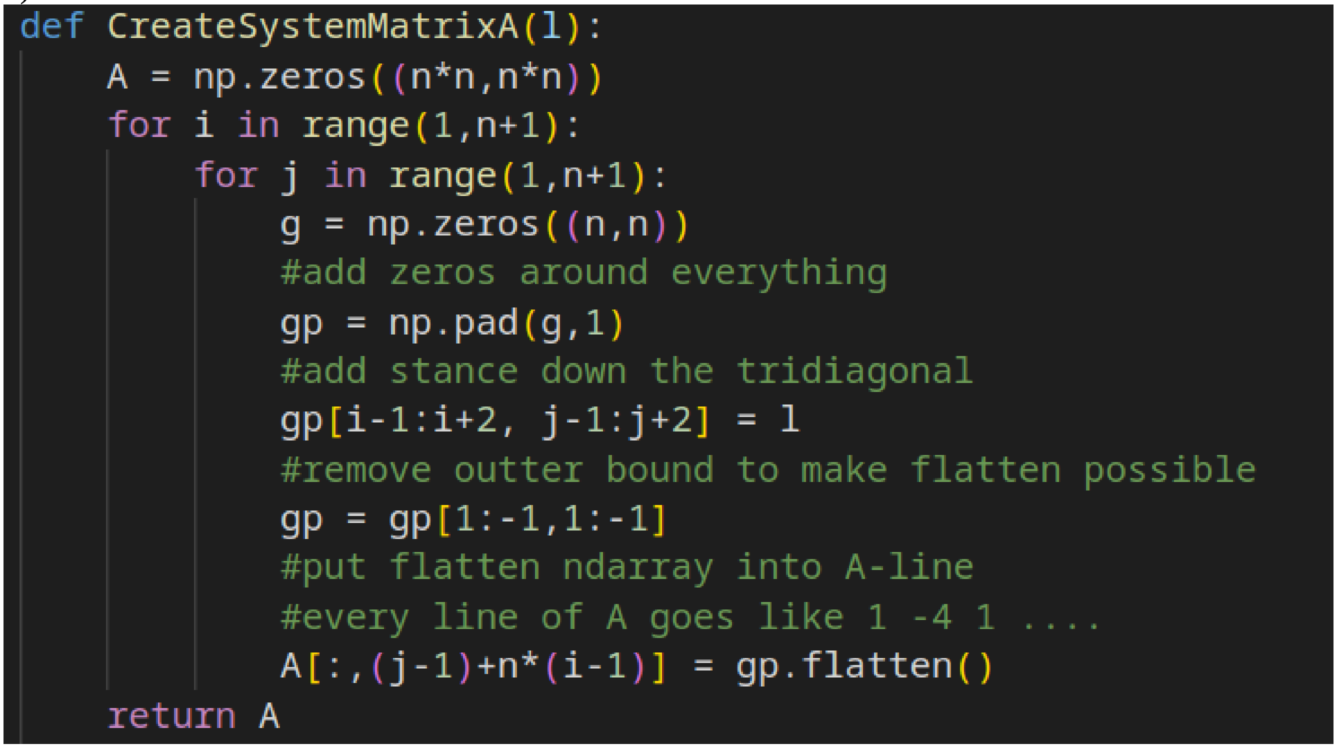
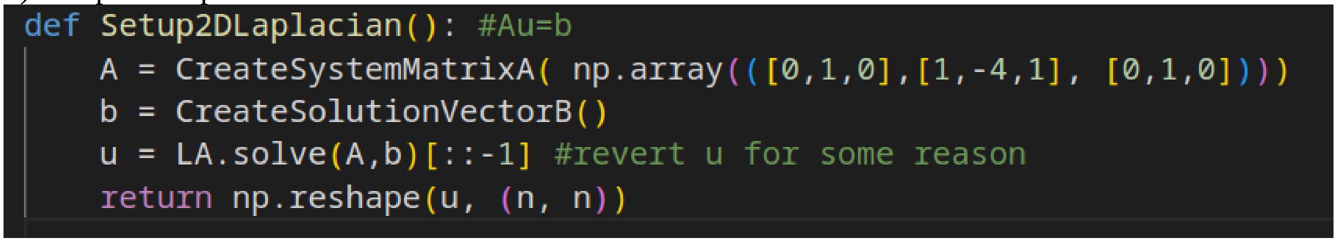
1) Define 2D Grid



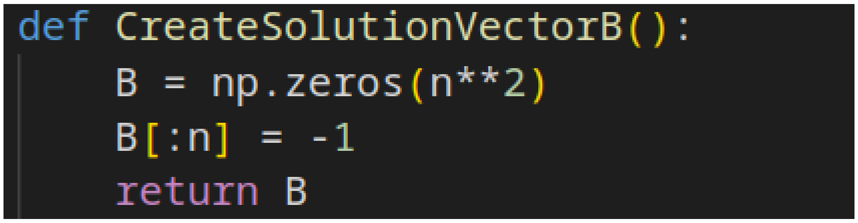
```python
def CreateSystemMatrixA(l):
    A = np.zeros((n*n,n*n))
    for i in range(1,n+1):
        for j in range(1,n+1):
            g = np.zeros((n,n))
            #add zeros around everything
            gp = np.pad(g,1)
            #add stance down the tridiagonal
            gp[i-1:i+2, j-1:j+2] = l
            #remove outter bound to make flatten possible
            gp = gp[1:-1,1:-1]
            #put flatten ndarray into A-line
            #every line of A goes like 1 -4 1 ....
            A[:,(j-1)+n*(i-1)] = gp.flatten()
    return A
```

