Método das Diferenças Finitas para problema de Sturm-Liouville

Prof. Doherty Andrade

www.metodosnumericos.com.br

1 Diferenças finitas para o problema de Sturm-Liouville

Vamos estudar o método das diferenças finitas para problema de Sturm-Liouville, isto é, problemas do tipo:

$$\begin{aligned} &y'' = p(x)y' + q(x)y + r(x),\ a < x < b,\\ &y(a) = \alpha,\\ &y(b) = \beta. \end{aligned}$$

Tomemos $h = \frac{b-a}{n}$, então $x_i = a + ih$. Usando diferenças centrais temos que:

$$\begin{aligned} y''(x_i) &= \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2},\\ p_i y'(x_i) &= p_i \frac{y_{i+1} - y_{i-1}}{2h}, \end{aligned}$$

donde segue que

$$-\left(1 + \frac{p_i h}{2}\right) y_{i-1} + \left(2 + q_i h^2\right) y_i - \left(1 - \frac{p_i h}{2}\right) y_{i+1} = -h^2 r_i, \tag{1}$$

$i = 1, \ldots, n-1$.

Vamos denotar por

$$\begin{aligned} \alpha_i &= -(1 + \frac{p_i h}{2}),\\ \beta_i &= 2 + qh^2,\\ \gamma_i &= -1 + \frac{p_i h}{2}. \end{aligned}$$

Notemos que o sistema é tridiagonal de dimensão $(n-1) \times (n-1)$ e pode ser escrito da forma $AX = b$, onde

$$A = (a_{ij}) = \begin{cases} 0, \text{ se } |i-j| > 1,\\ \beta_i, \text{ se } i = j,\\ \alpha_i, \text{ se } j = i+1,\\ \gamma_i, \text{ se } j = i-1 \end{cases}$$

e

$$b = (b_i) = \begin{cases} -r_1 h^2 + \alpha \left(1 + \frac{p_1 h}{2}\right), \text{ se } i = 1, \\ -r_i h^2, \text{ se } 2 \leq i \leq n-1, \\ -r_n h^2 + \beta \left(1 - \frac{p_n h}{2}\right), \text{ se } i = n. \end{cases}$$

Exemplo: Vamos considerar o PVF dado por

$$\begin{cases} y'' = -\frac{2}{x}y' + \frac{2}{x^2}y + \frac{\sinh(\ln(x))}{x^2}, 1 \leq x \leq 2 \\ y(1) = 1 \\ y(2) = 2. \end{cases}$$

