디지털 통신 최종 과제

2016707044 김영웅

- 기존에 작성한 코드 기반으로 최종 결과물 완성

- 채널 코딩의 경우 본인이 기존에 작업한 것 중 하나 선택

- 송신단에 square root raised cosine pulse shape 사용

- 수신단에 matched filter 사용

- QPSK, 16-QAM 두 가지 변조 방식을 사용하여 수행

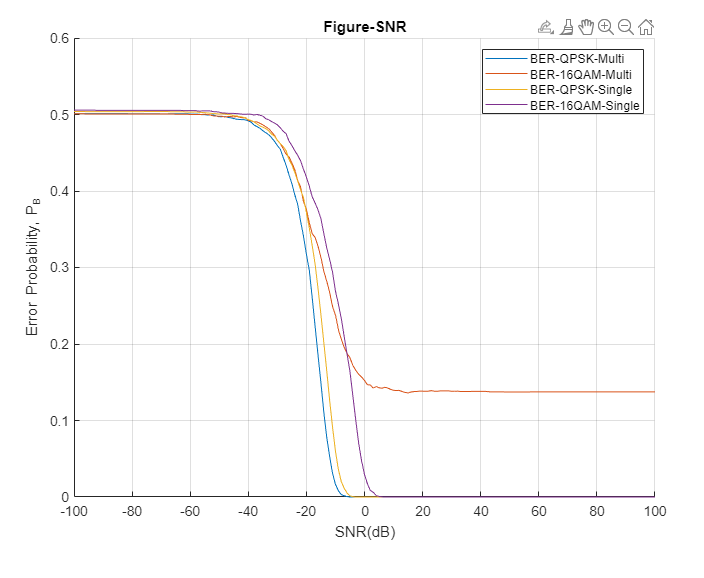
- 전송채널 : AWGN 그리고 AWGN + 수업시간에 이야기 한 다중 경로 채널 두 가지 고려

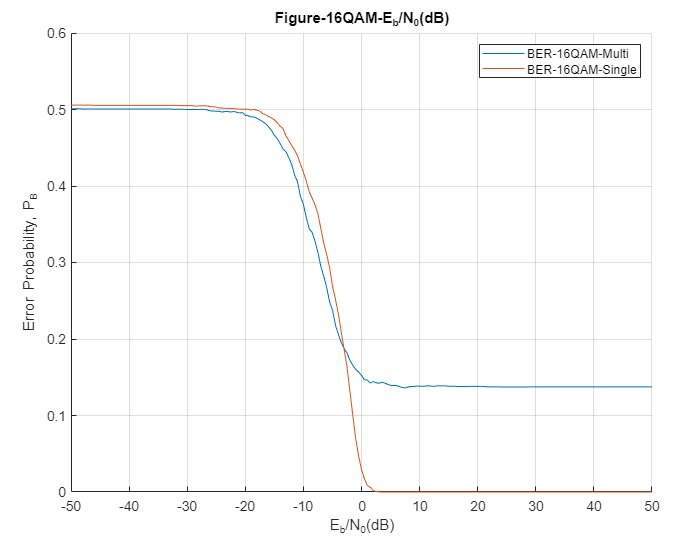
- BER vs SNR 과 BER vs Eb/No 을 그려 보고서 형식으로 제출할 것

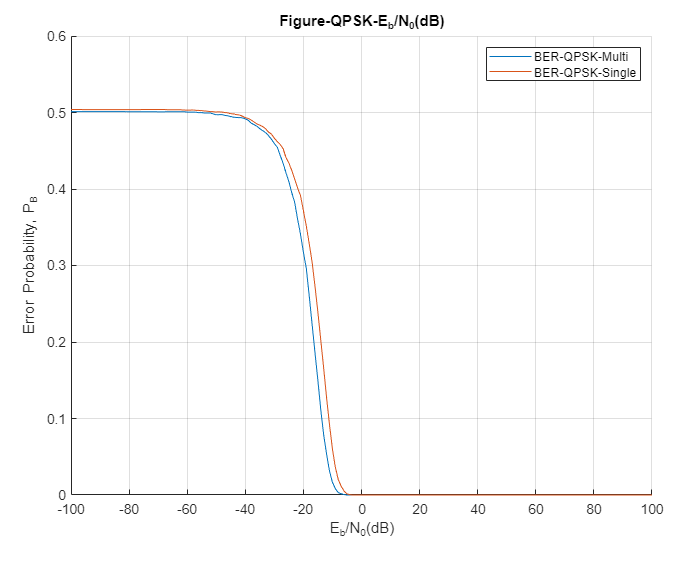
- 매트랩 코드 전체 제출하되, top file을 지정하여 타인이 직접 수행시킬 수 있도록 할 것

