Universidade de São Paulo
Instituto de Física de São Carlos

Lista 3

Pedro Calligaris Delbem 5255417

Professor: Attilio Cucchieri

Abril de 2025

Sumário

1 Runge-Kutta Methods

1.1 Exercício 1

Tarefa: Demonstrar que o erro da interpolação linear

$$f(x,y) = \frac{x - x_{n-1}}{h} f_n + \frac{x_n - x}{h} f_{n-1}$$

é $\mathcal{O}(h^2)$

Demonstração:

Seja $f(x) \in C^2$ queremos interpolar $f(x)$ entre os pontos x_{n-1} e $x_n = x_{n-1} + h$ utilizando interpolação linear, e estimar o erro:

$$E(x) = f(x) - \tilde{f}(x)$$

onde

$$\tilde{f}(x) = \frac{x - x_{n-1}}{h} f(x_n) + \frac{x_n - x}{h} f(x_{n-1})$$

Expandimos $f(x_n)$ em série de Taylor ao redor de x_{n-1}:

$$f(x_n) = f(x_{n-1} + h) = f(x_{n-1}) + hf'(x_{n-1}) + \frac{h^2}{2} f''(\xi_1), \quad \xi_1 \in (x_{n-1}, x_n)$$

Expandimos também $f(x)$ ao redor de x_{n-1}:

$$f(x) = f(x_{n-1}) + (x - x_{n-1}) f'(x_{n-1}) + \frac{(x - x_{n-1})^2}{2} f''(\xi_2), \quad \xi_2 \in (x_{n-1}, x)$$

Substituímos a expansão de $f(x_n)$ na expressão de $\tilde{f}(x)$:

$$\tilde{f}(x) = \frac{x - x_{n-1}}{h} \left[f(x_{n-1}) + hf'(x_{n-1}) + \frac{h^2}{2} f''(\xi_1) \right] + \frac{x_n - x}{h} f(x_{n-1})$$

Logo

$$\tilde{f}(x) = \frac{x - x_{n-1}}{h} f(x_{n-1}) + (x - x_{n-1}) f'(x_{n-1}) + \frac{(x - x_{n-1})h}{2} f''(\xi_1) + \frac{x_n - x}{h} f(x_{n-1})$$

Como $x_n = x_{n-1} + h$, temos $x_n - x = h - (x - x_{n-1})$ - assim:

$$\tilde{f}(x) = \left(\frac{x - x_{n-1}}{h} + \frac{h - (x - x_{n-1})}{h} \right) f(x_{n-1}) + (x - x_{n-1}) f'(x_{n-1}) + \frac{(x - x_{n-1})h}{2} f''(\xi_1)$$

Segue que:

$$\tilde{f}(x) = f(x_{n-1}) + (x - x_{n-1}) f'(x_{n-1}) + \frac{(x - x_{n-1})h}{2} f''(\xi_1)$$

Fazendo a subtração entre $f(x)$ e $\tilde{f}(x)$, obtemos:

$$\begin{aligned} f(x) - \tilde{f}(x) = &\left[f(x_{n-1}) + (x - x_{n-1}) f'(x_{n-1}) + \frac{(x - x_{n-1})^2}{2} f''(\xi_2) \right] \\ &- \left[f(x_{n-1}) + (x - x_{n-1}) f'(x_{n-1}) + \frac{(x - x_{n-1})h}{2} f''(\xi_1) \right] \end{aligned}$$

Deste modo:

$$\begin{aligned} f(x) - \tilde{f}(x) &= \frac{1}{2} \left[(x - x_{n-1})^2 f''(\xi_2) - (x - x_{n-1}) h f''(\xi_1) \right] \\ &= \frac{(x - x_{n-1})}{2} \left[(x - x_{n-1}) f''(\xi_2) - h f''(\xi_1) \right] \end{aligned}$$

Podemos reescrever esse erro como:

$$f(x) - \tilde{f}(x) = -\frac{(x - x_{n-1})(x - x_n)}{2} f''(\tilde{\xi}) \quad \text{para algum } \tilde{\xi} \in (x_{n-1}, x_n)$$

onde usamos que $(x - x_{n-1})(x - x_n) = (x - x_{n-1})^2 - h(x - x_{n-1})$, e agrupamos os termos com uma média das derivadas.

