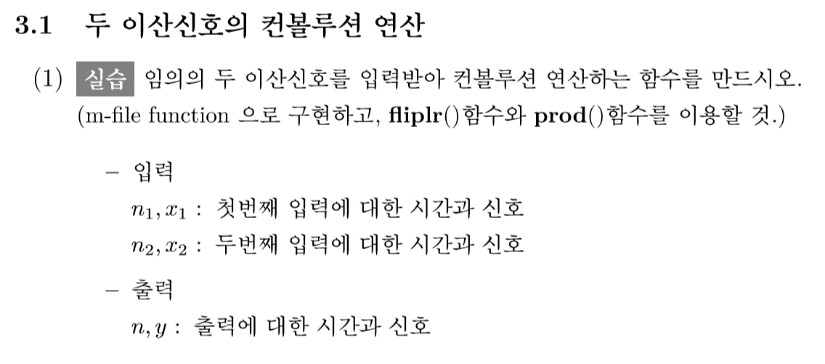
임베디드신호처리실습

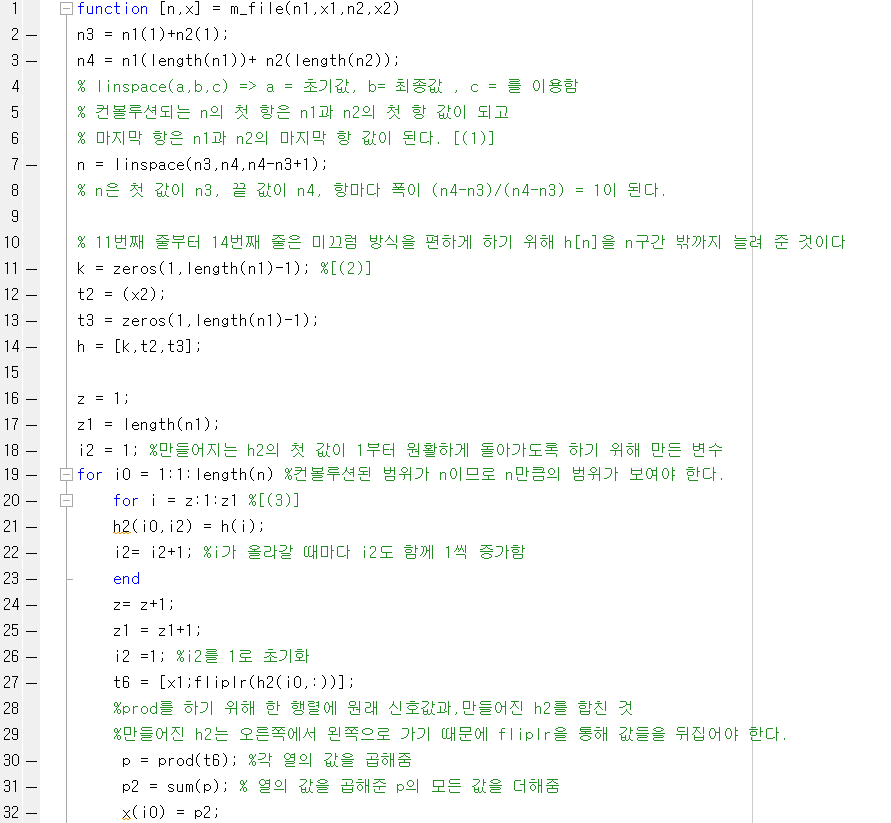
Lab 1. Convolution

결과보고서

2016146026 심재빈

2015146024 오창은

****



1. 이산신호 x1[n]이 n1<= n <= n2에서 정의되고, x2[n]가 n3<=n<=n4에서 정의될 때, 두 이산신호의 컨볼루션은 구간 n1+n3<=n<=n2+n4에서 정의되는 것을 이용하였다.
2. X(n) = {1,2,3,4}, h(n) = (2,3,4,5) n = 0,1,2,3라고 가정할 때

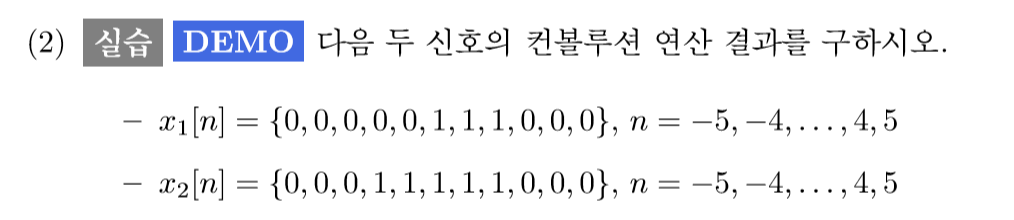
x(k) **1 2 3 4** 미끄럼 방식은 옆 그림과 같이 왼쪽에서 오른쪽으로, n값의 바깥이면 0을 대입하는 방

