



UNIVERSIDAD POLITÉCNICA  
DE AGUASCALIENTES

Universidad Politécnica de Aguascalientes  
Ingeniería en Sistemas Computacionales

Materia: Matemáticas para ingeniería II  
Docente: Isaac Vázquez Mendoza

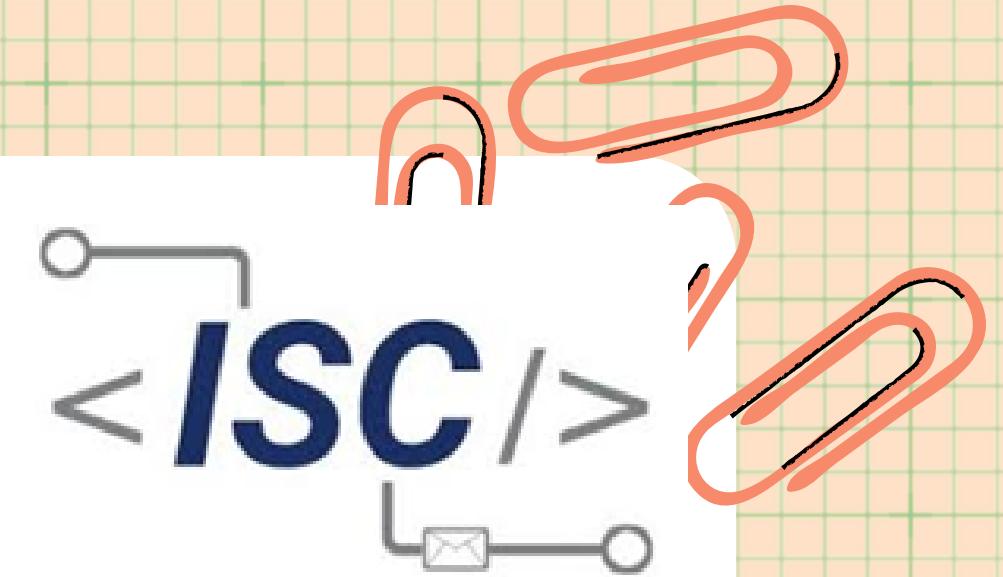
Equipo 4

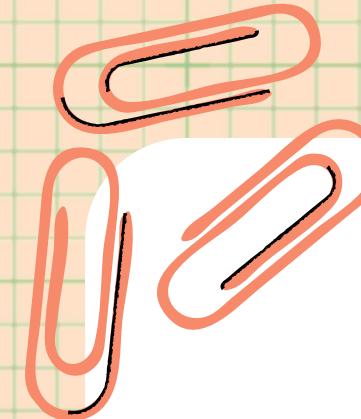
## Método Euler mejorado

Integrantes:

Vázquez Ríos Juan Ramon UP230190  
Olmedo Cruz Danna Giselle UP230183  
Surdez Espeleta Andrea Mariana UP230188  
Martin Cornejo Diego Ernesto UP230346  
López Nava Alberto Misael UP239756

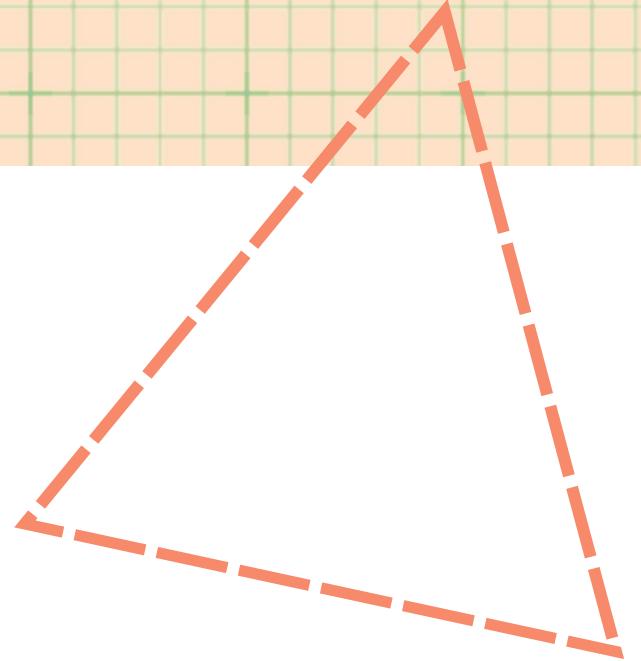
Grado y Grupo: ISC06A





# Introducción

---



El método de Euler mejorado es una técnica numérica para aproximar soluciones de ecuaciones diferenciales. Es una versión más precisa del método de Euler simple, porque en vez de usar solo la pendiente inicial, promedia dos pendientes: la del punto inicial y una pendiente estimada al final del intervalo.

