گزارش پروژه: شبیه‌سازی سیستم ارتباطی با استفاده از مدولاسیون QAM-16

ارشیا مددی 99655101

مقدمه

این پروژه به بررسی و شبیه‌سازی یک سیستم ارتباطی با استفاده از مدولاسیون QAM -16 (Quadrature Amplitude Modulation) می‌پردازد. هدف اصلی پروژه شامل تولید سمبل‌های تصادفی، مدولاسیون، افزودن نویز، دمدولاسیون و محاسبه نرخ خطا (BER) است. در این گزارش، مراحل انجام کار به تفصیل توضیح داده می‌شود.

مراحل انجام کار

1. وارد کردن کتابخانه‌ها

```
import numpy as np  # Import numpy for numerical operations
import matplotlib.pyplot as plt  # Import matplotlib for plotting graphs
```

- numpy: برای انجام عملیات عددی و تولید داده‌های تصادفی استفاده می‌شود.
- matplotlib: برای رسم نمودارها و تجسم داده‌ها به کار می‌رود.

2. تابع افزودن نویز AWGN

```
def add_awgn_noise(signal, SNR_dB):
    """
    Adds Gaussian noise to the signal based on the specified SNR (Signal-to-Noise Ratio).
    """
    SNR_linear = 10**(SNR_dB / 10)  # Linear SNR
    signal_power = np.mean(np.abs(signal)**2)  # Calculate signal power
```

```
    noise_power = signal_power / SNR_linear  # Calculate noise power based on SNR
    noise = np.sqrt(noise_power / 2) * (np.random.randn(len(signal)) + 1j *
np.random.randn(len(signal)))  # Generate noise
    return signal + noise  # Return the signal with added noise
```

- این تابع نویز گوسی سفید (AWGN) را به سیگنال ورودی اضافه می‌کند. SNR به دسی‌بل مشخص می‌شود و نویز به صورت تصادفی تولید می‌شود.

3. تنظیمات اولیه

```
num_symbols = 1000  # Number of symbols to generate
SNR_values = np.arange(0, 21, 2)  # SNR values from 0 to 20 dB in steps of 2
```

- num_symbols: تعداد سمبل‌های تولید شده را مشخص می‌کند.
- SNR_values: مقادیر SNR از 0 تا 20 دسی‌بل را در گام‌های 2 دسی‌بل تعریف می‌کند.

4. تولید سمبل‌های تصادفی

```
symbols = np.random.randint(0, 16, num_symbols)  # Generates 1000 random
symbols between 0 and 15
```

- این خط 1000 سمبل تصادفی بین 0 تا 15 تولید می‌کند که نمایانگر QAM-16 هستند.

5. مدولاسیون QAM -16

```
modulated = ((2 * (symbols % 4) - 3) + 1j * (2 * (symbols // 4) - 3)) / np.sqrt(10)
```

- سمبل‌ها به مختصات واقعی و موهومی در صفحه مختلط تبدیل می‌شوند. این تبدیل به گونه‌ای انجام می‌شود که قدرت سیگنال کنترل شود.

6. افزودن نویز و رسم نمودار کنستلاسیون

```
SNR = 20  # Set a specific SNR value (20 dB) for visualization
```

```
noisy_signal = add_awgn_noise(modulated, SNR)  # Add AWGN noise to the modulated signal
```

- نویز به سیگنال مدوله شده با SNR مشخص (20 دسی‌بل) اضافه می‌شود.

7. رسم نمودارهای کنستلاسیون

```
plt.figure()

plt.scatter(modulated.real, modulated.imag, c='b', label="Modulated Signal")  # Scatter plot for modulated symbols

plt.title("Constellation Diagram of Modulated Signal")

plt.xlabel("In-phase")

plt.ylabel("Quadrature")

plt.grid()

plt.legend()
```

- نمودار کنستلاسیون برای سیگنال مدوله شده رسم می‌شود.

```
plt.figure()

plt.scatter(noisy_signal.real, noisy_signal.imag, c='g', label="Noisy Signal")  # Scatter plot for noisy signal

plt.scatter(modulated.real, modulated.imag, c='r', label="Transmitted Symbols")  # Scatter plot for transmitted symbols

plt.title("Constellation Diagram of Noisy Signal with Transmitted Symbols")

plt.xlabel("In-phase")

plt.ylabel("Quadrature")
```

```
plt.grid()

plt.legend()
```

- نمودار کنستلاسیون برای سیگنال نویزدار و سمبل‌های ارسالی رسم می‌شود.

