import numpy as np

import matplotlib.pyplot as plt

from scipy.integrate import trapezoid

# --- Parameters ---

grid\_size = 120

timesteps = 180

dt = 0.12

width = 6

r = np.linspace(0, grid\_size, grid\_size)

dr = r[1] - r[0]

# --- Barrier: Center and width stay fixed, height oscillates ---

barrier\_center = 60

barrier\_width = 6

barrier\_base\_height = 0.07

barrier\_amplitude = 0.045 # Oscillation amplitude

barrier\_freq = 0.08 # Oscillation frequency (rad/step)

# --- MBT Breathing Seed Function ---

def breathing\_seed(center, freq, base\_phase, t):

phase = base\_phase + 0.5 \* np.sin(freq \* t)

return np.exp(-((r - center)\*\*2)/(2\*width\*\*2)) \* np.exp(1j \* phase)

# --- Evolution Function ---

def evolve\_packet(x0, k0, seed\_center, seed\_base\_phase, seed\_freq):

ψ = np.exp(-((r - x0)\*\*2)/(2\*width\*\*2)) \* np.exp(1j \* k0 \* r)

ψ\_total = []

barrier\_profiles = []

for t in range(timesteps):

# Oscillating barrier height

barrier\_height = barrier\_base\_height + barrier\_amplitude \* np.sin(barrier\_freq \* t)

V = np.zeros\_like(r)

V[(r > barrier\_center - barrier\_width/2) & (r < barrier\_center + barrier\_width/2)] = barrier\_height

barrier\_profiles.append(V.copy())

# MBT breathing seed (internal curvature)

ψ\_internal = breathing\_seed(seed\_center, seed\_freq, seed\_base\_phase, t)

lap\_int = np.zeros\_like(ψ, dtype=complex)

lap\_int[1:-1] = (ψ\_internal[2:] - 2\*ψ\_internal[1:-1] + ψ\_internal[:-2]) / dr\*\*2

lap\_ψ = np.zeros\_like(ψ, dtype=complex)

lap\_ψ[1:-1] = (ψ[2:] - 2\*ψ[1:-1] + ψ[:-2]) / dr\*\*2

ψ += dt \* (0.65 \* lap\_ψ - 0.5 \* V \* ψ + 0.65 \* lap\_int)

norm = np.sqrt(trapezoid(np.abs(ψ)\*\*2, r))

if norm != 0:

ψ /= norm

if t % 18 == 0:

ψ\_total.append(np.abs(ψ)\*\*2)

final = np.abs(ψ)\*\*2

T = trapezoid(final[r > (barrier\_center + barrier\_width)], r[r > (barrier\_center + barrier\_width)])

R = trapezoid(final[r < (barrier\_center - barrier\_width)], r[r < (barrier\_center - barrier\_width)])

return ψ\_total, barrier\_profiles, T, R

# --- Run experiment ---

snapshots, barrier\_profiles, T, R = evolve\_packet(

x0=25, k0=1.25, seed\_center=barrier\_center, seed\_base\_phase=0, seed\_freq=0.07

)

# --- Plot Results ---

fig, axs = plt.subplots(2, 1, figsize=(14,8))

# Evolution plot

for i, ψs in enumerate(snapshots):

axs[0].plot(r, ψs, alpha=0.5)

# Plot oscillating barrier at each snapshot

snap\_indices = np.linspace(0, timesteps-1, len(snapshots)).astype(int)

for i, idx in enumerate(snap\_indices):

V = barrier\_profiles[idx]

axs[0].plot(r, V / np.max([np.max(s) for s in snapshots]) \* np.max([np.max(s) for s in snapshots]), 'k--', alpha=0.15)

axs[0].set\_title("MBT Dispatch Tunneling with Breathing Barrier")

axs[0].set\_ylabel("Probability Density")

# Final transmission/reflection bar

axs[1].bar(['Transmission', 'Reflection'], [T, R], color=['lightgreen', 'gold'])

axs[1].set\_ylim(0, 1)

axs[1].set\_ylabel("Probability")

axs[1].set\_title("Final Probabilities")

plt.tight\_layout()

plt.show()