2) Setup 2D Laplacian and solve



```python
def Setup2DLaplacian(): #Au=b
    A = CreateSystemMatrixA( np.array(([0,1,0],[1,-4,1], [0,1,0])))
    b = CreateSolutionVectorB()
    u = LA.solve(A,b)[::-1] #revert u for some reason
    return np.reshape(u, (n, n))
```

3) Boundary conditions



```python
def CreateSolutionVectorB():
    B = np.zeros(n**2)
    B[:n] = -1
    return B
```

4) Results

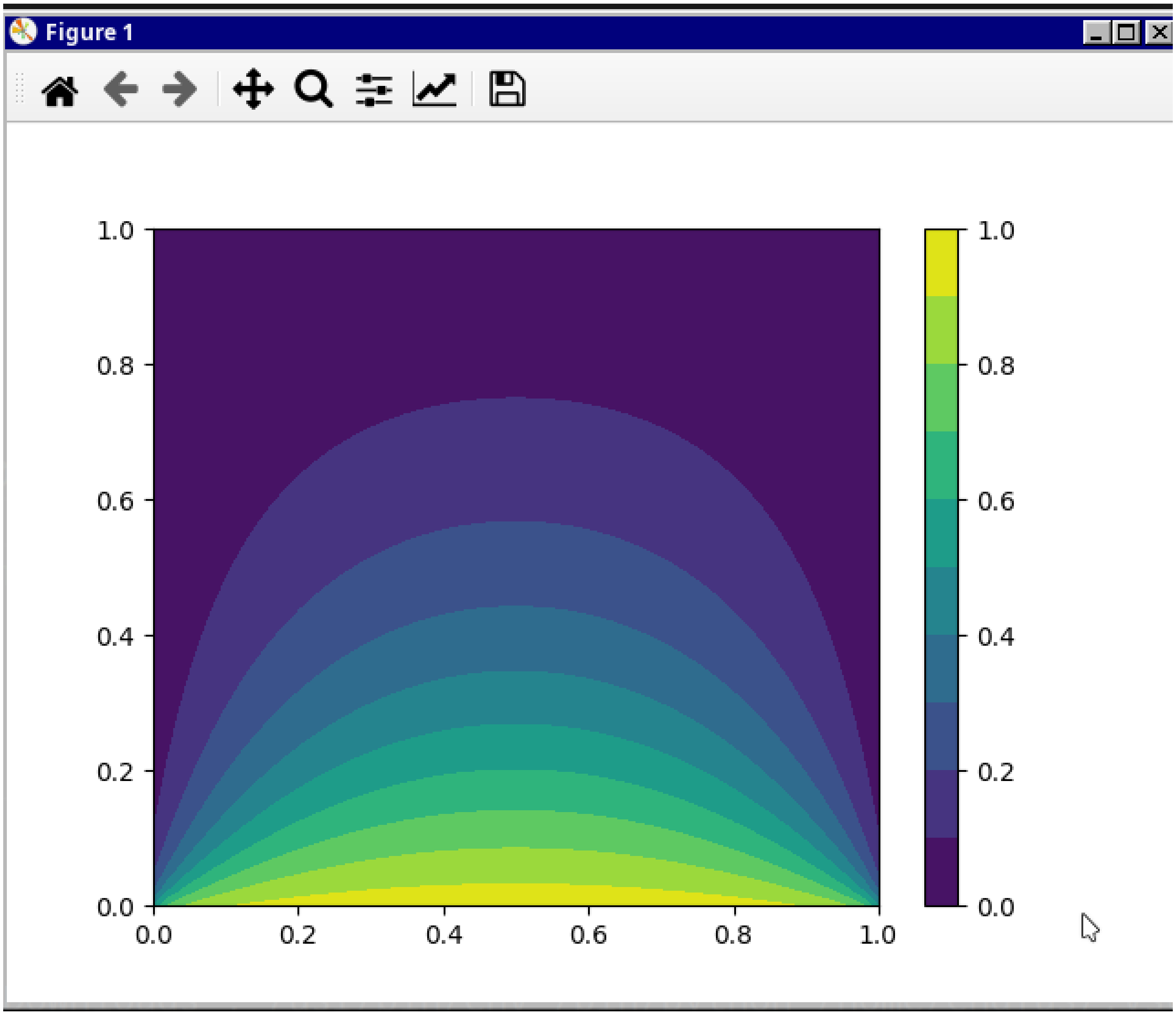


Figure 1
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0