Portanto, o erro da interpolação linear de $f(x) \in C^2$ entre dois pontos é dado por:

$$\boxed{f(x) - \tilde{f}(x) = -\frac{(x-x_{n-1})(x-x_n)}{2} f''(\xi) \quad \text{para algum } \xi \in (x_{n-1}, x_n)}$$

Como $|(x-x_{n-1})(x-x_n)| \leq \frac{h^2}{4}$, então:

$$|f(x) - \tilde{f}(x)| = \mathcal{O}(h^2)$$

1.2 Exercício 2

Tarefa: Resolva a equação

$$\frac{d^2y(x)}{dx^2} = -4\pi^2 y(x)$$

com as condições iniciais $y(x=0) = 0$ e $y'(x=0) = 0$ usando o algoritimo de Runge-Kutta.

Código Escrito:

```fortran
!-----------------------------------------------------------------
! File: L3-5255417-ex-2.f90
!
! Description:
!   Solve second order differential equation using Runge-Kutta
  method
!
! Dependencies:
!   - None
!
! Since:
!   - 03/2025
!
! Authors:
!   - Pedro C. Delbem <pedrodelbem@usp.br>
!-----------------------------------------------------------------
program Runge_Kutta

    !deactivate implicit typing
    implicit none

    !declare parameters
    integer, parameter :: n = 100
    real, parameter :: lambda = -4.0*(2.0*atan(1.0))**2.0

    !declare variables
    integer :: i
    real y(0:n), v(0:n), k1, k2, k3, k4, l1, l2, l3, l4, h

    !define h
    write(*,*) "Insert h:"
```

```fortran
    read(*,*) h

    !initialize y and v
    y(0) = 1.0
    v(0) = 0.0

    !open file for writing results
    open(1, file='results.txt', status='replace')

    !compute Runge-Kutta method
    do i = 0, n-1

        !print current step
        write(1,*) i, y(i), v(i)

        !compute coefficients
        k1 = v(i)
        l1 = h*y(i)
        k2 = v(i) + h*l1/2.0
        l2 = lambda*(y(i) + h*k1/2.0)
        k3 = v(i) + h*l2/2.0
        l3 = lambda*(y(i) + h*k2/2.0)
        k4 = v(i) + h*l3
        l4 = lambda*(y(i) + h*k3)

        !update y and v
        y(i+1) = y(i) + h*(k1 + 2.0*k2 + 2.0*k3 + k4)/6.0
        v(i+1) = v(i) + h*(l1 + 2.0*l2 + 2.0*l3 + l4)/6.0

    end do

    !print last step
    write(1,*) n, y(n), v(n)

    !close file
    close(1)

end program Runge_Kutta
```

O código foi compilado com o comando:

```
gfortran L3-5255417-ex-2.f90 -o L3-5255417-ex-2.exe
```

Resultados:

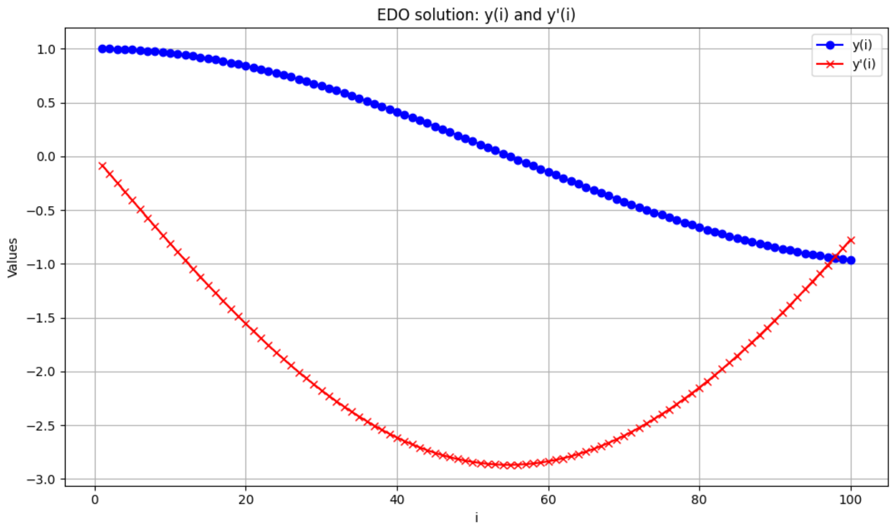

Figura 1: Gráficos de $y(x)$ e $y'(x)$ com h = 0.01

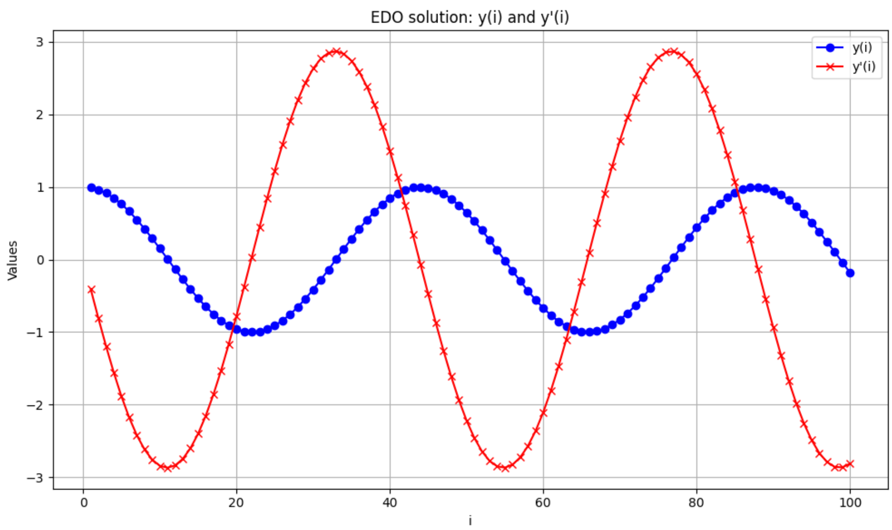

Figura 2: Gráficos de $y(x)$ e $y'(x)$ com h = 0.05

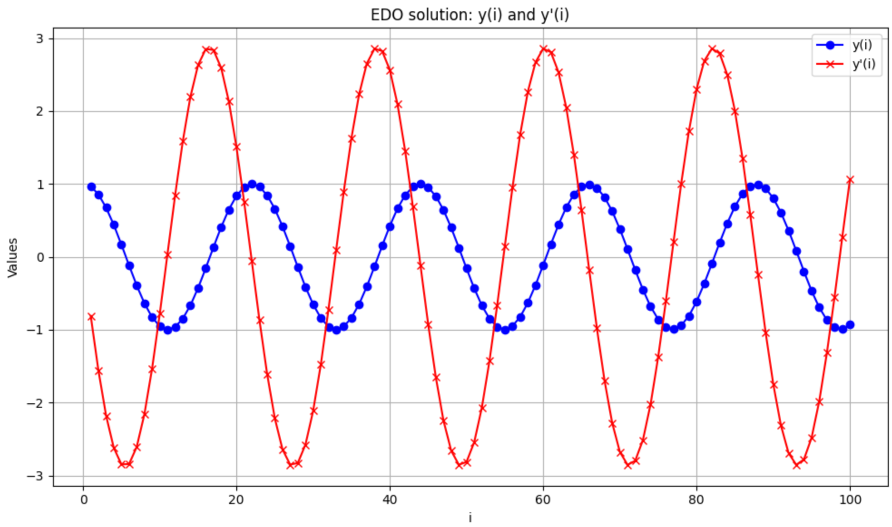

Figura 3: Gráficos de $y(x)$ e $y'(x)$ com h = 0.1

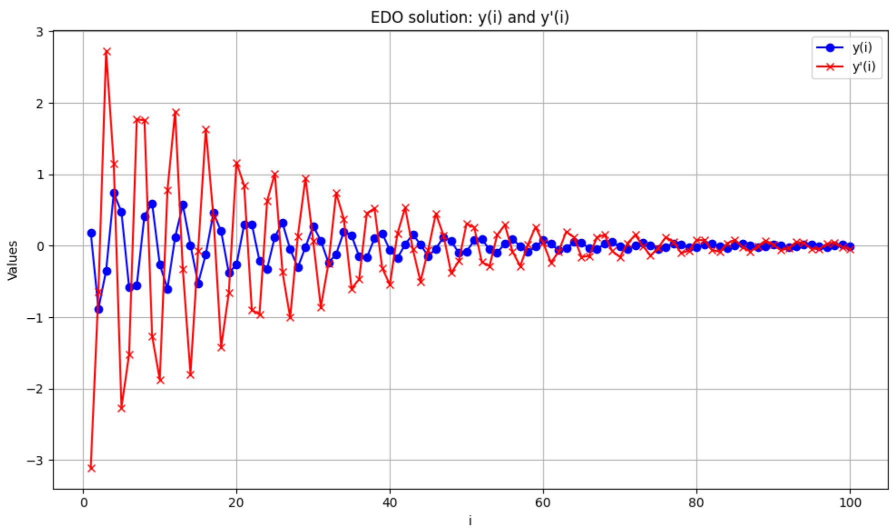

Figura 4: Gráficos de $y(x)$ e $y'(x)$ com h = 0.5

Nota-se que para os valores de h = 0.1, h = 0.01 e h = 0.05, o resultado obtido são funções cossenoidais que oscilam de -1 a 1 o que corresponde ao comportamento esperado para a solução da equação diferencial dada. Além disso, as derivadas são funções senoidais que oscilam de -2π a 2π o que também corresponde ao resultado esperado. Para o valor de h = 0.5, o resultado obtido não é satisfatório, pois a função não é cossenoidal e a derivada não é senoidal, o que indica que o valor de h = 0.5 é muito grande para o algoritmo de Runge-Kutta.

