import numpy as np

import matplotlib.pyplot as plt

# Parameters

N = 32 # Lattice size

timesteps = 200 # Time evolution

external\_phases = np.linspace(0, 2\*np.pi, 32) # "Flux sweep" in SQUID

noise = 0.12 # Temperature / decoherence

# MBT phase and memory fields

def mbt\_init():

phase = np.random.uniform(-np.pi, np.pi, (N, N))

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

return phase, memory

def mbt\_evolve(phase, memory, flux\_phase, j1=0.24, j2=0.24, barrier\_locs=[(N//2,0),(N//2,N-1)]):

new\_phase = phase.copy()

for i in range(N):

for j in range(N):

neighbors = []

for di, dj in [(-1,0),(1,0),(0,-1),(0,1)]:

ni, nj = (i+di)%N, (j+dj)%N

# Two Josephson junctions ("barriers" with different coupling)

if (i,j) in barrier\_locs and (ni,nj) in barrier\_locs:

if (i,j)==barrier\_locs[0] and (ni,nj)==barrier\_locs[1]:

coupling = j2

phase\_shift = flux\_phase

elif (i,j)==barrier\_locs[1] and (ni,nj)==barrier\_locs[0]:

coupling = j1

phase\_shift = -flux\_phase

else:

coupling = 0.15

phase\_shift = 0

elif (i,j) in barrier\_locs or (ni,nj) in barrier\_locs:

coupling = 0.15

phase\_shift = 0

else:

coupling = 1.0

phase\_shift = 0

delta = phase[ni, nj] - phase[i, j] + phase\_shift

neighbors.append(coupling \* np.sin(delta))

drift = sum(neighbors) + noise\*np.random.randn()

new\_phase[i,j] += 0.15 \* drift

memory[i,j] = 0.96 \* memory[i,j] + 0.04 \* np.cos(phase[i,j])

return new\_phase, memory

# Simulate SQUID response

def measure\_critical\_current(flux\_vals):

critical = []

for ext\_phase in flux\_vals:

phase, memory = mbt\_init()

for t in range(timesteps):

phase, memory = mbt\_evolve(phase, memory, ext\_phase)

# "Order parameter" = max phase coherence across junction

p1 = phase[N//2,0]

p2 = phase[N//2,N-1]

order = np.abs(np.cos(p1-p2))

critical.append(order)

return np.array(critical)

# Run

flux\_vals = external\_phases

critical\_current = measure\_critical\_current(flux\_vals)

# Visualize: Final phase/memory and SQUID oscillation

phase, memory = mbt\_init()

for t in range(timesteps):

phase, memory = mbt\_evolve(phase, memory, flux\_vals[8])

fig, axs = plt.subplots(1,3, figsize=(15,4))

axs[0].imshow(np.cos(phase), cmap='twilight', interpolation='nearest')

axs[0].set\_title('Final MBT Phase Field (cos)')

axs[1].imshow(memory, cmap='inferno', interpolation='nearest')

axs[1].set\_title('Final MBT Memory Field')

axs[2].plot(flux\_vals/(2\*np.pi), critical\_current, label='MBT Critical Current')

axs[2].set\_xlabel('Applied Phase / Flux (cycles)')

axs[2].set\_ylabel('Order (SQUID response)')

axs[2].set\_title('MBT “SQUID” Oscillation\n(First Principles Only)')

axs[2].legend()

plt.tight\_layout()

plt.show()