Este promedio reduce el error y hace que el método sea más exacto y estable, manteniendo al mismo tiempo un procedimiento sencillo de aplicar.

# Fórmula

$$y_{n+1} = y_n + \frac{h}{2} [f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n))]$$

- Tienes un punto  $(x_n, y_n)$  de la solución.
- Calculas la pendiente usando la ecuación diferencial:  
 $K_1 = f(x_n, y_n)$
- Estimación del siguiente valor:  
 $y_{pred} = y_n + hK_1$
- obtienes un punto aproximado:  
 $(x_n + h, y_{pred})$
- Calculas la pendiente en el punto predicho:  
 $K_2 = f(x_n + h, y_{pred})$
- Promedio de pendientes  
 $(K_1 + K_2)/2$
- Finalmente, usas ese promedio para obtener:

$$y_{n+1} = y_n + h \left( \frac{k_1 + k_2}{2} \right)$$

## Dónde:

$x_n$ : valor actual de variable independiente (eje x).

$y_n$ : valor aproximado de la solución

$h$ : tamaño del paso

$f(x_n, y_n)$ : ecuación diferencial.

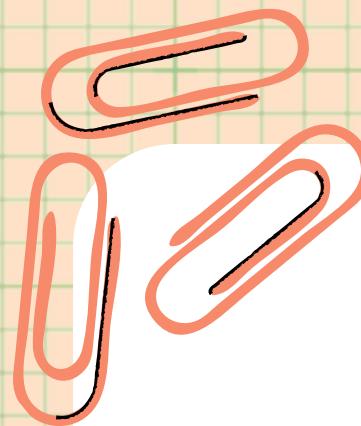
$K_1$ : pendiente inicial (euler simple)

$y_{pred}$ : valor predicho usando euler simple

$K_2$ : valor de la pendiente final

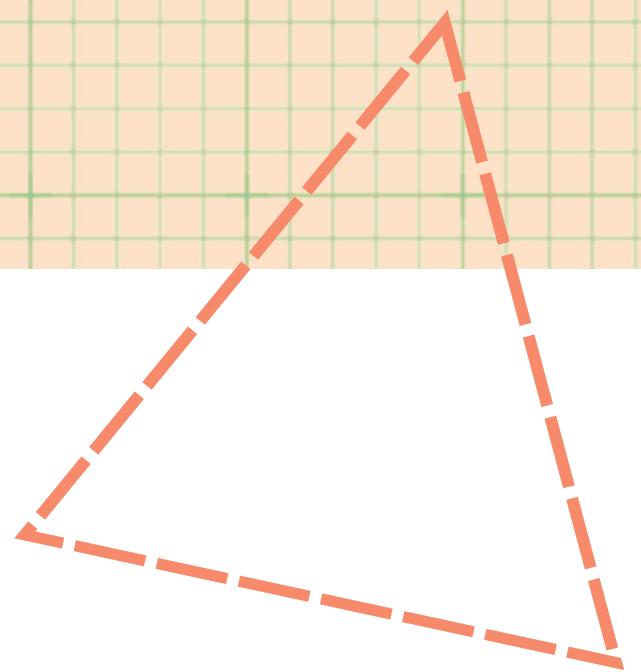
$(K_1 + K_2)/2$ : Promedio de las pendientes

$y_{n+1}$ : Nuevo valor aproximado1



## Ejemplo

- - - - -



$$y_{n+1} = y_n + \frac{h}{2} [f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n))]$$

