```
import numpy as np
import matplotlib.pyplot as plt
```

```
def step(x):
    return (x >= 0).astype(int)
```

```
def perceptron(X, w, b):
    return step(X.dot(w) + b)
```

```
w_or   = np.array([0.5, 0.5]); b_or   = -0.2
w_nand = np.array([-0.5,-0.5]); b_nand = 0.7
# output: AND of (OR, NAND)
w_and_out = np.array([0.5, 0.5]); b_and_out = -0.7
```

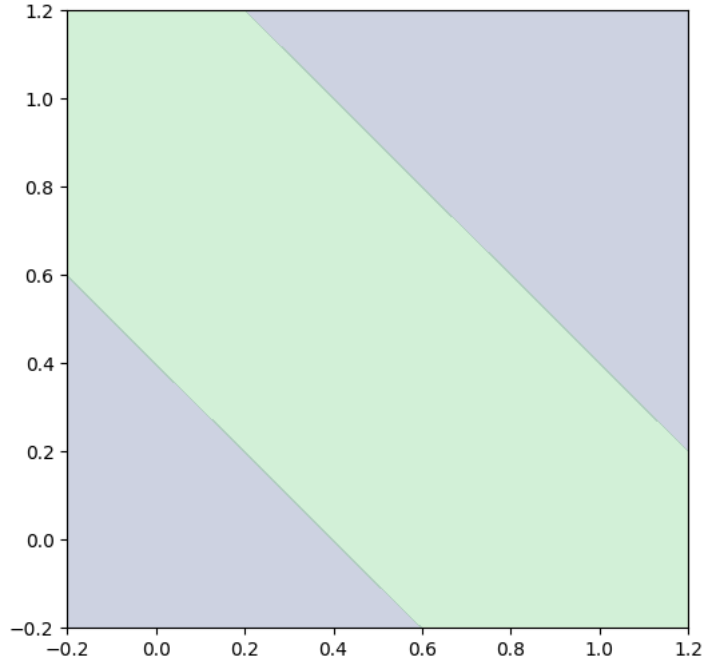
```
X = np.array([[0,0],[0,1],[1,0],[1,1]])
h1 = perceptron(X, w_or, b_or)      # OR
h2 = perceptron(X, w_nand, b_nand)  # NAND
y = perceptron(np.column_stack((h1,h2)), w_and_out, b_and_out)
```

```
print("A B | OR NAND | XOR")
for (a,b),o1,o2,yy in zip(X,h1,h2,y):
    print(f"{a} {b} |  {o1}    {o2}  |  {yy}")
```

```
A B | OR NAND | XOR
0 0 |  0    1  |  0
0 1 |  1    1  |  1
1 0 |  1    1  |  1
1 1 |  1    0  |  0
```

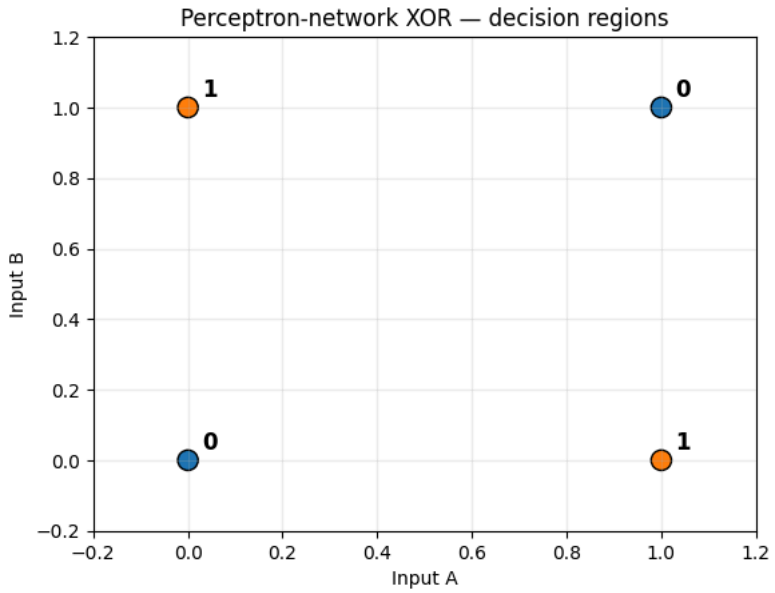
```
xx, yy = np.meshgrid(np.linspace(-0.2,1.2,300), np.linspace(-0.2,1.2,300))
grid = np.c_[xx.ravel(), yy.ravel()]

g_h1 = perceptron(grid, w_or, b_or)
g_h2 = perceptron(grid, w_nand, b_nand)
g_y  = perceptron(np.column_stack((g_h1,g_h2)), w_and_out, b_and_out)
zz = g_y.reshape(xx.shape)

plt.figure(figsize=(6,6))
plt.contourf(xx, yy, zz, levels=[-0.5,0.5,1.5], alpha=0.25)
```

```
<matplotlib.contour.QuadContourSet at 0x7e6ee9dfc470>
```




```
colors = ['C0' if lab==0 else 'C1' for lab in y]
plt.scatter(X[:,0], X[:,1], c=colors, s=120, edgecolor='k')
for (xi,yi),lab in zip(X,y):
    plt.text(xi+0.03, yi+0.03, f"{lab}", fontsize=12, weight='bold')
plt.title("Perceptron-network XOR — decision regions")
plt.xlabel("Input A"); plt.ylabel("Input B")
plt.xlim(-0.2,1.2); plt.ylim(-0.2,1.2)
plt.grid(alpha=0.25)
```

```
plt.show()
```


Perceptron-network XOR — decision regions
Input A
Input B
1
0
0
1
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2