UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA

Universidad Autónoma de Querétaro

FACULTAD DE INGENIERÍA

MÉTODO DE EULER Y RUNGE-KUTTA

Análisis numérico

Autor:
David Gómez Torres

26 Noviembre del 2021

ÍNDICE GENERAL

1. Introducción **1**

1.1. Método de Euler 1

1.1.1. Análisis del error para el método de Euler 2

1.1.2. Algoritmo para el método de Euler 2

1.2. Método de Heun 3

1.3. Método de Runge-Kutta 4

1.3.1. Método de Runge-Kutta 4 orden 4

2. Metodología **5**

2.1. Problema 25.13: Método de Heun 5

2.2. Problema 25.14: Método de Runge-Kutta 6

3. Bibliografía **7**

CAPÍTULO 1

INTRODUCCIÓN

1.1. Método de Euler

La primera derivada ofrece una estimación directa de la pendiente en x_i:

$$\phi = f(x_i, y_i) \tag{1.1}$$

donde f(x_i, y_i) es la ecuación diferencial evaluada en x_i y y_i.

La estimación se sustituye en la ecuación

$$y_{i+1} = y_i + \phi h \tag{1.2}$$

entonces

$$y_{i+1} = y_i + f(x_i, y_i)h \tag{1.3}$$

Esta fórmula se conoce como **método de Euler** (o de Euler-Cauchy o de **punto pendiente**). Se predice un nuevo valor de y usando la pendiente (igual a la primera derivada en el valor original de x) para extrapolar linealmente sobre el tamaño de paso h. Ver figura (1.1).

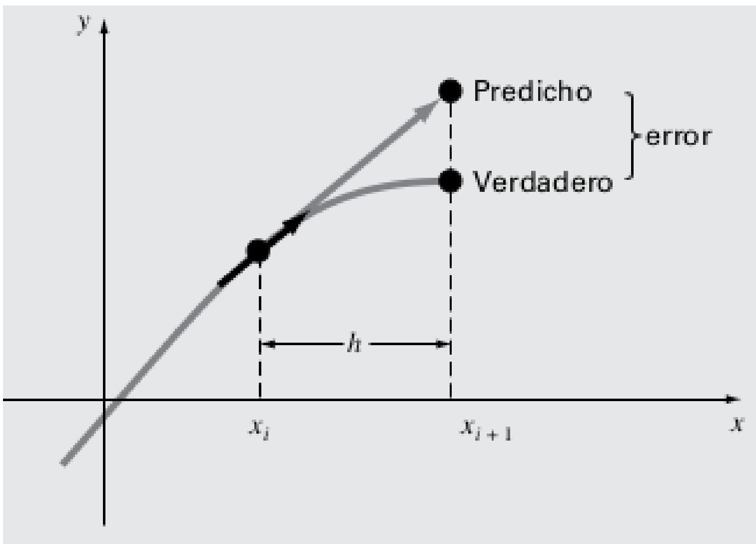

Figura 1.1: Método de Euler

1.1.1. Análisis del error para el método de Euler

La solución numérica de las EDO implica dos tipos de error:

1. Errores de truncamiento, originados por la naturaleza de las técnicas empleadas para aproximar los valores de y.

2. Errores de redondeo, causados por el número limitado de cifras significativas que una computadora puede retener.

El error de truncamiento en el método de Euler se atribuye a los términos remanentes en la expansión de la serie de Taylor, que no se incluyeron en la ecuación (1.3). Al restar la ecuación (1.3), se llega a

$$E_t \frac{f'(x_i, y_i)}{2!} h^2 + \cdots + O(h^{n+1}) \tag{1.4}$$

donde E_t =error de truncamiento local verdadero. Para h suficientemente pequeña, los errores en los términos de la ecuación (1.4) normalmente disminuyen, en tanto aumenta el orden y el resultado se representa como:

$$E_a = \frac{f'(x_i . y_i)}{2!} h^2 \tag{1.5}$$

o

$$E_a = O(h^2) \tag{1.6}$$

donde E_a = error de truncamiento local aproximado.

