \equiv \lim_{N\to\infty}\left[\sum_{k=0}^{N-1} f(x_k)\cdot h\right] \quad \text{with} \quad \begin{cases} h \equiv \frac{b-a}{N} \\ x_k \equiv a + k\cdot h \\ k = 0,\ldots,N-1 \end{cases}$$

This is an approximation for finite $N$, but improves for growing $N$ and is exact for $N \to \infty$.



# Numerical Method: Left Riemann Sum

Define $I_L(N)$ as the approximated integral:

$$I_L(N) \equiv \sum_{k=0}^{N-1} f(x_k) \cdot h$$

with

$$h \equiv \frac{b-a}{N}$$
$$x_k \equiv a + k \cdot h$$
$$k = 0, \ldots, N-1.$$

## Algorithm: Left Riemann Sum

**Input**: function $f(x)$; boundaries $a$ and $b$; small threshold $\epsilon$.

1. Set $N$ to some initial value, e.g. $N := 32$
2. Compute $I_{\mathrm{new}} := I_L(N)$
3. Loop:
   - Increase $N$, e.g. $N := 2N$
   - Set $I_{\mathrm{old}} := I_{\mathrm{new}}$
   - Compute $I_{\mathrm{new}} := I_L(N)$
   - If $|I_{\mathrm{new}} - I_{\mathrm{old}}| < \epsilon$ then exit the loop.

**Output**: $I_{\mathrm{new}}$, which is an approximate value of $\int_a^b f(x)\,\mathrm{d}x$

# Numerical Method: Left Riemann Sum

## Error scaling of the left Riemann sum

For large enough $N$ the error decreases faster or as fast as $(1/N)^1$, i.e.

$$\int_a^b f(x)\,\mathrm{d}x = I_L(N) + \mathcal{O}\left(\frac{1}{N}\right)$$

# Assignment 7

Write a FORTRAN program that computes $\int_a^b f(x)\,\mathrm{d}x$ for $f(x) = e^x$ using the left Riemann sum method:

- Write a function that takes the bounds $a$ and $b$, and the desired precision $\epsilon$.
- The function should integrate with the left Riemann sum, increasing N until the precision is achieved.
- The function should output the result at each step together with the current value for $N$.
- Test the function by calculating $\int_0^1 e^x\,\mathrm{d}x$ with error threshold $10^{-5}$ in the main program and print the result.
- Submit your code as `<surname>-assignment-7.f90` to gfux@ictp.it before the next lesson.

**Hints:**

- Create separate functions for $f(x)$ and the integration.
- Increase N significantly at each step (e.g., $N \leftarrow 2N$).
- No arrays are required.