import numpy as np

import matplotlib.pyplot as plt

# Parameters

timesteps = 201

# Always get exactly timesteps points!

collapse = np.concatenate([np.linspace(0, 1, timesteps//2), np.linspace(1, 0, timesteps - timesteps//2)])

noise\_strength = 0.2 # Try 0.0 (no noise), 0.1, 0.2, up to 0.5 for wildness

# Bell S (order) and MBT memory law

S\_quantum = 2.828

S\_classical = 2.0

S\_mbt = S\_quantum - (S\_quantum - S\_classical) \* collapse

# MBT memory law (mirrors order)

memory\_mbt = 1.0 - collapse

# Add noise: Gaussian random fluctuations

np.random.seed(42) # For repeatability

S\_noisy = S\_mbt + np.random.normal(0, noise\_strength, size=timesteps)

memory\_noisy = memory\_mbt + np.random.normal(0, noise\_strength, size=timesteps)

# Clip within physical range

S\_noisy = np.clip(S\_noisy, S\_classical, S\_quantum)

memory\_noisy = np.clip(memory\_noisy, 0, 1)

# Plot both (clean vs noisy)

fig, ax1 = plt.subplots(figsize=(8,5))

ax1.plot(S\_mbt, label='Bell S (Clean)', color='tab:blue', linewidth=2)

ax1.plot(S\_noisy, label='Bell S (Noisy)', color='tab:blue', linestyle='--')

ax1.axhline(S\_classical, color='grey', linestyle='--', label='Classical Limit')

ax1.axhline(S\_quantum, color='purple', linestyle='--', label='Quantum Limit')

ax1.set\_ylabel('Bell S (Order Parameter)')

ax1.set\_xlabel('Time (collapse → healing)')

ax2 = ax1.twinx()

ax2.plot(memory\_mbt, label='MBT Memory (Clean)', color='tab:green', linewidth=2)

ax2.plot(memory\_noisy, label='MBT Memory (Noisy)', color='tab:green', linestyle='--')

ax2.set\_ylabel('MBT Memory')

fig.suptitle('Decoherence/Noise: MBT Order and Memory (Quantum to Classical Transition)')

lines1, labels1 = ax1.get\_legend\_handles\_labels()

lines2, labels2 = ax2.get\_legend\_handles\_labels()

ax1.legend(lines1 + lines2, labels1 + labels2, loc='upper right')

plt.tight\_layout()

plt.savefig('mbt\_noise\_experiment.png', dpi=150)

plt.show()