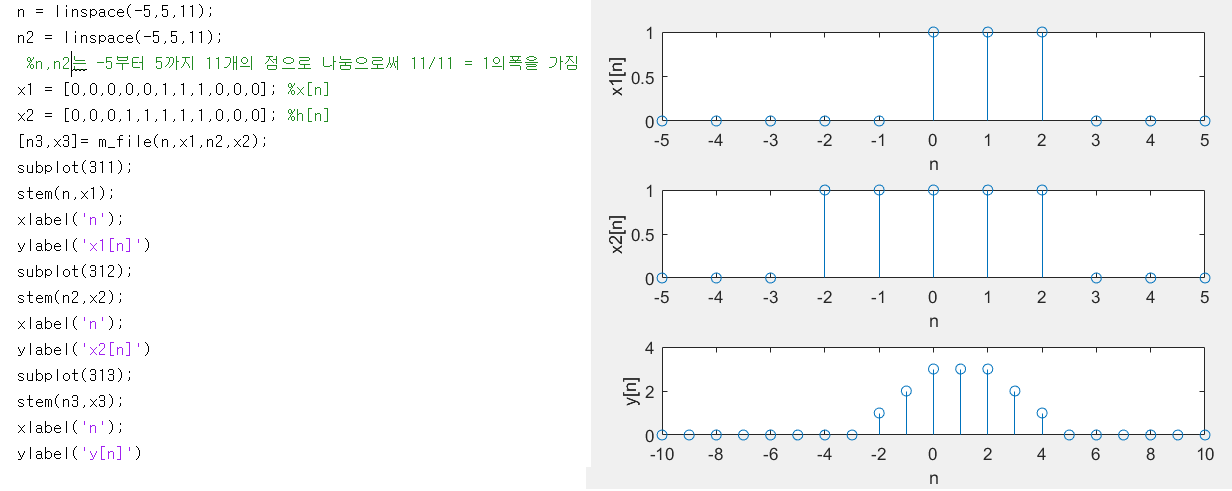
h(-k) 2 0 0 0 식으로 가기 때문에 (X[n]의 범위 크기)-1의 길이만큼 x[0] 앞에 0을(x=h[-1],h[-2]..) 를

h(-k+1) 3 2 0 0 넣어주면 미끄럼 방식의 계산이 편하다. 이와 같이 만들어진 h’(n)의 크기는 (x(n)의 크 기)\*2 + h(n)의 크기 -2 가 된다.

1. (2)에서의 방식을 통해 0을 만들어준 h’(n)은 (0,0,0,2,3,4,5,0,0,0)이 되고 이를 첫 항부터 x(k)의 크기만큼 잘라주기 위해 처음 크기를 z를 1, z2를 x[n]의 범위 크기만큼 설정하고, for문을 통해 h’[n]의 마지막 항까지 빠짐없이 포함시킬 때까지 z와 z2를 1씩 올려준다.

이 이중 for문을 통해 나오는 결과값을 h2[k]라고 하면, h2[0] = (0,0,0,2) , h2[1] = (0,0,2,3) ……h2[6] = (5,0,0,0) ((2)에서의 예시 범위가 0부터 6)과 같은 결과값이 나온다.





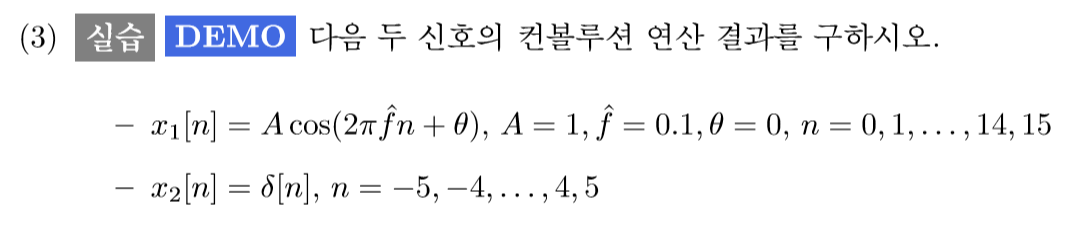
코드는 3-1-(1) 을 이용하여,x1과 x2를 컨볼루션 연산하였다.

