DAWID MULARCZYK

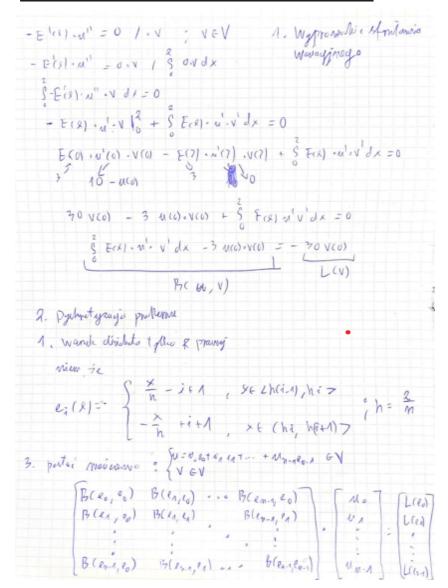
Elastic deformation differential problem

$$\begin{split} -\frac{d}{dx}\left(E(x)\frac{du(x)}{dx}\right) &= 0\\ u(2) &= 0\\ \frac{du(0)}{dx} + u(0) &= 10\\ E(x) &= \left\{ \begin{array}{ll} 3 & \text{dla } x \in [0,1]\\ 5 & \text{dla } x \in (1,2] \end{array} \right. \end{split}$$

Gdzie u to poszukiwana funkcja

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

1 <u>Derivation of the variational formulation</u>



```
#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) \leftarrow 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