

Método Preditor-Corretor



Utilizados para resolver equações diferenciais ordinárias (EDOs) em problemas reais que envolvem variação contínua no tempo ou espaço.

- **Engenharia:** modelagem de vibrações mecânicas, circuitos elétricos, transferência de calor, estruturas sujeitas a carregamentos variáveis.
- **Física:** simulações de movimento, dinâmica de partículas, sistemas de controle.
- **Biologia e Medicina:** modelagem de crescimento populacional, propagação de doenças, metabolismo.
- **Economia:** previsão de crescimento, dinâmica de mercados e fluxos financeiros.
- **Computação gráfica e simulações físicas:** sistemas de partículas, animações baseadas em física.

Método Preditor-Corretor



Adams-Bashforth

- **Método Explícito**

Vantagens:

- Simples de implementar;
- Computacionalmente eficiente (não requer resolver equações implícitas)

Desvantagens:

- Menos estável em certos casos;
- Requer valores anteriores (Runge-Kutta 4ª ordem)

Adams-Moulton

- **Método Implícito**

Vantagens:

- Mais estável que Adams-Bashforth;
- Permite maiores passos sem perder tanta precisão.

Desvantagens:

- Mais complexo de implementar;
- Precisa resolver uma equação (geralmente por iteração, como Newton-Raphson)

Método Preditor-Corretor



Adams-Bashforth

$$y_{i+1}^{(o)} = y_i + \frac{h}{24} [55 f_i - 59 f_{i-1} + 37 f_{i-2} - 9 f_{i-3}]$$

Adams-Moulton

$$y_{i+1}^{(k)} = y_i + \frac{h}{24} [9 f_{i+1}^{(k-1)} + 19 f_i - 5 f_{i-1} + f_{i-2}]$$

$$\left| \frac{y_{i+1}^{(k)} - y_{i+1}^{(k-1)}}{y_{i+1}^{(k)}} \right| \leq \xi \quad k = 1, 2, 3, \dots$$

Exemplo

Dado o seguinte PVI:

$$y' = \frac{1}{y^2} + x \ln y$$

$$y(1) = 0,8$$

Aproxime $y(2)$ usando $h=0,2$.

Sabe-se que pelo MRK 4ª ordem:

$$x_0 = 1; y_0 = 0,8 \quad f(x_0, y_0) = 1,3394$$

$$x_1 = 1,2; y_1 = 1,0227 \quad f(x_1, y_1) = 0,9830$$

$$x_2 = 1,4; y_2 = 1,2131 \quad f(x_2, y_2) = 0,9500$$

$$x_3 = 1,6; y_3 = 1,4116 \quad f(x_3, y_3) = 1,0534$$

Adote $\varepsilon = 0,5 * 10^{-2}$ e erro relativo como critério de parada.



Exemplo

Dado o seguinte PVI:

$$y' = \frac{1}{y^2} + x \ln y$$

$$y(1) = 0,8$$

Aproxime $y(2)$ usando $h=0,2$.

Sabe-se que pelo MRK 4ª ordem:

$$x_0 = 1; y_0 = 0,8 \quad f(x_0, y_0) = 1,3394$$

$$x_1 = 1,2; y_1 = 1,0227 \quad f(x_1, y_1) = 0,9830$$

$$x_2 = 1,4; y_2 = 1,2131 \quad f(x_2, y_2) = 0,9500$$

$$x_3 = 1,6; y_3 = 1,4116 \quad f(x_3, y_3) = 1,0534$$

Adote $\varepsilon = 0,5 * 10^{-2}$ e erro relativo como critério de parada.

R: 1,9241 (Resolução em anexo)



Função em Python



```
import sympy
import math
from sympy.abc import x, y

def predictor_corrector(func, y0, x0, h, n):
    """
    Resolve uma EDO  $dy/dx = func(x, y)$  usando o método preditor-corretor.

    Args:
        func: A função que define a EDO, como uma expressão simbólica do sympy.
        y0: O valor inicial de y em x0.
        x0: O valor inicial de x.
        h: O tamanho do passo.
        n: O número de passos a serem executados.

    Returns:
        Uma lista de tuplas (x_i, y_i) representando a solução aproximada.
    """
    points = [(x0, y0)]
    y_prev = y0
    x_prev = x0

    for _ in range(n):
        # Preditor (usando o método de Euler)
        y_predictor = y_prev + h * func.subs({x: x_prev, y: y_prev})

        # Corretor (usando o método trapezoidal)
        x_curr = x_prev + h
        y_corrector = y_prev + h/2 * (func.subs({x: x_prev, y: y_prev}) + func.subs({x: x_curr, y: y_predictor}))

        points.append((x_curr, y_corrector))
        y_prev = y_corrector
        x_prev = x_curr

    return points

edo_func = sympy.sympify(input('dy/dx = '))

# Defina os parâmetros iniciais
y_initial = float(input("Digite o valor inicial de y (y0): "))
x_initial = float(input("Digite o valor inicial de x (x0): "))
step_size = float(input("Digite o tamanho do passo (h): "))
num_steps = int(input("Digite o número de passos (n): "))

# Resolva a EDO usando o método preditor-corretor
solution = predictor_corrector(edo_func, y_initial, x_initial, step_size, num_steps)

# Imprima a solução
for point in solution:
    print(f"x = {point[0]:.2f}, y = {point[1]:.4f}")
```

