DAWID MULARCZYK

Elastic deformation differential problem

$$-\frac{d}{dx}\left(E(x)\frac{du(x)}{dx}\right) = 0$$

$$u(2) = 0$$

$$\frac{du(0)}{dx} + u(0) = 10$$

$$E(x) = \begin{cases} 3 & \text{dla } x \in [0,1] \\ 5 & \text{dla } x \in (1,2] \end{cases}$$

Gdzie u to poszukiwana funkcja

$$[0,2] \ni x \rightarrow u(x) \in \mathbb{R}$$

1 Derivation of the variational formulation

1. Wyprowadzenie sformułowania wariacyjnego

$$-E'(x)\cdot u'' = 0 \quad / \cdot v \quad ; \quad v \in V$$

$$-E'(x)\cdot u'' = 0 \cdot v \quad / \int_0^2 0 \cdot v \, dx$$

$$\int_0^2 -E'(x)\cdot u'' \cdot v \, dx = 0$$

$$-E(x)\cdot u' \cdot v \Big|_0^2 + \int_0^2 E(x)\cdot u' \cdot v' \, dx = 0$$

$$\underbrace{E(0)}_{3} \cdot \underbrace{u'(0)}_{10-u(0)} \cdot v(0) - \underbrace{E(2)}_{3} \cdot u'(2) \cdot \underbrace{v(2)}_{0} + \int_0^2 E(x)\cdot u' \cdot v' \, dx = 0$$

$$30\, v(0) - 3\, u(0)\cdot v(0) + \int_0^2 E(x)\, u' v' \, dx = 0$$

$$\underbrace{\int_0^2 E(x)\cdot u' \cdot v' \, dx - 3\, u(0)\cdot v(0)}_{B(u,v)} = \underbrace{-30\, v(0)}_{L(v)}$$

2. Dyskretyzacja problemu

1. warunek dirichleta tylko z prawej

wiem, że

$$e_i(x) = \begin{cases} \frac{x}{h} - i + 1 & , x \in \langle h(i-1), hi \rangle \\ -\frac{x}{h} + i + 1 & , x \in (hi, h(i+1) \rangle \end{cases} \quad ; h = \frac{2}{n}$$

3. postać macierzowa: $\begin{cases} u = u_0 e_0 + u_1 e_1 + \dots + u_{n-1} e_{n-1} \in V \\ v \in V \end{cases}$

$$\begin{bmatrix} B(e_0, e_0) & B(e_1, e_0) & \cdots & B(e_{n-1}, e_0) \\ B(e_1, e_0) & B(e_1, e_1) & & B(e_{n-1}, e_1) \\ \vdots & \vdots & \ddots & \vdots \\ B(e_{n-1}, e_0) & B(e_{n-1}, e_1) & \cdots & B(e_{n-1}, e_{n-1}) \end{bmatrix} \cdot \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{bmatrix} = \begin{bmatrix} L(e_0) \\ L(e_1) \\ \vdots \\ L(e_{n-1}) \end{bmatrix}$$

2 Code solution

 # E(x) function

```
def E(x):
    return 3 if x <= 1 else 5
```

 # FEM Base function and derivative

```
def e(n, i, x):
    h = 2 / n
    return max(0, 1 - abs((x / h - i)))


def e_prim(n, i, x):
    h = 2 / n
    if x <= (i - 1) * h or x >= (i + 1) * h:
        return 0
    else:
        return 1 / h if x <= i * h else -1 / h
```

 # Calculation of numerical integral

```
def calculate_integral(n, i, j):
    start = 2 * max(max(i, j) - 1, 0) / n
    end = 2 * min(min(i, j) + 1, n) / n
    return integration(lambda x: E(x) * e_prim(n, i, x) * e_prim(n, j, x), start,
end)[0] if abs(j - i) <= 1 else 0
```

 # Filling matrix B and vector L

```
def fill(n):
    B, L = np.zeros((n, n)), np.zeros(n)
    L[0] = -30 * e(n, 0, 0)
    for i in range(n):
        for j in range(n):
            integral = calculate_integral(n, i, j)
            B[i, j] = -3 * e(n, i, 0) * e(n, j, 0) + integral
    return B, L
```

 # Plot visualization

```
def show_plot(solution, n):
     sns.lineplot(x=np.linspace(0, 2, n + 1), y=solution)
     plt.title('Elastic deformation plot')
     plt.xlabel('x')
     plt.ylabel('deformation')
     plt.grid(True)
     plt.show()
```

 # Main function

```
if __name__ == '__main__':
   user_input = int(input("Input n: "))
   B, L = fill(user_input)
   show_plot(np.concatenate((np.linalg.solve(B, L), [0])), user_input
```