import numpy as np

import matplotlib.pyplot as plt

from scipy.ndimage import laplace

# --- PARAMETERS ---

N = 60 # grid size

timesteps = 240 # more steps for escape

barrier\_prob = 0.23 # tune for more/less percolation

n\_runs = 10 # how many universes to simulate

trans\_history = []

for run in range(n\_runs):

# Random barrier landscape

rng = np.random.default\_rng(run \* 1357 + 42)

V = np.zeros((N, N))

mask = rng.random((N, N)) < barrier\_prob

V[mask] = 1.1 + 0.5 \* rng.random(np.count\_nonzero(mask))

# Initial wavepacket (low k for “stealth”)

x0, y0 = N//6, N//2

width = 6

kx, ky = 0.18, 0

X, Y = np.meshgrid(np.arange(N), np.arange(N), indexing='ij')

psi = np.exp(-((X-x0)\*\*2 + (Y-y0)\*\*2) / (2\*width\*\*2)) \* np.exp(1j \* (kx\*X + ky\*Y))

psi = psi / np.sqrt(np.sum(np.abs(psi)\*\*2)) # normalize

# Track transmission at right edge each step

transmissions = []

for t in range(timesteps):

lap = laplace(psi, mode='wrap')

psi += 0.14 \* (0.58 \* lap - 0.56 \* V \* psi)

norm = np.sqrt(np.sum(np.abs(psi)\*\*2))

if norm != 0:

psi /= norm

# Measure right edge

edge = np.abs(psi[-1, :])\*\*2

transmissions.append(np.sum(edge))

trans\_history.append(transmissions)

# --- PLOT ---

plt.figure(figsize=(9,5))

for i, trans in enumerate(trans\_history):

plt.plot(trans, label=f"Run {i+1}", alpha=0.7)

plt.xlabel("Time Step")

plt.ylabel("Transmission at Right Edge")

plt.title(f"MBT 2D Quantum Percolation — Transmission Across {n\_runs} Mazes")

plt.legend(ncol=2, fontsize=8, loc='upper right')

plt.tight\_layout()

plt.show()