2 The Numerov Algorithm

2.1 Exercício 3

Tarefa: Resolva a equação

$$\frac{d^2y(x)}{dx^2} = -4\pi^2 y(x)$$

com as condições iniciais $y(0) = 1$ e $y'(0) = 0$ usando o algoritimo de Numerov. Explique como foram escolhidos os valores iniciais y_0 e y_1. Compare o resultado com a solução obtida no exercício anterior.

Código Escrito:

```
!------------------------------------------------------------
! File: L3-5255417-ex-3.f90
!
! Description:
!   Solve second order differential equation using Runge-Kutta method
!
! Dependencies:
!   - None
!
! Since:
!   - 03/2025
!
! Authors:
!   - Pedro C. Delbem <pedrodelbem@usp.br>
!------------------------------------------------------------
program Numerov

    !deactivate implicit typing
    implicit none

    !declare parameters
    integer, parameter :: n = 100
    real, parameter :: lambda = 4.0*(2.0*atan(1.0))**2.0 !y''(x) + lambda*y(x) = 0

    !declare variables
    integer :: i
    real y(0:n), h

    !define h
    write(*,*) "Insert h:"
    read(*,*) h

    !initialize y and v
    y(0) = 1.0
    y(1) = 1.0 + lambda*(h**2.0)/2.0

    !open file for writing results
    open(1, file='results.txt', status='replace')
```

```fortran

    !compute Numerov method
    do i = 1, n-1

        !print current step
        write(1,*) i, y(i)

        !update y
        y(i+1) = ( 2.0*y(i)*(1.0 - 5.0*h**2*lambda/12.0) - y(i-1)*(1.0 + h**2*lambda/12.0) ) / (1.0 + h**2*lambda/12.0)

    end do

    !print last step
    write(1,*) n, y(n)

    !close file
    close(1)

end program Numerov
```

O código foi compilado com o comando:

```
gfortran L3-5255417-ex-3.f90 -o L3-5255417-ex-3.exe
```

Resultados:

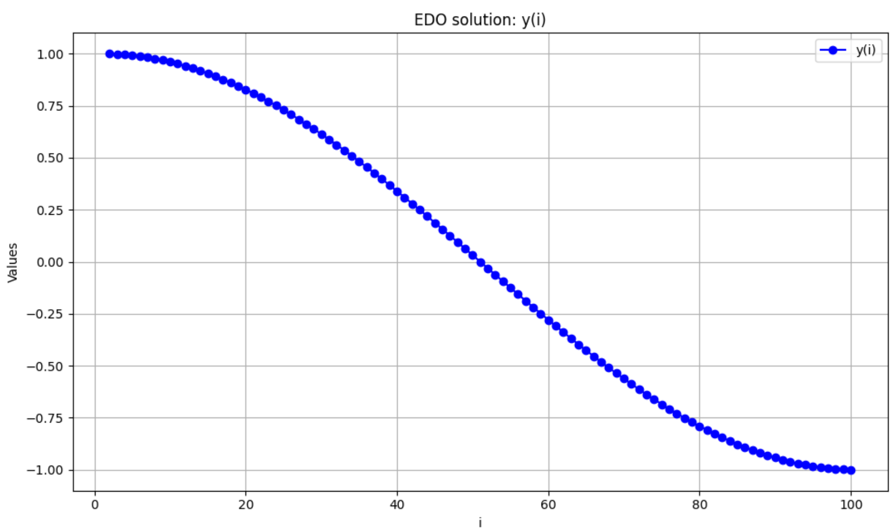

Figura 5: Gráficos de $y(x)$ e $y'(x)$ com h = 0.01

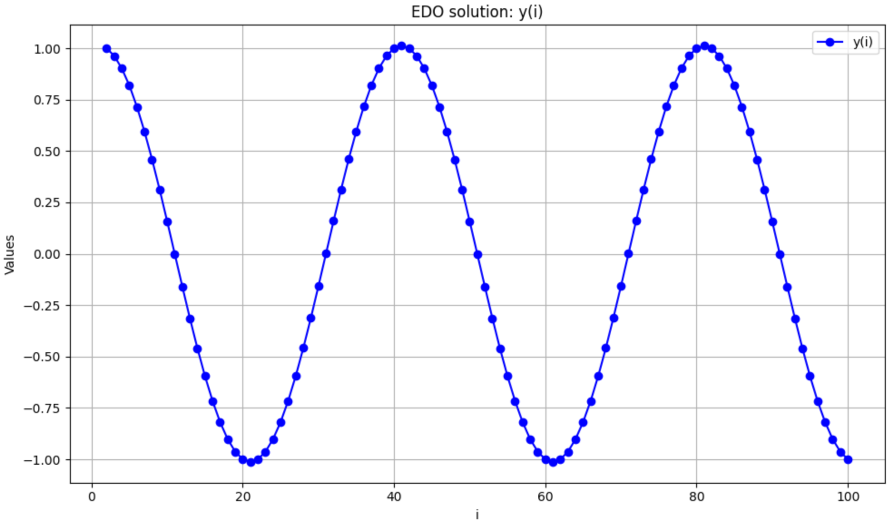

Figura 6: Gráficos de $y(x)$ e $y'(x)$ com h = 0.05

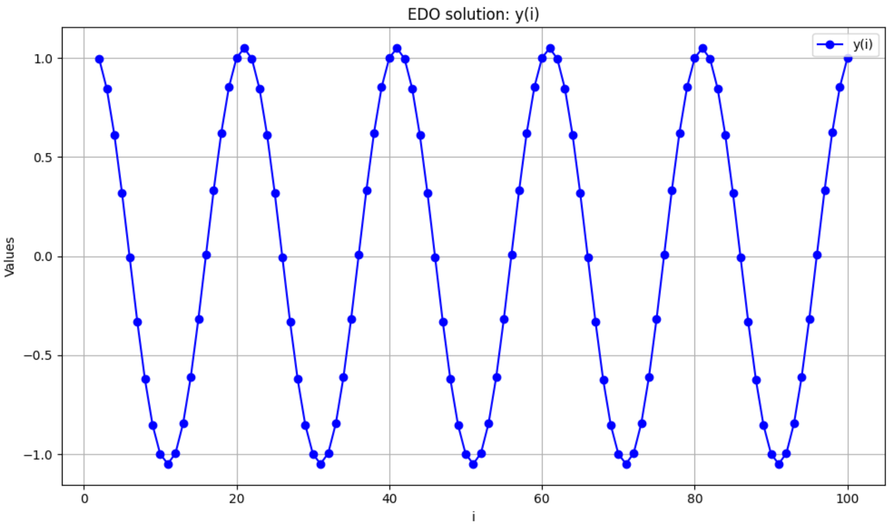

Figura 7: Gráficos de $y(x)$ e $y'(x)$ com h = 0.1

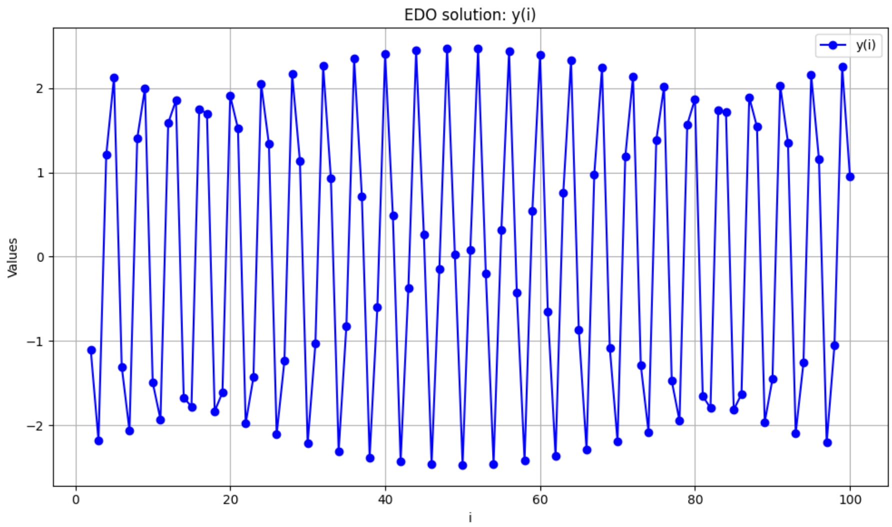

Figura 8: Gráficos de $y(x)$ e $y'(x)$ com h = 0.5

Nota-se que para os valores de h = 0.1, h = 0.01 e h = 0.05, o resultado obtido são funções cossenoidais que oscilam de -1 a 1 o que corresponde ao comportamento esperado para a solução da equação diferencial. Para o valor de h = 0.5, o resultado obtido não é satisfatório, pois a função não é cossenoidal, o que indica que o valor de h = 0.5 é muito grande para o algoritmo de Numerov.