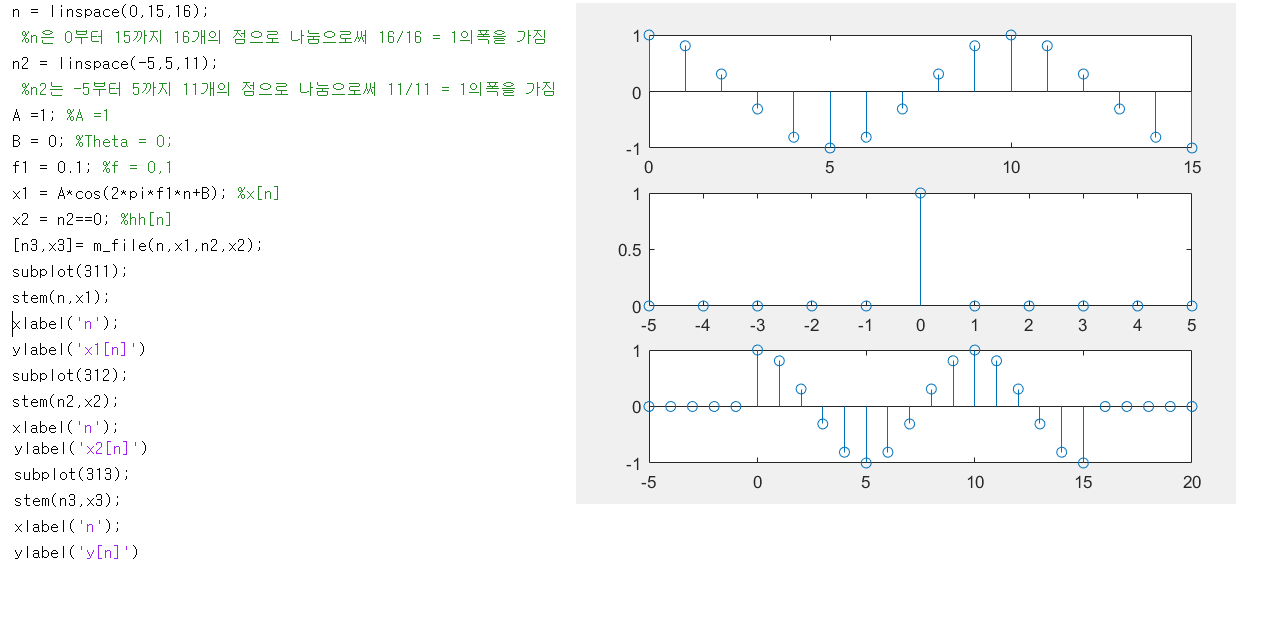
표현은 이산 시퀀스 데이터 플로팅 함수인 stem을 사용하였다.

이 코드를 실행시키면 오른쪽의 그래프가 나온다.

컨볼루션 된 신호의 세부적인 값은 위와 같다.(왼쪽부터)





 코드는 3-1-(1) 을 이용하여,x1과 x2를 컨볼루션 연산하였다.

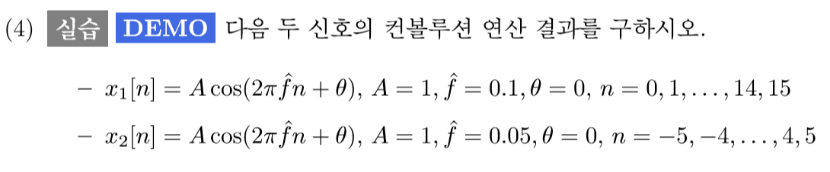
표현은 이산 시퀀스 데이터 플로팅 함수인 stem을 사용하였다.

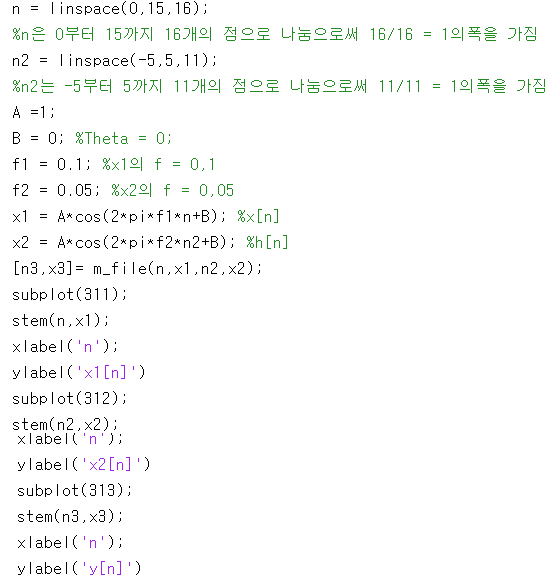
이 코드를 실행시키면 오른쪽의 그래프가 나온다.