1.1.2. Algoritmo para el método de Euler

Algorithm 1 Pseudocódigo para el método de Euler

```
y = valor inicial variable dependiente
xi = valor inicial variable independiente
xf = valor final variable independiente
dx = cálculo del tamaño de paso
xout = intervalo de salida
x = xi
m = 0
xp_m = x
yp_m = y
for xend = x + xout do
    if (xend ¿xf) then
        xend = xf
        h = dx
        m = m + 1
        xp_m = x
        yp_m = y
    end if
end for
```

1.2. Método de Heun

Un método para mejorar la estimación de la pendiente emplea la determinación de dos derivadas en el intervalo (una en el punto inicial y otra en el final). Las dos derivadas se promedian después con la finalidad de obtener una mejor estimación de la pendiente en todo el intervalo.
Este procedimiento, conocido como **método de Heun**, ver Figura (1.2).

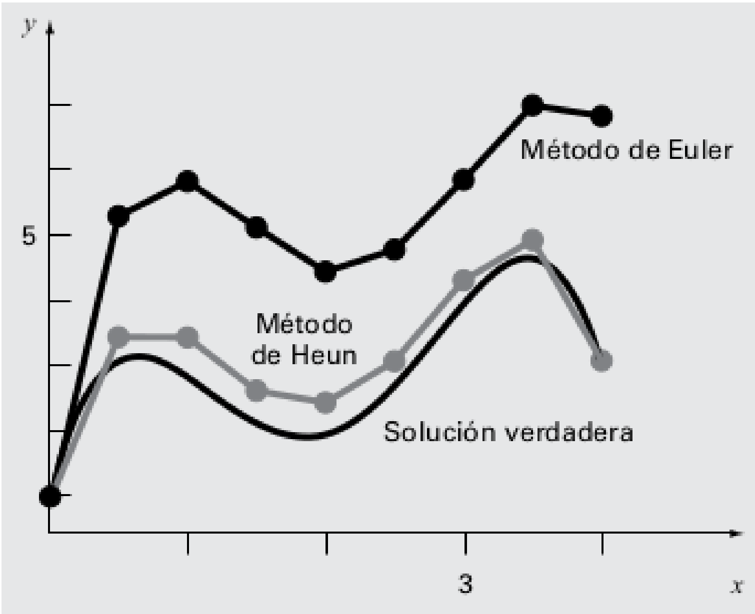

Figura 1.2: Comparación entre métodos

La **ecuación predictora** o simplemente predictor. Da una estimación de y_{i+1} que permite el cálculo de una estimación de la pendiente al final del intervalo:

$$y'_{i+1} = f(x_{i+1}, y^0_{i+1}) \tag{1.7}$$

Así, se combinan las dos pendientes para obtener una pendiente promedio en el intervalo:

$$\bar{y'} = \frac{y'_i + y'_{i+1}}{2} = \frac{f(x_i, y_i) + f(x_{i+1}, y^0_{i+1})}{2} \tag{1.8}$$

Esta pendiente promedio se utiliza después para extrapolar linealmente desde y_i hasta y_{i+1} con el método de Euler:

$$y_{i+1} = y_i + dfracf(x_i, y_i) + f(x_{i+1}, y^0_{i+1})2 \tag{1.9}$$

que se conoce como **ecuación correctora** o simplemente corrector.

Como en los métodos iterativos similares analizados en secciones anteriores de este libro, un criterio de terminación para la convergencia del corrector está dado por:

$$|\epsilon_a| = \left| \frac{y^j_{i+1} - y^{j-1}_{i+1}}{y^j_{i+1}} \right| 100\,\% \tag{1.10}$$

1.3. Método de Runge-Kutta

Los métodos de *Runge-Kutta (RK)* logran la exactitud del procedimiento de la serie de Taylor sin necesitar el cálculo de derivadas de orden superior. Tienen la forma generalizada

$$y_{i+1} = y_i + \phi(x_i, y_i, h)h \tag{1.11}$$

donde $\phi(x_i, y_i, h)$ se conoce como **función incremento**, la cual puede interpretarse como una pendiente representativa en el intervalo. La función incremento se escribe en forma general como:

$$\phi = a_1k_1 + a_2k_2 + \cdots + a_nk_n \tag{1.12}$$

donde las a son constantes y las k son

$$k_1 = f(x_i, y_i) \tag{1.13}$$

$$k_2 = f(x_i + p_1h, y_i + q_{11}k_1h) \tag{1.14}$$

$$k_3 = f(x_i + p_2h, y_i + q_{21}k_1h + q_{22}k_2h) \tag{1.15}$$

$$k_n = f(x_i + p_{n-1}h, y_i + q_{n-1}k_1h + q_{n-1,2}k_2h + \cdots + q_{n-1,n-1}k_{n-1}h) \tag{1.16}$$

donde las p y las q son constantes.