결과적으로 여러분이 보고서에 포함하여야 하는 BER curve는 총 8개입니다.

Summary Final graph







input info bits: 2000

payload: 8bits / redundancy: 4bit

codeword: 12bits

square root raised cosine filter

- range [-3 3]

- roll off factor: 0.25

- over sampling rate :8

1. SNR graph

먼저 QPSK가 가장 효율이 좋은 것을 확인할 수 있었다. decision boundary가 QPSK가 16 -QAM에 비해서 넓은 영역에 대해서 판단 기준을 갖는다. QPSK는 한 사분면을 판단 기준으로 삼는데 에 비해, QAM은 한 사분 면 내에서도, 각도와 원점 간의 거리로 총 4 영역으로 나뉘기 때문이다.

single channel과 multichannel의 비교에 있어서는 single channel 보다 multichannel의 SNR이 더 좋은 것을 확인 할 수 있었다. 교수님께서는 ISI 성분을 설명해주시면서 single channel의 SNR이 더 좋다고 설명해 주셨지만, 실제 plotting에서 multichannel의 성능이 더 좋게 표현된 것은, ISI성분으로 작용한 것이 아니라 송신 파워에서 기존 채널에 비해 50% 증가된 파워의 신호를 수신한 것으로 생각한다.

하지만, 16 - QAM에서는 multichannel의 성능이 좋지 못한 것은 decision boundary가 좁기 때문에, 동일한 량만큼 ISI가 생성되었을 때, 다른 decision 영역을 ISI가 침범하기 때문이다.

1. Eb/N0 graph

Eb/NO graph는 SNR graph에서 X-axis를 Eb/No으로 표현하였다. 그렇게 표현하기 위해서 구한 각각 구한 4가지 SNR에 대해서 (C/)을 곱하여 구하였다. 각각 qpsk와 qam에서 M은 4와 16가 되고, C는 8/12로 0.75로 하였다.

고찰:

마지막 과제를 통해서 전체적인 통신의 구조를 총 망라할 수 있었던 것 같다.

information bits가 들어오면 scrambler를 통해서 비트를 섞어주고, 섞인 bits를 cyclic channel coding을 통해서 code rate가 0.75가 되도록 redundancy bit를 더하여 에러에 조금 더 강하게 하였다. 이후 symbol mapping을 통해서 지난번에 하지 않았던 QPSK와 16 – QAM 방식으로 변조를 실행하였고, 이를 송신하기 전에 Pulse mapping으로 square root raised cosine filter를 적용하였다. 이후 교수님께서 말씀해주신 두가지 형태의 채널로 신호를 송신하였고, 수신 부에서는 square root raised cosine filter를 사용하였기 때문에 동일한 matched filter로 square root raised cosine filter를 사용하여 pulse를 수신하였다. 이후, demodulation 과정에서는 각각 QPSK과 16 -QAM에 맞게 QPSK는 각도로 사분면을 판단하였고, 16 -QAM은 각도로 먼저 나누고, 나눠진 구간 내에서 거리를 통해서 해당 Pulse를 decision하였다. 이후 redundancy bits를 통해서 syndrome을 계산하여 해당 syndrome에 맞는 error를 correction해주었고, 마지막 단계로 de – scrambling을 통하여 섞어주었던 비트의 순서를 원래대로 정렬하여 최종적으로 송신했던 신호를 복원하였다.