컨볼루션 된 신호의 세부적인 값은 위와 같다.(왼쪽부터)



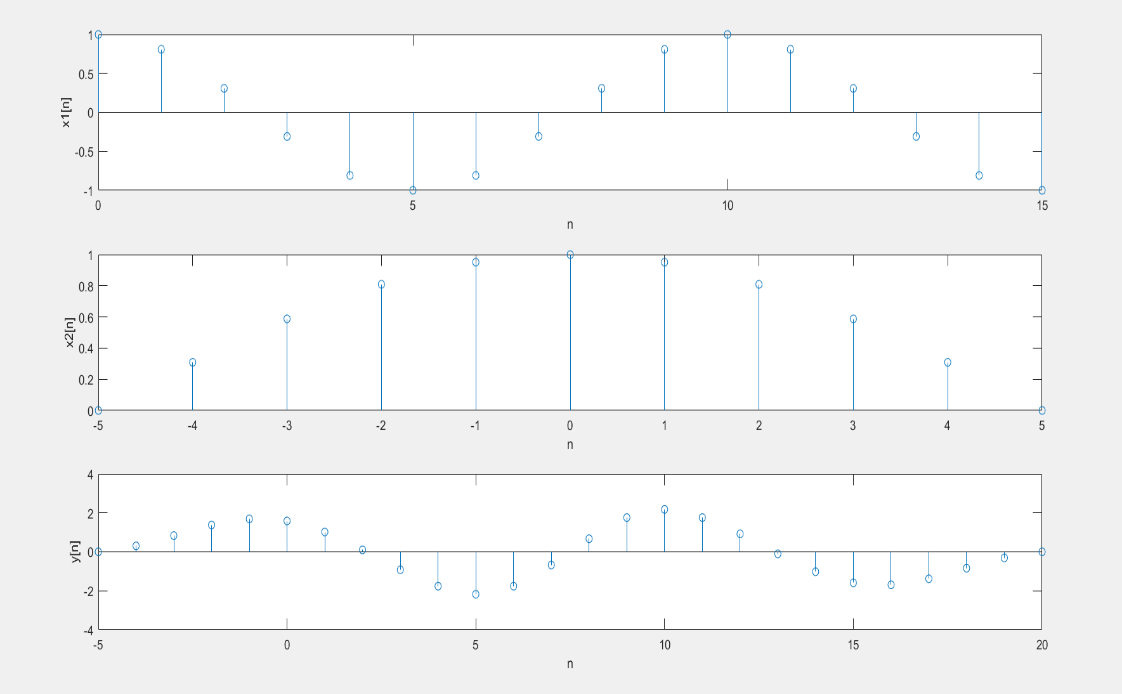






코드는 3-1-(1) 을 이용하여,x1과 x2를 컨볼루션 연산하였다.

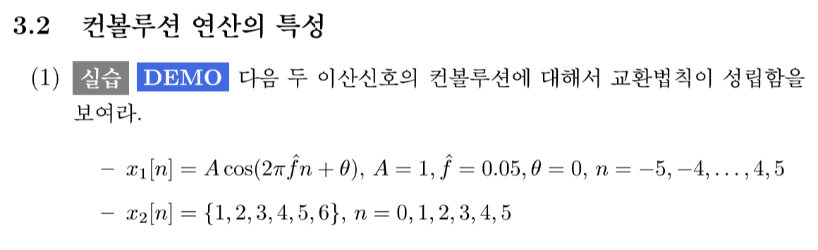
표현은 이산 시퀀스 데이터 플로팅 함수인 stem을 사용하였다.

