Code for OFDM

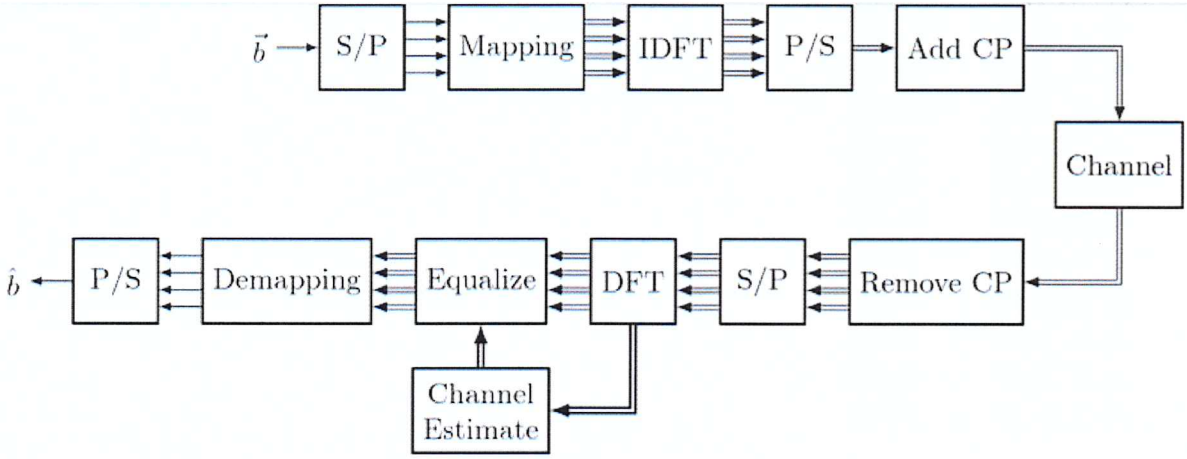
무선통신공학

융합전자공학과

2022236964

태윤섭

OFDM 의 block diagram



각 부분을 python code로 구현

1. 기본적인 parameter 정의

```
K = 64 # number of OFDM subcarriers
CP = K//4  # length of the cyclic prefix: 25% of the block
P = 8 # number of pilot carriers per OFDM block
pilotValue = 3+3j # The known value each pilot transmits
allCarriers = np.arange(K)  # indices of all subcarriers ([0, 1, ... K-1])

pilotCarriers = allCarriers[::K//P] # Pilots is every (K/P)th carrier.

# For convenience of channel estimation, let's make the last carriers also be a pilot
pilotCarriers = np.hstack([pilotCarriers, np.array([allCarriers[-1]])])
P = P+1

# data carriers are all remaining carriers
dataCarriers = np.delete(allCarriers, pilotCarriers)
```

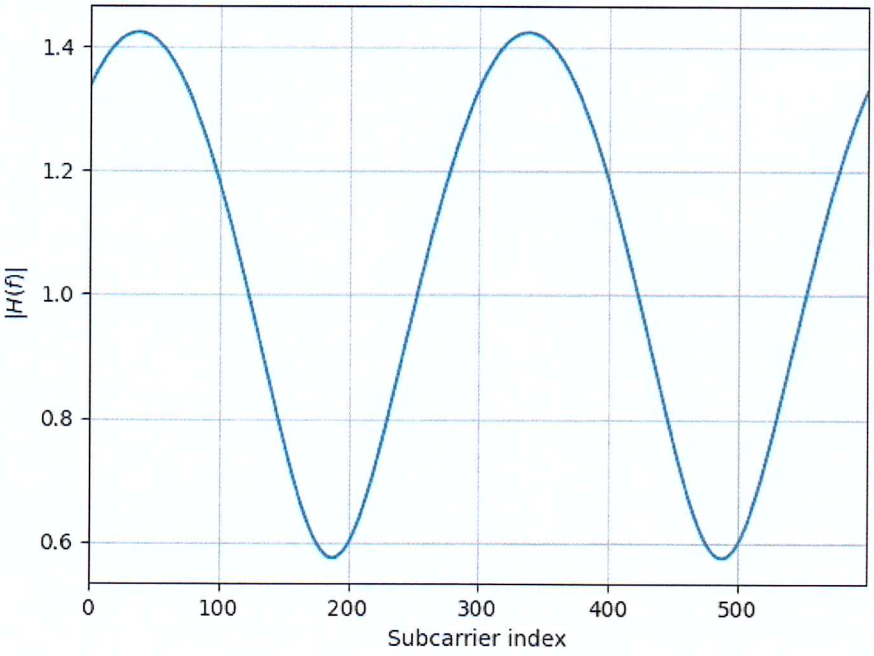
- demapping table 코드

```
demapping_table = {v : k for k, v in mapping_table.items()}
```

- Channel define 코드

```
channelResponse = np.array([1, 0, 0.3+0.3j])  # the impulse response of
 the wireless channel
H_exact = np.fft.fft(channelResponse, K)
plt.plot(allCarriers, abs(H_exact))
plt.xlabel('Subcarrier index'); plt.ylabel('$|H(f)|$'); plt.grid(True);
 plt.xlim(0, K-1)

SNRdb = 25  # signal to noise-ratio in dB at the receiver
```



2. random bit sequence 생성

```
bits = np.random.binomial(n=1, p=0.5, size=(payloadBits_per_OFDM, ))
```