1.3.1. Método de Runge-Kutta 4 orden

El más popular de los métodos RK es el de *cuarto orden*. Como en el caso de los procedimientos de segundo orden, hay un número infinito de versiones. La siguiente, es la forma comúnmente usada y, por lo tanto, le llamamos **método clásico RK de cuarto orden**:

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h \tag{1.17}$$

donde

$$k_1 = f(x_i, y_i) \tag{1.18}$$

$$k_2 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1h) \tag{1.19}$$

$$k_3 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2h) \tag{1.20}$$

$$k_4 = f(x_i + h, y_i + k_3h) \tag{1.21}$$

CAPÍTULO 2

METODOLOGÍA

2.1. Problema 25.13: Método de Heun

Problema 23.13

Haga un programa amigable para el usuario para el método de Heun con corrector iterativo.

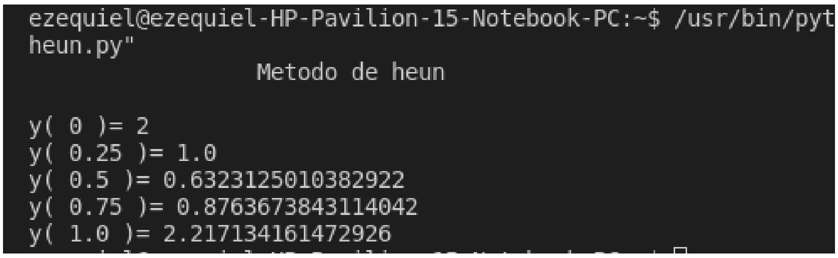

Figura 2.1: Ejecución del código

```python
"""
Programa para calcular método de Heun
"""
from math import *
import numpy as np

cadena0 = "Metodo de Heun\n".capitalize()
print(cadena0.center(50, " "))

def f(t,y):
    func = t*exp(3*t)-2*y
    return func

def Heun(t,y,h,n):
    print('y(',t,')=',y)
    for k in range(n):
        y = y + h*f(t,y)
        t = t + h
        print('y(',t,')=',y)

Heun(0,2,0.25,4)
```

2.2. Problema 25.14: Método de Runge-Kutta

Problema 25.14

Desarrolle un programa de computadora amigable para el usua- rio para el método clásico de RK de cuarto orden.

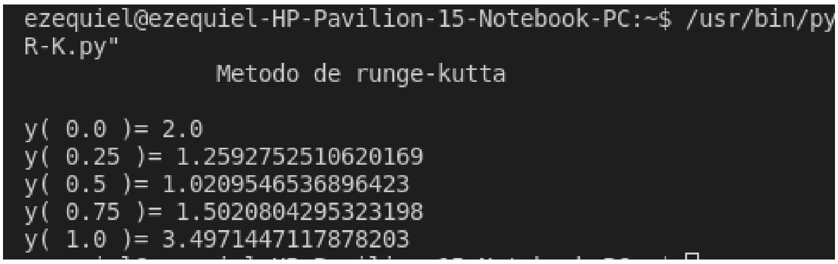

Figura 2.2: Ejecución del código

```python
"""
Programa para calcular método de R-K orden 4
"""

from math import *
import numpy as np

def f(t,y):
    func = t*exp(3*t)-2*y
    return func

cadena0 = "Metodo de Runge-Kutta\n".capitalize()
print(cadena0.center(50, " "))

def RungeKutta04(t0,y0,h,n):
    t = np.zeros(n+1)
    y = np. zeros(n+1)
    t[0] = t0
    y[0] = y0
    print('y(',t[0],')=',y[0])
    for k in range(n):
        k1 = f(t[k],y[k])
        k2 = f(t[k]+h/2,y[k]+(h/2)*k1)
        k3 = f(t[k]+h/2,y[k]+(h/2)*k2)
        k4 = f(t[k]+h,y[k]+h*k3)
        y[k+1] = y[k] +(h/6)*(k1+2*k2+2*k3+k4)
        t[k+1] = t[k]+h
        print('y(',round(t[k+1],3),')=',y[k+1])

RungeKutta04(0,2,0.25,4)
```

CAPÍTULO 3

BIBLIOGRAFÍA

1. Chapra, S. C., Canale, R. P. (2011). *Numerical methods for engineers* (Vol. 1221). New York: Mcgraw-hill.