Assim, temos que

$$p(x) = -\frac{2}{x}$$

$$q(x) = \frac{2}{x^2}$$

e

$$r(x) = \frac{\sinh(\ln(x))}{x^2}.$$

2 Script Python

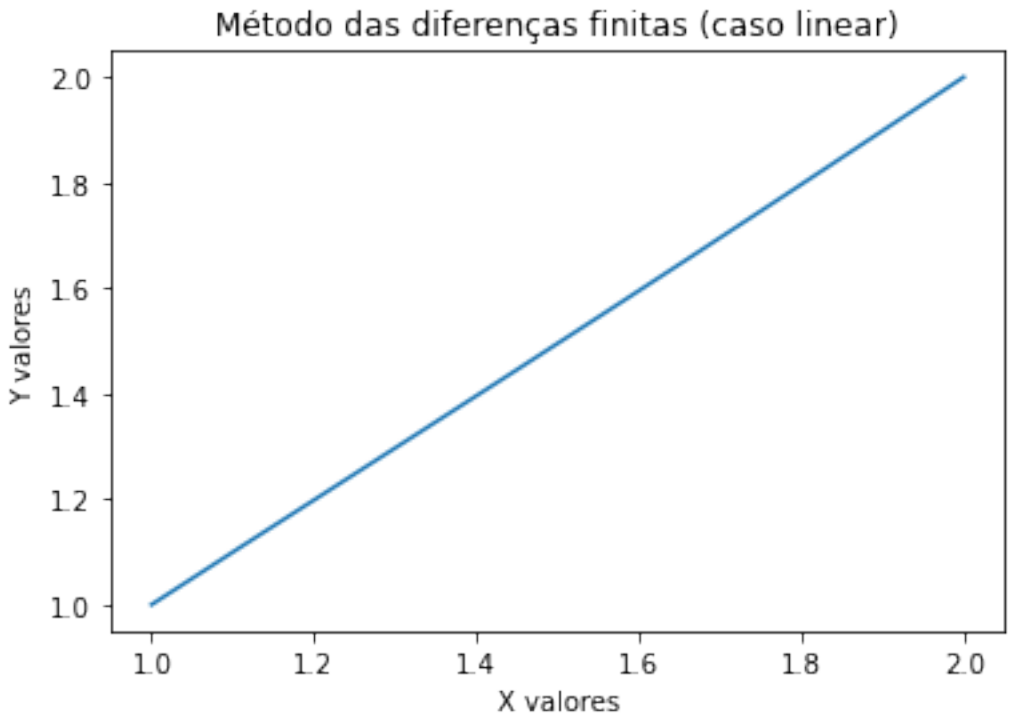
O script abaixo resolve numericamente este PVF por meio do método das diferenças finitas. Você pode modificar as fun'~oes p, q e r e as condições de fronteira de acordo com o seu problema.

```
In [3]: # Método das diferenças finitas caso linear
        # Burden-Faires

        import numpy as np
        import matplotlib.pyplot as plt

        def dif(aa,bb,alpha,beta,n):

            a = np.zeros([n+2])  # 2 cuz we need w_0 and w_n+1
            b = np.zeros([n+2])
            c = np.zeros([n+2])
            d = np.zeros([n+2])

            # n number of x points
            h = (bb-aa)/(n+2)
            print('O valor do passo é h=', h)
            x = aa + h
            a[1] = 2.0 + (h**2)*q(x)
            b[1] = -1.0 + (h/2)*p(x)
            d[1] = -(h**2)*r(x) + (1 + (h/2)*p(x))*alpha

            for i in range(2,n):
```

```
        x = aa + i*h
        a[i] = 2.0 + (h**2)*q(x)
        b[i] = -1.0 + (h/2)*p(x)
        c[i] = -1.0 - (h/2)*p(x)
        d[i] = -(h**2)*r(x)

    x = bb-h
    a[n] = 2.0 + (h**2)*q(x)
    c[n] = -1.0 - (h/2)*p(x)
    d[n] = -(h**2)*r(x) + (1.0 - (h/2)*p(x))*beta

    l = np.zeros([n+2])
    u = np.zeros([n+2])
    z = np.zeros([n+2])

    # Crout algorithm
    l[1] = a[1]
    u[1] = b[1]/a[1]
    z[1] = d[1]/l[1]

    for i in range(2,n):
        l[i] = a[i]-c[i]*u[i-1]
        u[i] = b[i]/l[i]
        z[i] = (d[i] - c[i]*z[i-1])/l[i]


    l[n] = a[n] - c[n]*u[n-1]
    z[n] = (d[n] -c[n]*z[n-1])/l[n]

    w = np.zeros([n+2])

    w[0] = alpha
    w[n+1] = beta
    w[n] = z[n]

    for i in range(n-1,0,-1):
        w[i] = z[i] - u[i]*w[i+1]

    print('As aproximações da solução:',w)
    return w


def p(x):
    return -2/x

def q(x):
    return 2/(x**2)
```

```
def r(x):
    return np.sin(np.log(x))/(x**2)

def main():
    a = 1.0
    b = 2.0
    alpha = 1.0
    beta = 2.0
    n = 8

    w = dif(a,b,alpha,beta,n)

    x = np.linspace(a,b,n+2) # add x_0 and x_n+1
    plt.plot(x,w)
    plt.xlabel("X valores")
    plt.ylabel("Y valores")
    plt.title("Método das diferenças finitas (caso linear)")
    plt.show()

main()
```

```
O valor do passo é h= 0.1
As aproximações da solução: [1.         1.11055933 1.22023971 1.32985255 1.43984526 1.55045727
 1.66180678 1.77394055 1.88686324 2.        ]
```



Exemplo: Consideremos a EDO

$$\begin{cases} y'' = y' - xy + \exp(x)(x^2+1), \ x \in (0,1) \\ y(0) = 0 \\ y(1) = \mathrm{e}. \end{cases}$$