8. رسم سیگنال‌ها در زمان

```
plt.figure(figsize=(10, 8))  # Create a larger figure for multiple subplots
```

- یک شکل بزرگ برای نمایش زیرنمودارها ایجاد می‌شود.

```
plt.subplot(2, 2, 1)

plt.plot(modulated.real)

plt.title("Real Part of Modulated Signal")

plt.xlabel("Sample")

plt.ylabel("Amplitude")

plt.grid()
```

- قسمت واقعی سیگنال مدوله شده رسم می‌شود.

```
plt.subplot(2, 2, 2)

plt.plot(modulated.imag)

plt.title("Imaginary Part of Modulated Signal")

plt.xlabel("Sample")

plt.ylabel("Amplitude")

plt.grid()
```

- قسمت موهومی سیگنال مدوله شده رسم می‌شود.

```
plt.subplot(2, 2, 3)

plt.plot(noisy_signal.real)

plt.title("Real Part of Noisy Signal")

plt.xlabel("Sample")

plt.ylabel("Amplitude")

plt.grid()
```

- قسمت واقعی سیگنال نویزدار رسم می‌شود.

```
plt.subplot(2, 2, 4)

plt.plot(noisy_signal.imag)

plt.title("Imaginary Part of Noisy Signal")

plt.xlabel("Sample")

plt.ylabel("Amplitude")

plt.grid()
```

- قسمت موهومی سیگنال نویزدار رسم می‌شود.

9. دمدولاسیون و محاسبه نرخ خطا

```
def demodulate_16qam(received):

  """
```

```
    Demodulates a received signal based on 16-QAM modulation scheme.
    """
    real_part = np.clip(np.round((received.real * np.sqrt(10) + 3) / 2), 0, 3)  # Quantize real part
    imag_part = np.clip(np.round((received.imag * np.sqrt(10) + 3) / 2), 0, 3)  # Quantize imaginary part
    return (real_part + 4 * imag_part).astype(int)  # Mapping back to symbol index
```

- این تابع سیگنال دمدوله شده را بر اساس مدولاسیون QAM-16 محاسبه می‌کند. مختصات واقعی و موهومی به نزدیک‌ترین نقطه کنستلاسیون نزدیک می‌شوند.

```
demodulated = demodulate_16qam(noisy_signal)  # Demodulate the noisy signal
num_errors = np.sum(symbols != demodulated)  # Count the number of errors
error_rate = num_errors / num_symbols  # Calculate the error rate
```

- سیگنال نویزدار دمدوله شده و تعداد خطاها محاسبه می‌شود.

10. تحلیل نرخ خطا برای SNRهای مختلف

```
ber = []  # List to store the Bit Error Rate (BER) for different SNR values
for snr in SNR_values:
    noisy_signal = add_awgn_noise(modulated, snr)  # Add noise to the modulated signal
    demodulated = demodulate_16qam(noisy_signal)  # Demodulate the noisy signal
    num_errors = np.sum(symbols != demodulated)  # Count the number of errors
    ber.append(num_errors / num_symbols)  # Store the error rate for the current SNR
```

- نرخ خطا برای SNRهای مختلف محاسبه و ذخیره می‌شود.

11. رسم منحنی نرخ خطا بر حسب SNR

```
plt.figure()
plt.semilogy(SNR_values, ber, 'o-', linewidth=2)  # Plot BER on logarithmic scale
plt.xlabel("SNR (dB)")
plt.ylabel("Bit Error Rate (BER)")
plt.title("BER vs SNR for 16-QAM")
plt.grid()
```

- نمودار نرخ خطا بر حسب SNR رسم می‌شود.

نتایج

- تعداد خطاها و نرخ خطا برای SNR = 20 دسی‌بل محاسبه و در کنسول نمایش داده می‌شود.
- نمودار نرخ خطا بر حسب SNR نشان‌دهنده رابطه معکوس بین SNR و نرخ خطا است.

نتیجه‌گیری

این پروژه نشان داد که با استفاده از مدولاسیون QAM-16 می‌توان یک سیستم ارتباطی شبیه‌سازی کرد و تأثیر نویز بر روی کیفیت سیگنال را تحلیل نمود. نتایج نشان می‌دهد که با افزایش SNR، نرخ خطا کاهش می‌یابد.