import numpy as np

import matplotlib.pyplot as plt

def MMSE(y, H, noise\_var, signal\_var, num\_antennas, num\_users):

    x\_mmse = np.linalg.inv(noise\_var / signal\_var \* np.eye(num\_users) + np.dot(np.conj(H.T), H)).dot(np.dot(np.conj(H.T), y))

    x\_mmse = np.sign(np.real(x\_mmse)) + 1j \* np.sign(np.imag(x\_mmse))

    return x\_mmse

def calculate\_BER(x, x\_est):

    return np.mean(x != x\_est)

# Function to generate additive white Gaussian noise (AWGN)

def awgn\_noise(signal, noise\_power):

    noise = np.random.randn(\*signal.shape) \* np.sqrt(noise\_power)

    return noise

# Function to simulate Massive MIMO system with MMSE detection

def simulate\_mimo\_system(num\_antennas, num\_users, modulation\_order, snr\_db, num\_trials=10000):

    # Generate synthetic data

    transmitted\_symbols = (2 \* np.random.randint(0, 2, size=(num\_users, 1)) - 1) + 1j \* (2 \* np.random.randint(0, 2, size=(num\_users, 1)) - 1)

    # Initialize lists to store Bit Error Rate (BER) for each SNR value

    snr\_values\_db = np.arange(0, 16, 2)  # SNR range from 0 dB to 15 dB

    ber\_values = []

    for snr\_db in snr\_values\_db:

        noise\_var = 2 \* num\_users / num\_antennas \* 10\*\*(-snr\_db / 10)  # noise variance controlled by SNR in dB

        signal\_var = 2  # signal variance in QPSK

        num\_errors = 0

        for \_ in range(num\_trials):

            # Generate random channel matrix H

            H = 1 / np.sqrt(2 \* num\_antennas) \* np.random.randn(num\_antennas, num\_users) + 1j / np.sqrt(2 \* num\_antennas) \* np.random.randn(num\_antennas, num\_users)

            # Generate AWGN noise

            w = awgn\_noise(np.dot(H, transmitted\_symbols), noise\_var)

            # Received signal

            received\_signal = np.dot(H, transmitted\_symbols) + w

            # MMSE detection

            estimated\_symbols = MMSE(received\_signal, H, noise\_var, signal\_var, num\_antennas, num\_users)

            # Calculate Bit Error Rate (BER)

            num\_errors += np.sum(transmitted\_symbols != estimated\_symbols)

        ber = num\_errors / (num\_trials \* num\_users)

        ber\_values.append(ber)

    return snr\_values\_db, ber\_values

# Parameters

num\_antennas = 16

num\_users = 8

num\_trials = 1000

# Simulate Massive MIMO system with MMSE detection

snr\_values\_db, ber\_values = simulate\_mimo\_system(num\_antennas, num\_users, 2, 10, num\_trials)

# Plot SNR vs BER

plt.figure()

plt.semilogy(snr\_values\_db, ber\_values, marker='o', linestyle='-')

plt.grid(True)

plt.xlabel('SNR (dB)')

plt.ylabel('BER')

plt.title('SNR vs. BER (Massive MIMO with MMSE detection)')

plt.show()

import numpy as np

import matplotlib.pyplot as plt

def ZF(y, H):

    x\_zf = np.linalg.pinv(H).dot(y)

    x\_zf = np.sign(np.real(x\_zf)) + 1j \* np.sign(np.imag(x\_zf))

    return x\_zf

def calculate\_BER(x, x\_est):

    return np.mean(x != x\_est)

# Function to generate additive white Gaussian noise (AWGN)

def awgn\_noise(signal, noise\_power):

    noise = np.random.randn(\*signal.shape) \* np.sqrt(noise\_power)

    return noise

# Function to simulate Massive MIMO system with Zero Forcing (ZF) detection

def simulate\_mimo\_system\_zf(num\_antennas, num\_users, modulation\_order, snr\_db, num\_trials=10000):

    # Generate synthetic data

    transmitted\_symbols = (2 \* np.random.randint(0, 2, size=(num\_users, 1)) - 1) + 1j \* (2 \* np.random.randint(0, 2, size=(num\_users, 1)) - 1)

    # Initialize lists to store Bit Error Rate (BER) for each SNR value

    snr\_values\_db = np.arange(0, 16, 2)  # SNR range from 0 dB to 15 dB

    ber\_values = []

    for snr\_db in snr\_values\_db:

        noise\_var = 2 \* num\_users / num\_antennas \* 10\*\*(-snr\_db / 10)  # noise variance controlled by SNR in dB

        num\_errors = 0

        for \_ in range(num\_trials):

            # Generate random channel matrix H

            H = 1 / np.sqrt(2 \* num\_antennas) \* np.random.randn(num\_antennas, num\_users) + 1j / np.sqrt(2 \* num\_antennas) \* np.random.randn(num\_antennas, num\_users)

            # Generate AWGN noise

            w = awgn\_noise(np.dot(H, transmitted\_symbols), noise\_var)

            # Received signal

            received\_signal = np.dot(H, transmitted\_symbols) + w

            # Zero Forcing (ZF) detection

            estimated\_symbols = ZF(received\_signal, H)

            # Calculate Bit Error Rate (BER)

            num\_errors += np.sum(transmitted\_symbols != estimated\_symbols)

        ber = num\_errors / (num\_trials \* num\_users)

        ber\_values.append(ber)

    return snr\_values\_db, ber\_values

# Parameters

num\_antennas = 128

num\_users = 32

num\_trials = 1000

# Simulate Massive MIMO system with Zero Forcing (ZF) detection

snr\_values\_db, ber\_values = simulate\_mimo\_system\_zf(num\_antennas, num\_users, 2, 10, num\_trials)

# Plot SNR vs BER

plt.figure()

plt.semilogy(snr\_values\_db, ber\_values, marker='o', linestyle='-')

plt.grid(True)

plt.xlabel('SNR (dB)')

plt.ylabel('BER')

plt.title('SNR vs. BER (Massive MIMO with Zero Forcing detection)')

plt.show()