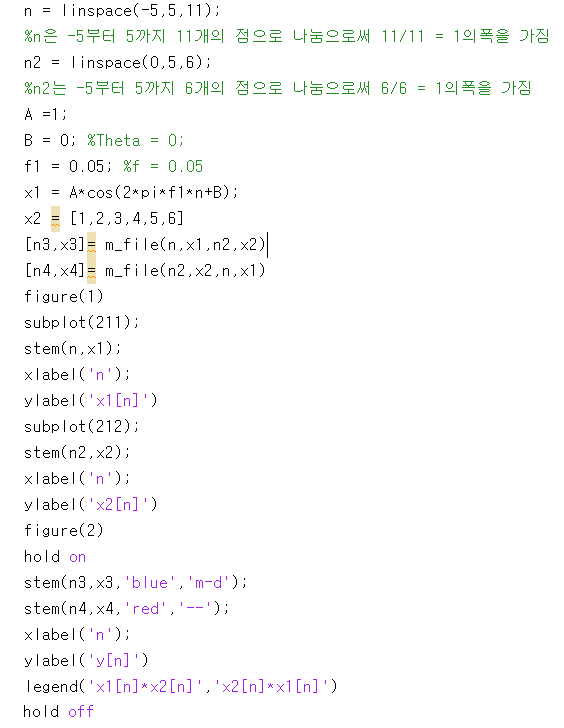






이 코드를 실행시키면 이러한 그래프가 나오고, 컨볼루션 된 신호의 세부적인 값은 위와 같다.(왼쪽부터)





x3는 x1을 x[n], x2를 h[n]으로 설정하여 컨볼루션 연산하였고,

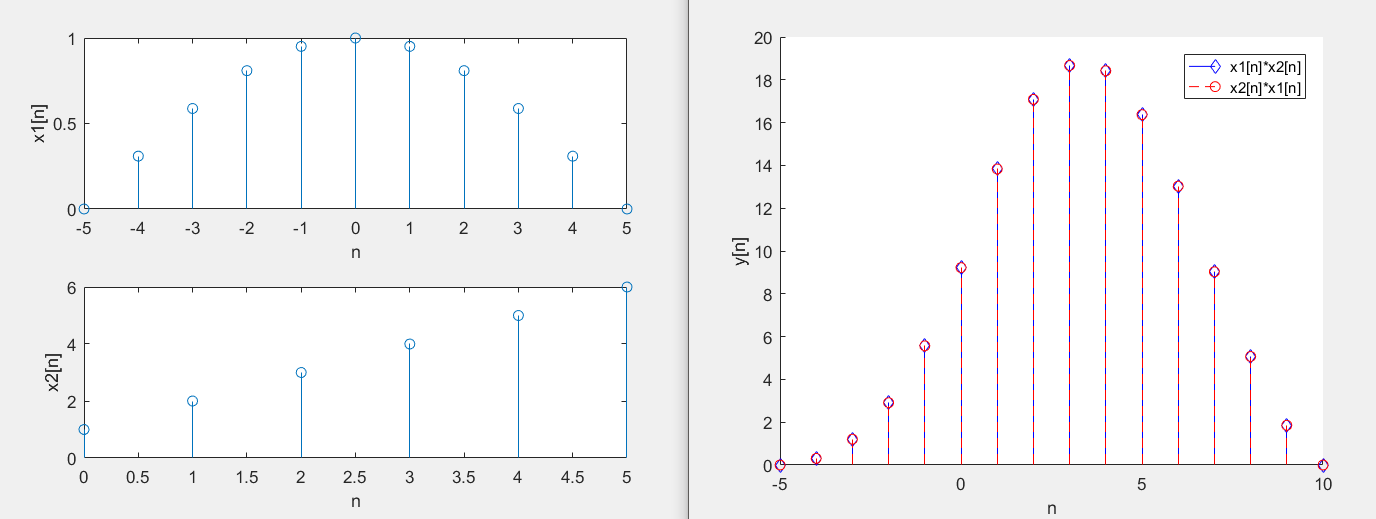
x4는 x2를 x[n], x1을 h[n]으로 설정하여 컨볼루션 연산하였다.

코드는 3-1-(1) 을 이용하여,1과 x2를 컨볼루션 연산하였고,

표현은 이산 시퀀스 데이터 플로팅 함수인 stem을 사용하였다.

또한 x3, x4를 비교하기위해 두 신호를 한 화면에 표시하기 위해 hold on을 사용하였다.

이 코드를 실행시키면 아래와 같은 그래프들이 그려진다.



X3는 파랑색 선에 다이아몬드 표시이고,

X4는 빨강색 선의 동그라미 표시인데,

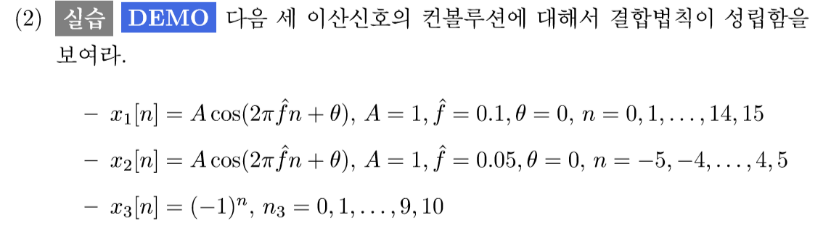
이 두 신호가 완전히 일치함을 알 수 있다.