$$y' = x + y, \quad y(0) = 1, \quad h = 0.01$$

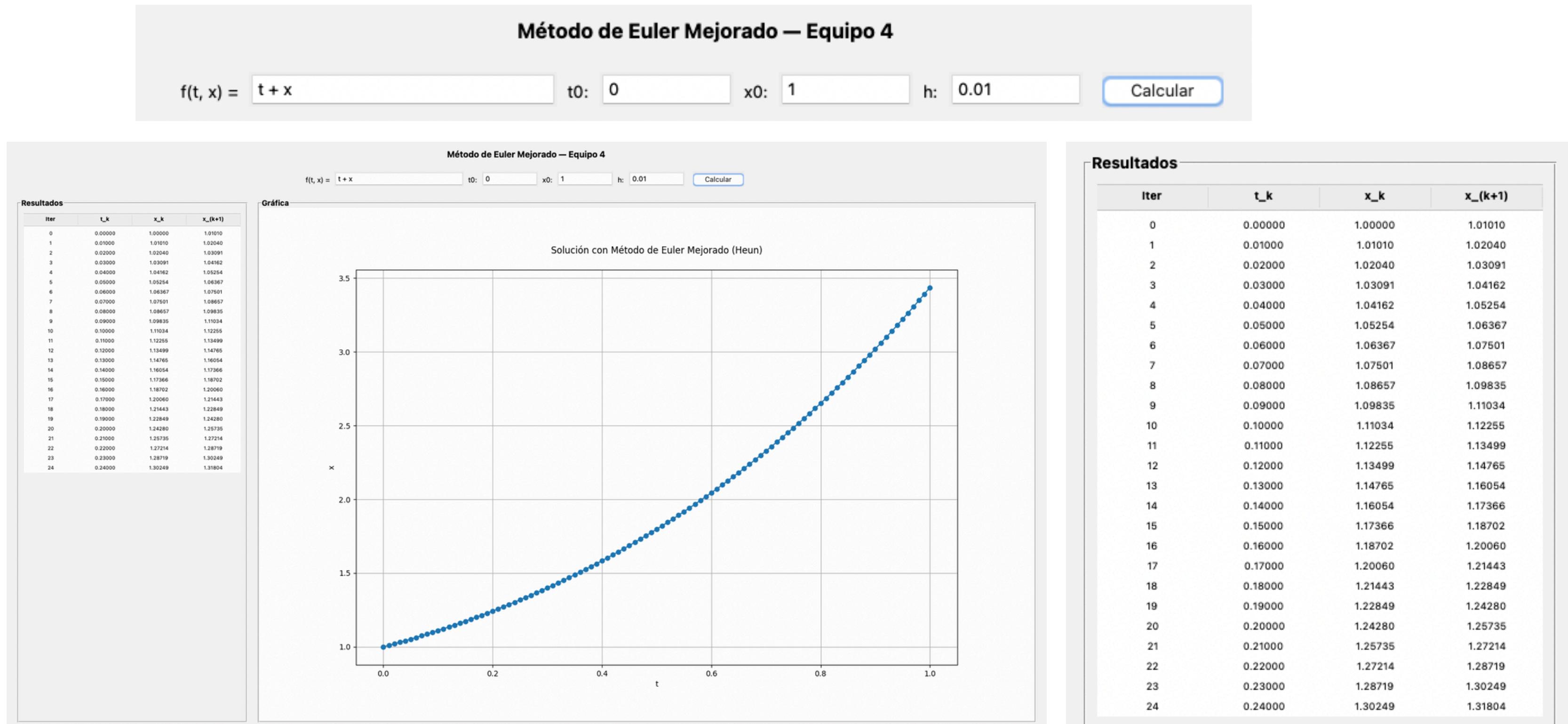
$$k_1 = f(0, 1) = 1$$

$$y_{pred} = 1 + 0.01(1) = 1.01$$

$$k_2 = f(0.01, 1.01) = 1.02$$

$$y_1 = 1 + 0.01\left(\frac{1 + 1.02}{2}\right) = 1.0101$$

# Explicación del programa



# Explicación del código

```
import sympy as sp
import numpy as np
import tkinter as tk from tkinter import ttk
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
```

Es importante tener instaladas esas librerías ya que hacen que el programa tenga un funcionamiento adecuado, de lo contrario no reconocería algunas funciones.

**Sympy:** Ayuda al programa a reconocer las expresiones algebraicas.

**Numpy:** Ayuda a hacer los cálculos de manera mas aproximada.

**Multiplot:** Sirve para mostrar varias gráficas en una misma ventana.

```
def euler_method():

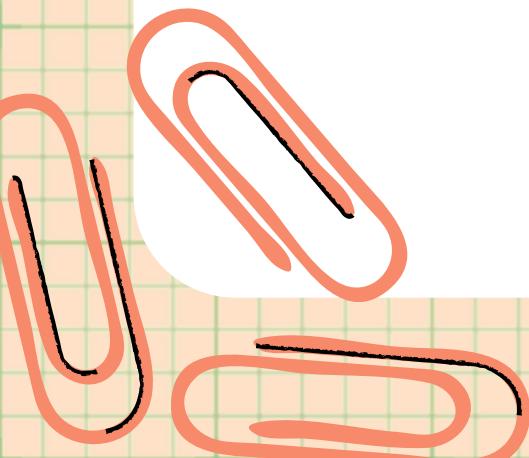
    # LIMPIA TABLA Y GRÁFICA
    tabla.delete(*tabla.get_children())
    fig.clear()

    # DEFINIENDO LOS SÍMBOLOS
    t = sp.Symbol('t')
    x = sp.Symbol('x')
```

Metodo euler:

Borra todo el contenido actual de la tabla.

**sp.Symbol:** Sirve para declarar una variable simbólica, para que reconozca la función que escribe el usuario.



```

try:
    f_str = entry_funcion.get()
    f_tx = sp.sympify(f_str)

    t0 = float(entry_x0.get())
    x0 = float(entry_y0.get())
    h = float(entry_h.get())
    tf = x0

    if h == 0:
        raise ValueError("El paso 'h' no puede ser cero.")

    # TRADUCTOR DE PYTHON EN POCAS PALABRAS
    f = sp.lambdify((t, x), f_tx, "numpy")

    # LISTAS PARA GRAFICAR
    ts = [t0]
    xs = [x0]

    tK_ = t0
    xK = x0
    K = 0

```

1. Obtiene el texto escrito en el campo (la función).
2. Convierte el texto en una expresión simbólica.
3. Valores iniciales
4. Línea de condición "if": evita que h sea cero y genere loops infinitos.
5. `lambdify`: convierte la expresión simbólica a una función normal de python.
6. Listas para graficar.
7. Inicializa las variables

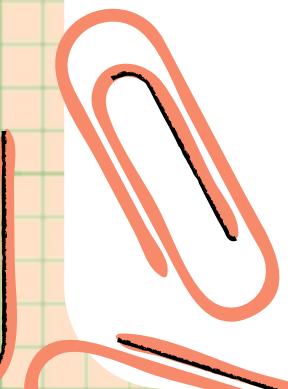
```

while (h > 0 and tK_ < tf) or (h < 0 and tK_ > tf):
    # Euler mejorado (Heun)
    K1 = f(tK_, xK)
    K2 = f(tK_ + h, xK + h * K1)
    x_next = xK + (h/2) * (K1 + K2)

    t_next = tK_ + h

```

1. El ciclo indica que se va a hacer esa acción mientras el tiempo actual esté dentro del intervalo de integración del paso "h" y el límite final "tf".
2. Acepta que se desplace hacia adelante o atrás según sea el caso.
3. `K1 = f(tK_, xK)`: Calcula la pendiente inicial en el punto actual.
4. `K2 = f(tK_ + h, xK + h * K1)`: Calcula la pendiente en el tiempo estimado usando euler.
5. `x_next = xK + (h/2) * (K1 + K2)`: Promedia ambas pendientes y avanza el valor de `x`.
6. `t_next = tK_ + h`: Avanza el tiempo al siguiente paso.



```

tabla.insert("", "end", values=[  

    K,  

    f"{{tK_:.5f}",  

    f"{{xK:.5f}",  

    f"{{x_next:.5f}"  

])  
  

ts.append(t_next)  

xs.append(x_next)  
  

tK_ = t_next  

xK = x_next  

K += 1

```

```

ax = fig.add_subplot(111)
ax.plot(ts, xs, marker='o', linestyle='-', label="Euler Mejorado")
ax.set_title("Solución con Método de Euler Mejorado (Heun)\n",  

            fontsize=12)
ax.set_xlabel("t")
ax.set_ylabel("x")
ax.grid(True)
canvas.draw()

except Exception as e:  

    tabla.insert("", "end", values=["Error", "", "", str(e)])

```

Agrega una nueva fila a la tabla  
K: número de iteración

tK: valor actual de  $t$

xK: valor actual de  $x(t)$

x\_next: muestra el nuevo valor de  $x(t+h)$   
(Euler mejorado)

5f: indica que los resultados se mostraran con 5 decimales.

tK\_=t\_next: mueve "t" al siguiente punto

xK = x\_next : actualiza "x" al nuevo valor calculado.

xK = x\_next: aumenta el número de iteración.

1. Agrega un eje fila, col, y posición en una gráfica
2. Valores de  $t$ ,  $x(t)$ , dibuja los círculos, los une con una línea y agrega un nombre a la curva.
3. Agrega un título a la gráfica.
4. agrega etiquetas, muestra cuadrícula.
5. Dibuja la gráfica en la ventana tkinter.
6. Si hay algún errores muestra ERROR y el mensaje.

# Conclusión

---

El método de Euler mejorado corrige las limitaciones del Euler simple mediante un esquema de aproximación más confiable, logrando mayor estabilidad y precisión en la resolución de ecuaciones diferenciales. Al ser de segundo orden, reduce significativamente el error global sin alcanzar la complejidad de métodos superiores como Runge-Kutta. Además, al integrar la solución plasmada en una gráfica facilita la representación de los resultados, lo que refuerza la interpretación y análisis de los modelos dinámicos.