import numpy as np

import matplotlib.pyplot as plt

from scipy.integrate import trapezoid

# --- Grid and 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 Definition ---

V = np.zeros\_like(r)

V[(r > 50) & (r < 56)] = 0.08

V[(r > 62) & (r < 72)] = 0.04

# --- Trailing Seed Function ---

def trailing\_seed(t, freq, base\_phase, path, delay=12.0):

center = path(t) - delay # Seed trails ψ₁ by 'delay' units

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

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

# --- Dispatch Evolution with Trailing Seed ---

def evolve\_trailing(base\_phase=np.pi, freq=0.06, x0=95, k0=-1.2, delay=12.0):

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

ψ\_total = []

# ψ₁’s estimated path

def packet\_path(t):

return x0 + k0 \* dt \* t

for t in range(timesteps):

ψ\_internal = trailing\_seed(t, freq, base\_phase, packet\_path, delay=delay)

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 % 15 == 0:

ψ\_total.append(np.abs(ψ)\*\*2)

final = np.abs(ψ)\*\*2

T = trapezoid(final[r > 85], r[r > 85])

R = trapezoid(final[r < 35], r[r < 35])

return ψ\_total, T, R

# --- Run the trailing seed experiment ---

snap\_trail, T\_trail, R\_trail = evolve\_trailing()

# --- Plot Results ---

fig, axs = plt.subplots(2, 2, figsize=(15,8))

# ψ₁ Evolution Snapshots

for ψs in snap\_trail:

axs[0,0].plot(r, ψs, alpha=0.5)

axs[0,0].plot(r, V / np.max(V) \* np.max([np.max(p) for p in snap\_trail]), 'k--', lw=2)

axs[0,0].set\_title("Trailing Seed: ψ₁ Wake Coupling")

# Transmission Bar

axs[0,1].bar(['→Forward'], [T\_trail], color='lightgreen')

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

axs[0,1].set\_ylabel("Transmission Probability")

axs[0,1].set\_title("Dispatch Access with Trailing Seed")

# Reflection Bar

axs[1,1].bar(['→Forward'], [R\_trail], color='gold')

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

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

axs[1,1].set\_title("Dispatch Rejection with Trailing Seed")

axs[1,0].axis('off') # Unused subplot

plt.tight\_layout()

plt.show()