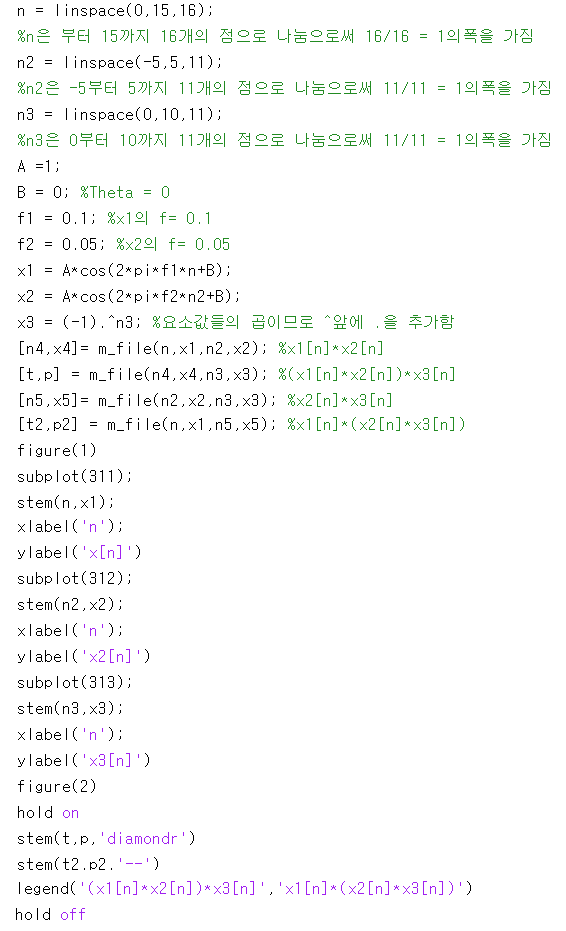
이로써 x1[n]\*x2[n] = x2[n]\*x1[n] 가 성립해, 교환법칙이 성립함을 알 수 있다.

이 컨볼루션된 두 신호의 세부적인 값은 다음과 같다. (왼쪽부터)









X4는 x1을 x[n], x2를 h[n]으로 설정하여 컨볼루션 연산하였고,

X5는 x2를 x[n], x3을 h[n]으로 설정하여 컨볼루션 연산하였고,

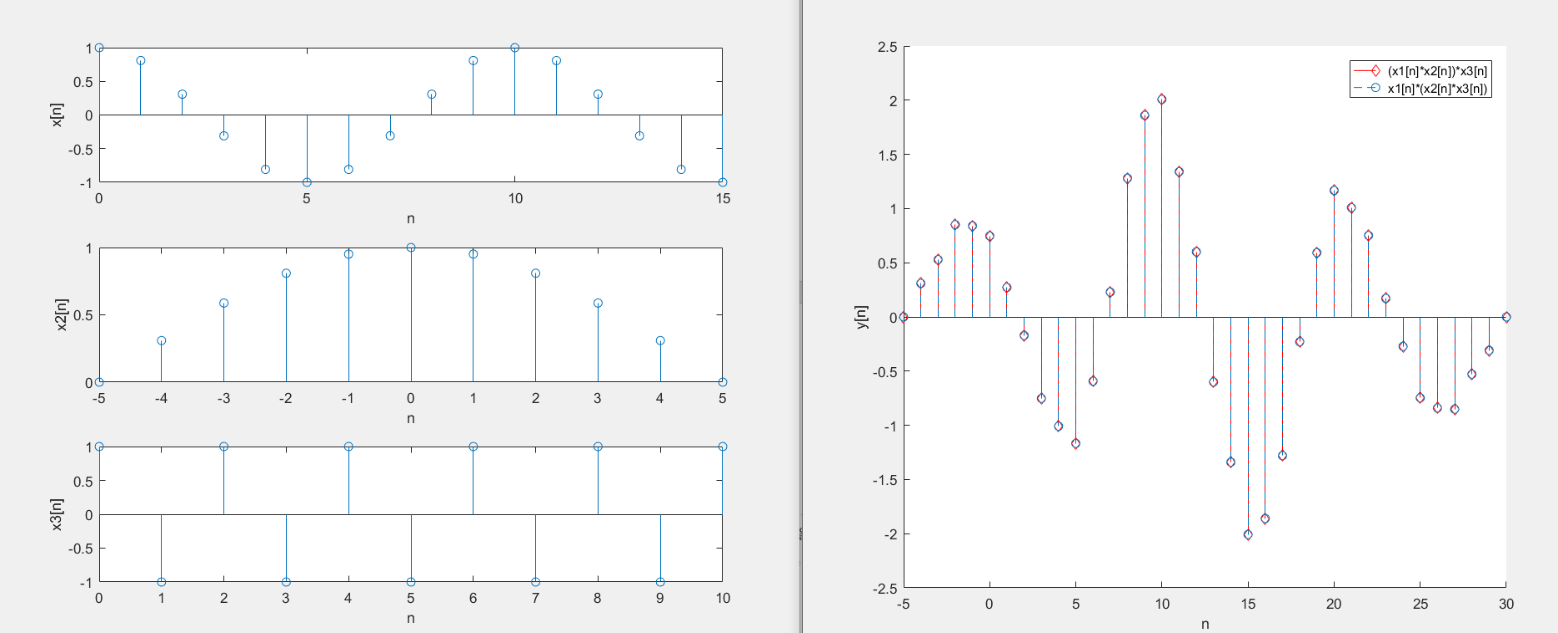
x3는 x4을 x[n], x3를 h[n]으로 설정하여 컨볼루션 연산하였고,