Assim, temos que

$$p(x) = 1$$
$$q(x) = -x$$

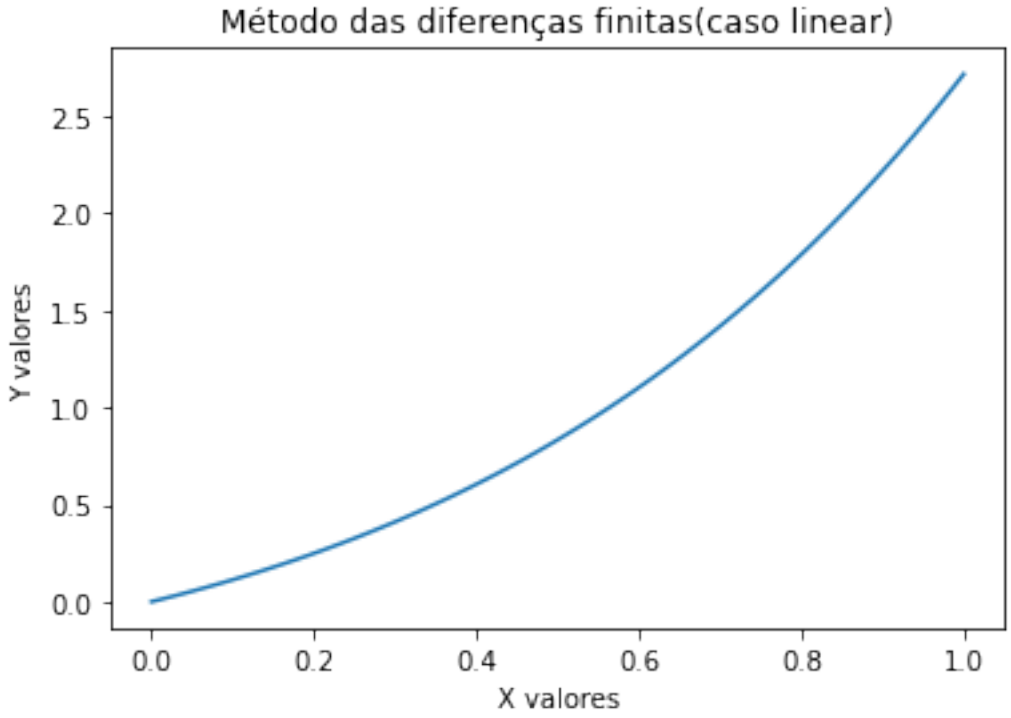
e

$$r(x) = \exp(x)(x^2+1).$$

```
In [2]: def p(x):
            return 1

        def q(x):
            return -x

        def r(x):
            return np.exp(x)*(x**2+1)

        def main():
            a = 0.0
            b = 1.0
            alpha = 0.0
            beta = np.exp(1.0)
            n = 98

            w = dif(a,b,alpha,beta,n)

            x = np.linspace(a,b,n+2) # add x_0 and x_n+1
            plt.plot(x,w)
            plt.xlabel("X valores")
            plt.ylabel("Y valores")
            plt.title("Método das diferenças finitas(caso linear)")
            plt.show()

        main()
```

```
O valor do passo é h= 0.01
As aproximações da solução: [0.         0.01044802 0.02110256 0.03196671 0.04304361 0.05433644
 0.0658484  0.07758277 0.08954284 0.10173197 0.11415354 0.12681098
 0.13970778 0.15284745 0.16623358 0.17986977 0.19375969 0.20790706
 0.22231562 0.2369892  0.25193164 0.26714686 0.28263881 0.29841151
 0.31446901 0.33081543 0.34745494 0.36439175 0.38163015 0.39917447
 0.41702908 0.43519844 0.45368704 0.47249945 0.49164027 0.51111418
 0.53092593 0.5510803  0.57158216 0.59243641 0.61364806 0.63522214
```

```
0.65716377 0.67947812 0.70217044 0.72524603 0.74871027 0.77256861
0.79682657 0.82148972 0.84656374 0.87205433 0.89796732 0.92430856
0.95108402 0.97829971 1.00596174 1.03407629 1.06264962 1.09168805
1.12119801 1.151186   1.1816586  1.21262246 1.24408434 1.27605108
1.30852958 1.34152685 1.37504999 1.40910619 1.44370272 1.47884693
1.5145463  1.55080838 1.58764081 1.62505133 1.66304779 1.70163813
1.74083039 1.7806327  1.82105331 1.86210056 1.90378291 1.9461089
1.98908721 2.03272661 2.07703597 2.12202429 2.16770067 2.21407434
2.26115462 2.30895097 2.35747296 2.40673027 2.45673271 2.50749022
2.55901285 2.61131078 2.66439432 2.71828183]
```



In [ ]: