UNIVERSIDAD POLITÉCNICA DE AGUASCALIENTES

Método de Euler

Solución Numérica de Ecuaciones Diferenciales

Equipo 4

Integrantes:

Vázquez Ríos Juan Ramón 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

Análisis y Simulación en

Python

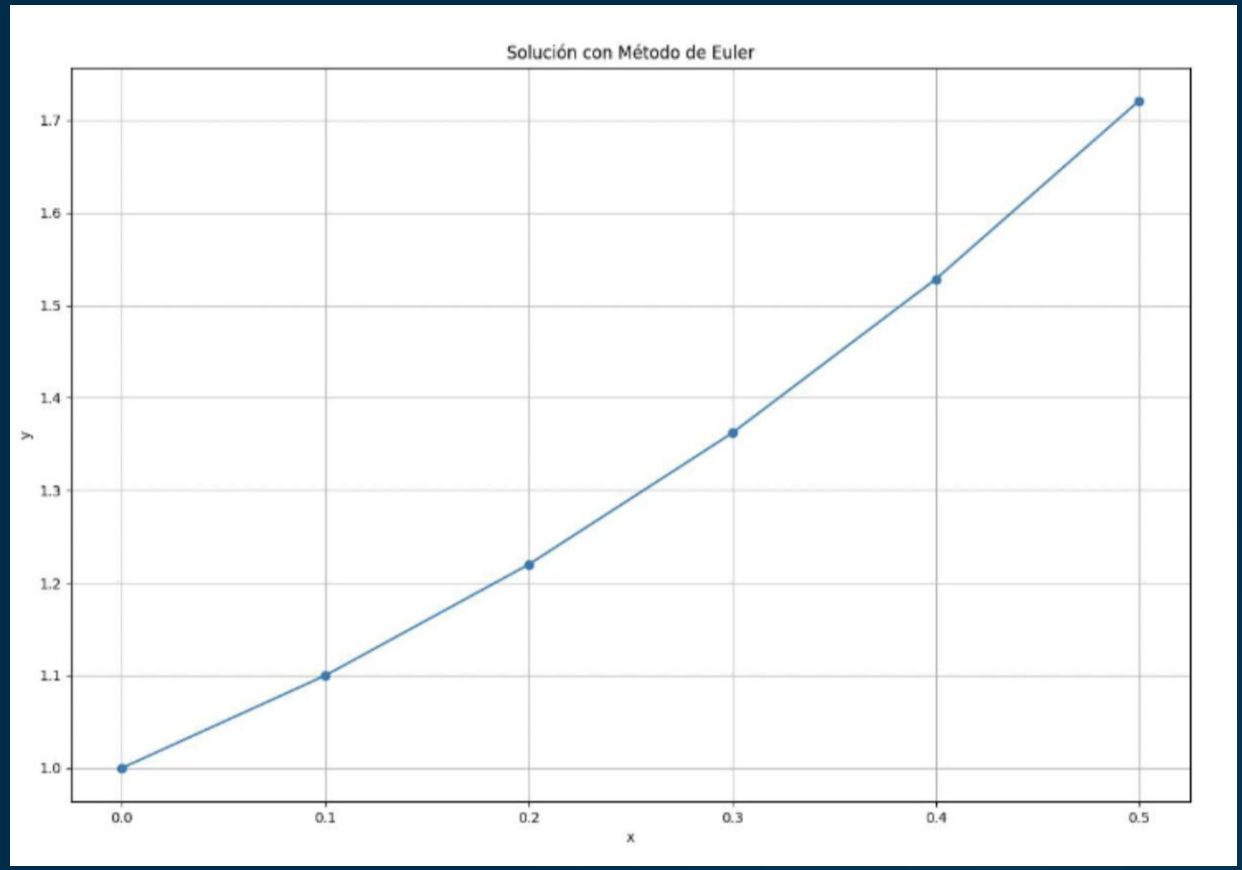
¿Qué es el Método de Euler?