x4는 x1를 x[n], x5을 h[n]으로 설정하여 컨볼루션 연산하였다.

코드는 3-1-(1) 을 이용하여,1과 x2를 컨볼루션 연산하였고,

표현은 이산 시퀀스 데이터 플로팅 함수인 stem을 사용하였다.

또한 p, p2를 비교하기위해 두 신호를 한 화면에 표시하기 위해 hold on을 사용하였다.



p는 빨강색 선에 다이아몬드 표시이고,

p2는 파랑색 점선의 동그라미 표시인데,

이 두 신호가 완전히 일치함을 알 수 있다.

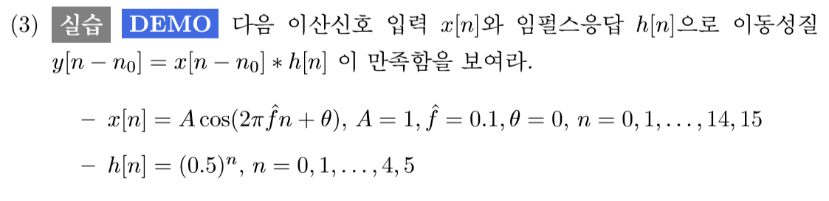
이로써 (x1[n]\*x2[n])\*x3[n] = x1[n]\*(x2[n]\*x3[n])가 성립해, 결합 법칙이 성립함을 알 수 있다.

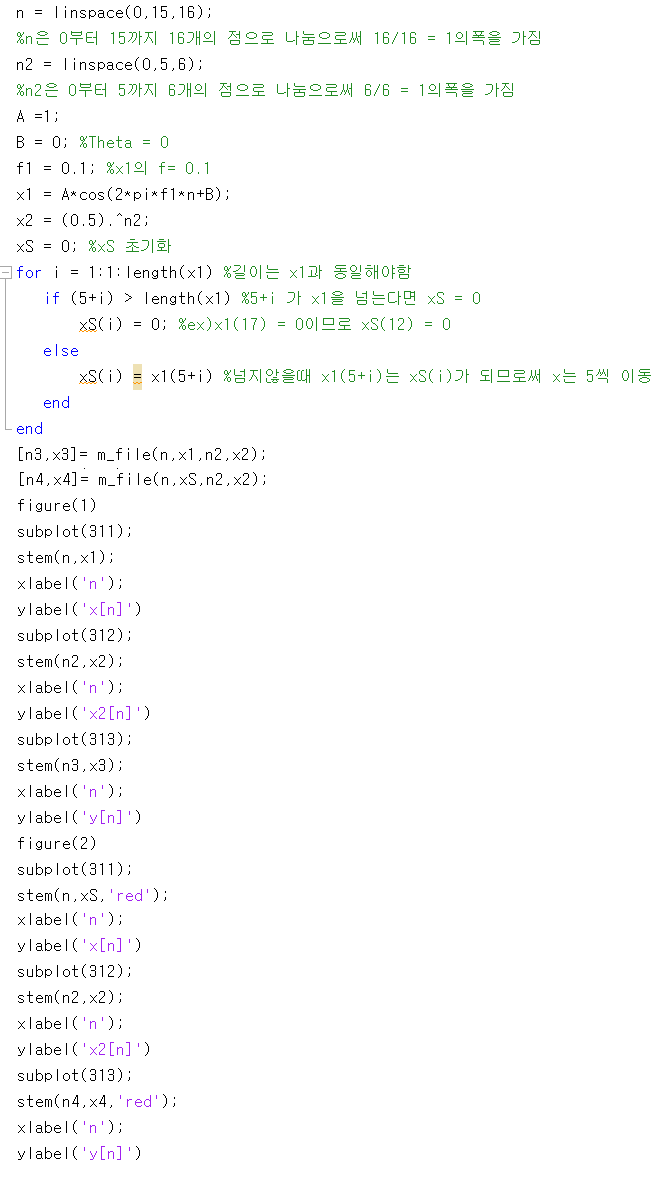
이 컨볼루션된 두 신호의 세부적인 값은 다음과 같다. (왼쪽부터)