3. Series to Parallel Convertor 정의

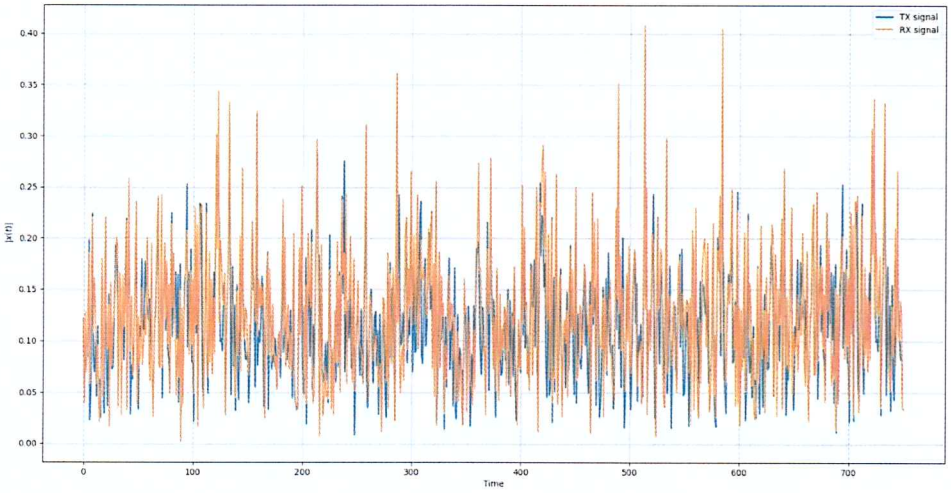
```
def SP(bits):
    return bits.reshape((len(dataCarriers), mu))
bits_SP = SP(bits)
```

8. wireless channel 정의

```
def channel(signal):
    convolved = np.convolve(signal, channelResponse)
    signal_power = np.mean(abs(convolved**2))
    sigma2 = signal_power * 10**(-
SNRdb/10)  # calculate noise power based on signal power and SNR

    print ("RX Signal power: %.4f. Noise power: %.4f" % (signal_power,
sigma2))

    # Generate complex noise with given variance
    noise = np.sqrt(sigma2/2) * (np.random.randn(*convolved.shape)+1j*n
p.random.randn(*convolved.shape))
    return convolved + noise
OFDM_TX = OFDM_withCP
OFDM_RX = channel(OFDM_TX)
```



9. receiver단에서 CP를 제거

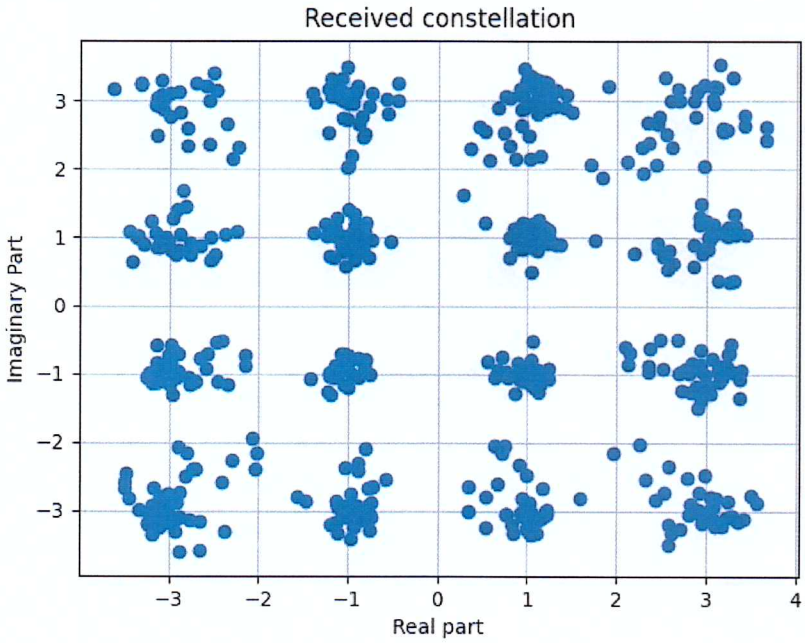
```
def removeCP(signal):
    return signal[CP:(CP+K)]
OFDM_RX_noCP = removeCP(OFDM_RX)
```

10. FFT

```
def DFT(OFDM_RX):
    return np.fft.fft(OFDM_RX)
OFDM_demod = DFT(OFDM_RX_noCP)
```

13. Extract the data carriers from the equalized symbol

```
def get_payload(equalized):
    return equalized[dataCarriers]
QAM_est = get_payload(equalized_Hest)
```



14. Demapping(수신된 constellation 을 matching 시킴)

```
def Demapping(QAM):

    # array of possible constellation points
    constellation = np.array([x for x in demapping_table.keys()])

    # calculate distance of each RX point to each possible point
    dists = abs(QAM.reshape((-1,1)) - constellation.reshape((1,-1)))

    # for each element in QAM, choose the index in constellation
    # that belongs to the nearest constellation point
    const_index = dists.argmin(axis=1)

    # get back the real constellation point
    hardDecision = constellation[const_index]

    # transform the constellation point into the bit groups
    return np.vstack([demapping_table[C] for C in hardDecision]), hardD
ecision

PS_est, hardDecision = Demapping(QAM_est)
```