Es el procedimiento numérico más básico para resolver problemas de valor inicial en ecuaciones diferenciales ordinarias.

Concepto Clave:

Utiliza la pendiente (derivada) en un punto conocido para proyectar una **línea recta** y aproximar el siguiente valor de la función.

Aunque simple, ilustra perfectamente cómo podemos simular el cambio de variables continuas (como ***t*** y ***x***) usando pasos discretos.



Explicación Matemática

La Ecuación Diferencial

Partimos de una ecuación de la forma:

$$\frac{dx}{dt} = f(t, x)$$

La Fórmula Iterativa

Para encontrar el siguiente punto:

$$x_{k+1} = x_k + h \cdot f(t_k, x_k)$$

Definición de Variables

t, x Variables independiente y dependiente.

h Paso de integración (incremento en t).

f(t,x) La pendiente evaluada en el punto actual.

Ejemplo Paso a Paso: Planteamiento

Problema

Resolver la siguiente ecuación diferencial:

$$\frac{dx}{dt} = t + x$$

Nuestra función pendiente es ***f(t, x) = t + x***

Valores Iniciales

- ***t0*** = 0
- ***x0*** = 1
- ***h*** (paso) = 0.1

Ejecución de Iteraciones

Iteración (k)	(tk)	(xk)	Cálculo: xk + h · (tk + xk)	Nuevo (xk+1)
0	0.0	1.0	1.0 + 0.1 · (0 + 1)	**1.1**
1	0.1	1.1	1.1 + 0.1 · (0.1 + 1.1)	**1.22**
2	0.2	1.22	1.22 + 0.1 · (0.2 + 1.22)	**1.362**

El proceso se repite sucesivamente incrementando ***t*** en ***h*** en cada paso.

Arquitectura del Programa

Para implementar este método numérico, utilizamos las siguientes librerías de Python:

Tkinter

Responsable de la **Interfaz Gráfica (GUI)**. Permite al usuario ingresar funciones y valores iniciales sin editar código.

Sympy + Numpy

Sympy traduce el texto "t + x" a una expresión matemática simbólica. **Numpy** evalúa esa expresión numéricamente a alta velocidad.

Matplotlib

Genera la visualización gráfica de los resultados, integrándose directamente en la ventana de la aplicación.

Lectura y Conversión Simbólica

El núcleo de la flexibilidad del programa. Convertimos el texto del usuario en una función matemática ejecutable.

- Definimos t y x como símbolos.
- Sympify convierte el string de entrada en una ecuación.
- Lambdify compila esa ecuación para usarla con Numpy.

```
 # DEFINIENDO LOS SÍMBOLOS  t = sp.Symbol('t') x =
 sp.Symbol('x') # ENTRADAS DEL USUARIO  f_str =
 entry_funcion.get() f_tx = sp.sympify(f_str)
# TRADUCTOR A FUNCIÓN PYTHON  f = sp.lambdify((t,
 x), f_tx, "numpy")
```

El Algoritmo de Euler

```
while (h > 0 and tk_ < tf) or (h < 0 and tk_ > tf):

    x_next = xk + h * f(tk_, xk)
    t_next = tk_ + h
```

Este ciclo while es el corazón del método numérico.

Calcula x_next usando exactamente la fórmula matemática que vimos:

$$xk + h * f(tk\_, xk)$$

En cada iteración, actualizamos los valores actuales para el siguiente paso.

Visualización de Resultados

Finalmente, tomamos las listas acumuladas ts y xs para graficar la solución aproximada.

Utilizamos FigureCanvasTkAgg para incrustar esta gráfica de Matplotlib dentro de nuestra ventana de Tkinter, creando una experiencia de usuario unificada.

```
ax = fig.add_subplot(111)
ax.plot(ts, xs, marker='o', linestyle='-')
ax.set_title("Solución con Método de Euler")
ax.set_xlabel("t")
ax.set_ylabel("x")
ax.grid(True)
canvas.draw()
```

¿Preguntas?

Gracias por su

atención

Equipo 4