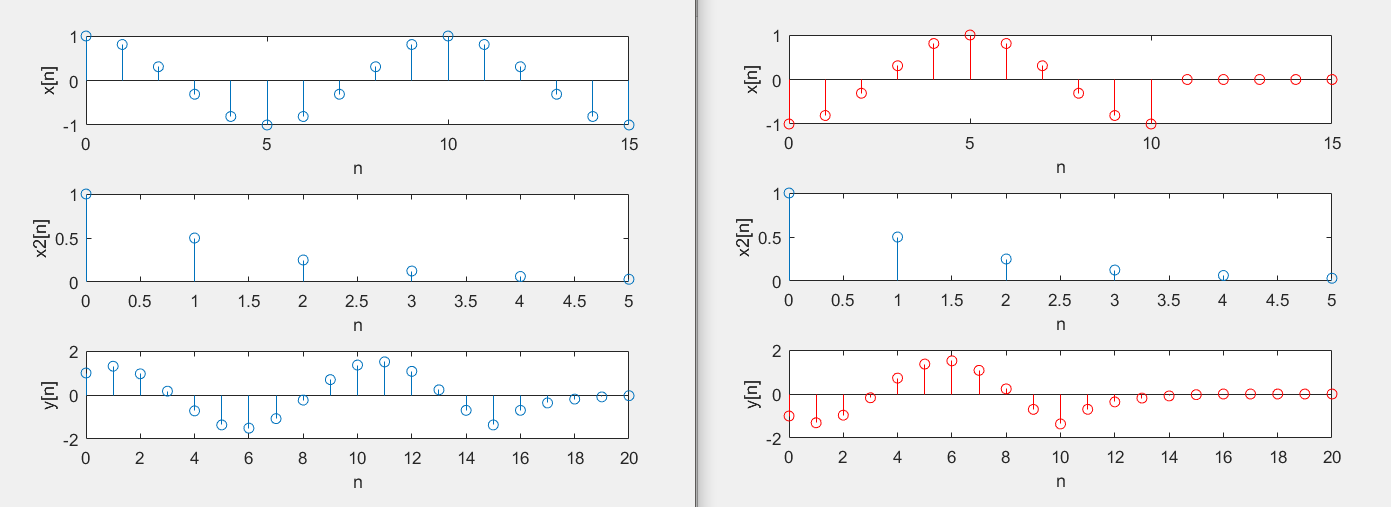


x3는 x1을 x[n], x2를 h[n]으로 설정하여 컨볼루션 연산하였고,

x4는 x1[n-5]를 x[n], x1을 h[n]으로 설정하여 컨볼루션 연산하였다.

코드는 3-1-(1) 을 이용하여,1과 x2를 컨볼루션 연산하였고,

표현은 이산 시퀀스 데이터 플로팅 함수인 stem을 사용하였다.



왼쪽이 x1[n]과 x2[n]을 컨볼루션 연산한 x3의 그래프고,

오른쪽이 x1[n-5]와 x2[n]을 컨볼루션 연산한 x3의 그래프다.

그래프를 보면 왼쪽의 y[5]와 오른쪽의 y[0]부터 왼쪽의 y[20]과 오른쪽의 y[15]까지 일치하는 것을 보아

이동성질 y[n−n0]= x[n−n0]∗h[n]을 만족함을 알 수 있다.

x3 신호의 세부적인 값은 다음과 같다. (왼쪽부터)





X4 신호의 세부적인 값은 다음과 같다. (왼쪽부터)



세부적으로 보면 초반에 약간의 오차가 있지만 이동성질을 만족함을 알 수 있다.