

```
1: import numpy as np
2: import matplotlib.pyplot as plt
3:
4: # PDE:
5: #  $-u''(x) + a*u(x) = f(x)$   $x$  in  $(0,1)$ 
6: #  $u(0) = 0$ 
7: #  $u'(1) = \alpha*(g_b - u(1))$ 
8:
9: # parameters
10: a = 1
11: f = 3
12: alpha = 1
13: gb = 1
14: n = 10 # elements
15:
16: # mesh
17: m = n+1 # nodes
18: h = 1.0/n
19: x = np.linspace(0,1,m)
20:
21: # local stiffness matrix
22: K_loc = np.zeros((2,2))
23: A = (1.0/h) * np.array([[ 1,-1], [-1, 1]])
24: B = (a*h/6) * np.array([[ 2, 1], [ 1, 2]]) ✓
25: K_loc = A+B
26:
27: # Assembling
28: K = np.zeros((m,m))
29: F = np.zeros(m)
30: for i in range(n):
31:     K[i:i+2,i:i+2] += K_loc
32:     F[i:i+2] += np.full(2, f*h/2)
33:
34: # Boundary conditions
35: # Dirichlet:  $u(0) = 0$ 
36: K[0,:] = 0
37: K[0,0] = 1
38: F[0] = 0
39: # Neumann:  $u'(1) = \alpha*(g_b - u(1))$ 
40: A[-1,-1] += alpha
41: F[-1] += alpha*gb ✓
42:
43: u = np.linalg.solve(K, F)
44: plt.plot(x, u, "-o", label="u_h")
45: plt.title(f"n = {n} | h = {h} | a = {a} | f(x) = {f} | alpha = {alpha} | gb = {gb}")
46: plt.xlabel("x")
47: plt.ylabel("u(x)")
48: plt.legend()
49: plt.grid(True)
50:
51: print("K = ", K)
52: print("f = ", F)
53: print("u = ", u)
54:
55: plt.tight_layout()
56: plt.savefig("../task_a.png", dpi=300)
57: plt.show()
```

```
1: import numpy as np
2: import matplotlib.pyplot as plt
3:
4: # PDE:
5: #  $-(\lambda(x)u'(x))' = 0$   $x$  in  $(0,1)$ 
6: #  $u(0) = 0$ 
7: #  $u(1) = 1$ 
8: #  $\lambda(x) = \begin{cases} 1 & x \text{ in } (0, 1/\sqrt{2}) \\ 10 & x \text{ in } (1/\sqrt{2}, 1) \end{cases}$ 
9: #
10:
11: # parameters
12: n = 2 # elements (must be even)
13:
14: # mesh
15: m = n+1 # nodes
16: nh = int(n/2) # elements per subdomain
17: mh = nh+1 # nodes per subdomain
18: jump = 1/np.sqrt(2)
19: x1 = np.linspace(0, jump, mh)
20: x2 = np.linspace(jump, 1, mh)
21: x = np.concatenate((x1[:-1], x2))
22: h1 = jump/nh
23: h2 = (1-jump)/nh
24:
25: # local stiffness matrix
26: K_loc1 = (1.0/h1) * np.array([[1, -1], [-1, 1]])
27: K_loc2 = (10.0/h2) * np.array([[1, -1], [-1, 1]])
28:
29: # Assembling
30: K = np.zeros((m,m))
31: F = np.zeros(m)
32: for i in range(nh):
33:     K[i:i+2, i:i+2] += K_loc1
34: for i in range(nh, n):
35:     K[i:i+2, i:i+2] += K_loc2
36:
37: # Boundary conditions
38: # Dirichlet:  $u(0) = 0$ 
39: K[0, :] = 0
40: K[0, 0] = 1
41: F[0] = 0
42: # Dirichlet:  $u(1) = 1$ 
43: K[-1, :] = 0
44: K[-1, -1] = 1
45: F[-1] = 1
46:
47: u = np.linalg.solve(K, F)
48: plt.plot(x, u, "-o", label="u_h")
49: plt.title(f"n = {n} (already exact)")
50: plt.xlabel("x")
51: plt.ylabel("u(x)")
52: plt.legend()
53: plt.grid(True)
54:
55: print("K = ", K)
56: print("f = ", F)
57: print("u = ", u)
58:
59: plt.tight_layout()
60: plt.savefig("../task_b.png", dpi=300)
61: plt.show()
```

```
1: import numpy as np
2: import matplotlib.pyplot as plt
3:
4: # Peclet problem:
5: #  $-u''(x) + pu'(x) = 0$   $x$  in  $(0,1)$ 
6: #  $u(0) = 0$ 
7: #  $u(1) = 1$ 
8:
9: # parameters
10: p = 70
11: n_values = [10,20,30,40,70] # elements
12:
13: # exact solution
14: x_exact = np.linspace(0,1,1000)
15: u_exact = (np.exp(p*x_exact)-1)/(np.exp(p)-1)
16:
17: for n in n_values:
18:     # mesh
19:     m = n+1 # nodes
20:     h = 1.0/n
21:     x = np.linspace(0,1,m)
22:
23:     # local stiffness matrix
24:     K_loc = np.zeros((2,2))
25:     A = (1.0/h) * np.array([[ 1,-1], [-1, 1]])
26:     B = (p/2) * np.array([[ -1, 1], [ -1, 1]])
27:     K_loc = A+B
28:
29:     # Assembling
30:     K = np.zeros((m,m))
31:     F = np.zeros(m)
32:     for i in range(n):
33:         K[i:i+2,i:i+2] += K_loc
34:
35:     # Boundary conditions
36:     # Dirichlet:  $u(0) = 0$ 
37:     K[0,:] = 0
38:     K[0,0] = 1
39:     F[0] = 0
40:     # Dirichlet:  $u(1) = 1$ 
41:     K[-1,:] = 0
42:     K[-1,-1] = 1
43:     F[-1] = 1
44:
45:     u = np.linalg.solve(K, F)
46:     plt.plot(x, u, "-o", markersize=2, label=f"n = {n}")
47:
48: plt.plot(x_exact, u_exact, "black", label="exact")
49: plt.title(f"p = {p}")
50: plt.xlabel("x")
51: plt.ylabel("u(x)")
52: plt.legend()
53: plt.grid(True)
54:
55: print("K = ", K)
56: print("f = ", F)
57: print("u = ", u)
58:
59: plt.tight_layout()
60: plt.savefig("../task_c.png", dpi=300)
61: plt.show()
```