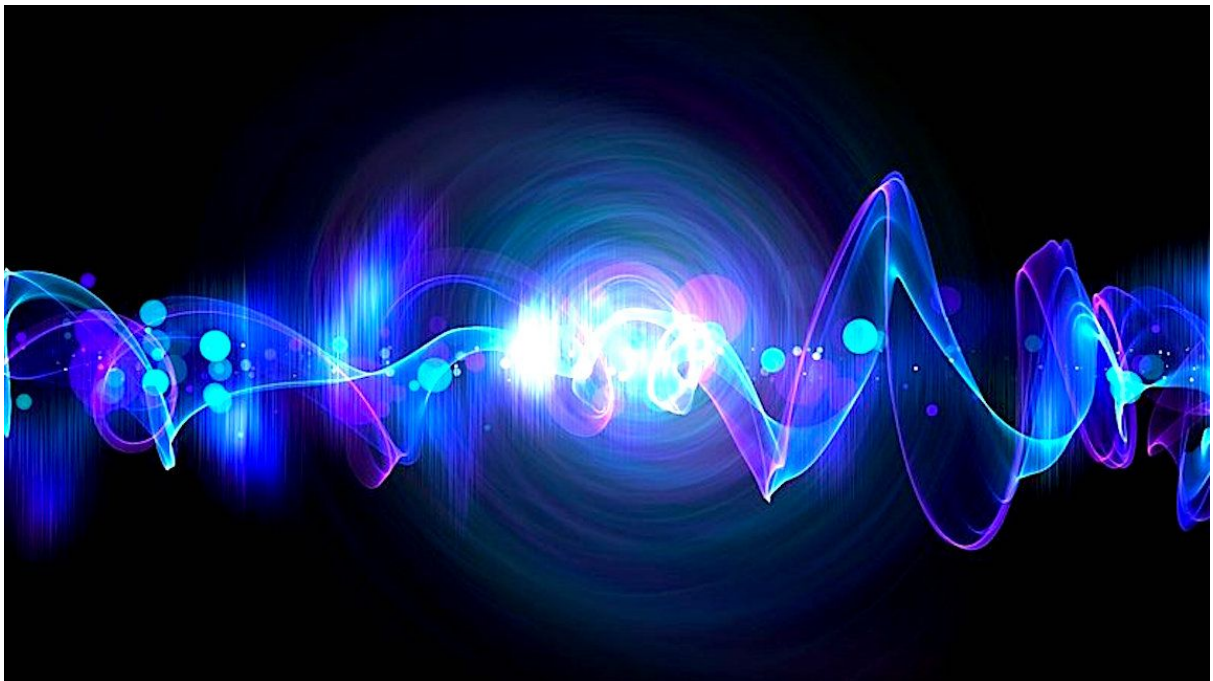


SIGNALS AND SYSTEMS

CONVOLUTION AND CORRELATION



DATE - 30TH August 2019

NAME - Shivam Malviya

ROLL NO. - 18EC01044

THEORY

● CONVOLUTION

Convolution is a mathematical way of combining two signals to form a third signal. It is the single most important technique in Digital Signal Processing.

Using the strategy of impulse decomposition, systems are described by a signal called the impulse response. Convolution is important because it relates the three signals of interest: the input signal, the output signal, and the impulse response.

Convolution is used to express the relation between input and output of an LTI system. It relates input, output and impulse response of an LTI system as

$$y(t)=x(t)*h(t)$$

Where $y(t)$ = output of LTI

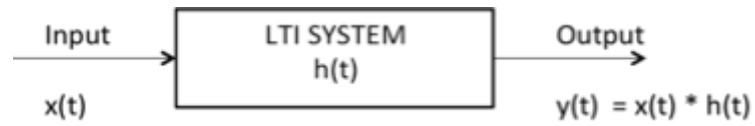
$x(t)$ = input of LTI

$h(t)$ = impulse response of LTI

There are two types of convolutions:

- *Continuous convolution*
- *Discrete convolution*

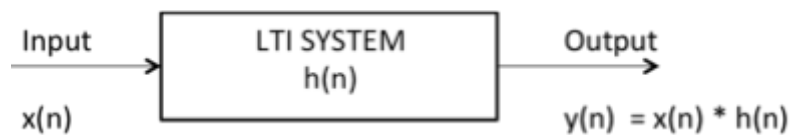
CONTINUOUS CONVOLUTION



$$y(t) = x(t) * h(t)$$

$$y(t) = \int_{-\infty}^{\infty} x(\tau) h(t - \tau) d\tau$$

DISCRETE CONVOLUTION



$$y(n) = x(n) * h(n)$$

$$y(n) = \sum_{k=-\infty}^{\infty} x(k) h(n - k) = \sum_{k=-\infty}^{\infty} x(n - k) h(k)$$

● **CROSS CORRELATION**

In general, correlation describes the mutual relationship which exists between two or more things. The same definition holds good even in the case of signals. That is, correlation between signals indicates the measure up to which the given signal resembles another signal.

In other words, if we want to know how much similarity exists between the signals 1 and 2, then we need to find out the correlation of Signal 1 with respect to Signal 2 or vice versa.

Depending on whether the signals considered for correlation are the same or different, we have two kinds of correlation:

- *Autocorrelation .*
- *Cross-correlation.*

Autocorrelation

This is a type of correlation in which the given signal is correlated with itself, usually the time-shifted version of itself. Mathematical expression for the autocorrelation of continuous time signal $x(t)$ is given by

$$R_{xx}(\tau) = \int_{-\infty}^{\infty} x(t) * x^*(t-\tau) dt$$

where $$ denotes the complex conjugate.*

Similarly the autocorrelation of the discrete time signal $x[n]$ is expressed as

$$R_{xx}[m] = \sum_{n=-\infty}^{\infty} x[n] * x^*[n-m]$$

Next, the autocorrelation of any given signal can also be computed by resorting to graphical technique. The procedure involves sliding the time-shifted version of the given signal upon itself while computing the samples at every interval. That is, if the given signal is digital, then we shift the given signal by one sample every time and overlap it with the original signal. While doing so, for every shift and overlap, we perform multiply and add.

Cross-Correlation

This is a kind of correlation, in which the signal in-hand is correlated with another signal so as to know how much resemblance exists between them. Mathematical expression for the cross-correlation of continuous time signals $x(t)$ and $y(t)$ is given by

$$R_{xy}(\tau) = \int_{-\infty}^{\infty} x(t) y^*(t-\tau) dt$$

Similarly, the cross-correlation of the discrete time signals $x[n]$ and $y[n]$ is expressed as

$$R_{xy}[m] = \sum_{n=-\infty}^{\infty} x[n] y^*[n-m]$$

Next, just as is the case with autocorrelation, cross-correlation of any two given signals can be found via graphical techniques. Here, one signal is slid upon the other while computing the samples at every interval. That is, in the case of digital signals, one signal is shifted by one sample to the right each time, at which point the sum of the product of the overlapping samples is computed.

DISCUSSION

APPLICATION OF CONVOLUTION

1. Image Processing

Image processing in spatial domain is a visually rich area of study dealing with pixel-manipulation techniques. Different operations are performed over the images, which are treated simply as two-dimensional arrays.

Normally, all these matrix-based operations are performed between a larger matrix (representing the complete image) and a smaller matrix (which is known as a 2D kernel). The kernel size and the associated values determine the impact exerted by it on the image considered.

2. Polynomial Multiplication

Consider a case where we want to multiply two polynomials, say, $(2x^2 + 3x - 1)$ and $(3x^3 - 2x)$. The work used to achieve this result is shown below

$$\begin{aligned}(2x^2 + 3x - 1) \times (3x^3 - 2x) &= 6x^5 - 4x^3 + 9x^4 - 6x^2 - \\ &3x^3 + 2x \\ &= 6x^5 + 9x^4 - 7x^3 - 6x^2 + 2x\end{aligned}$$

Equation 1.

Now, let's form two arrays whose elements are the coefficients of the polynomials mentioned and then convolve them. In this example, the arrays would be array 1 = $A_1 = \{2, 3, -1\}$ and array 2 = $A_2 = \{3, 0, -2, 0\}$.

We'll convolve them as shown in Table 1 by using a multiply-and-add technique, which you can read about in Part I of this article series.

From the table, their convolved result is $\{6, 9, -7, -6, 2, 0\}$. Expressed in its polynomial form, this is equivalent to what's shown in Equation 1.

Array A_1			3	0	-2	0			
Flipped Version of Array A_2	-1	3	2						6
		-1	3	2					$9 + 0 = 9$
			-1	3	2				$-3 + 0 - 4 = -7$
				-1	3	2			$0 + -6 + 0 = -6$
					-1	3	2		$2 + 0 = 2$
						-1	3	2	$0 = 0$



Convolved Result

Table 1.

The resultant table indicates that, when we are multiplying two polynomials, we are actually convolving their coefficients from a mathematical perspective. This neatly demonstrates that convolution aids us in performing the multiplication of polynomials.

3. Audio Processing

Auditoriums, cinema halls, and other similar constructions rely heavily on the concept of reverberation because it enhances the quality of sound greatly.

The process in which reverberation is digitally simulated is technically termed "convolution reverb". With convolution reverb, you can convolve an area's known impulse response with that of a desired sound in order to simulate the reverberation effect of a particular area.

Here's something cool we can do with such a simulation: We can figure out the effect of an auditorium's acoustics on a violinist's performance without being present there physically. Yet another way of using this technique is to merge two sounds—say, the voice of a singer with that of a veena, to produce a new sound.

4. Artificial Intelligence

Neural networks is an area of artificial intelligence which designs circuits by imitating connections in a human brain. The interconnection between neurons within the brain is modeled as the interconnection between the nodes of multiple layers constituting a network.

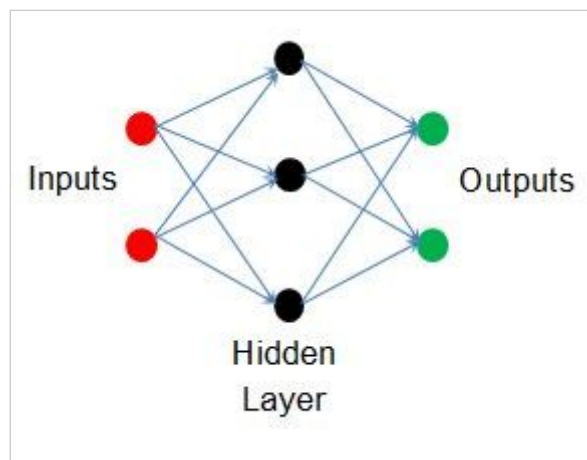


Figure 4. A simple artificial neural network

Figure 4 shows such an artificial neural network (ANN) in its simplest form. The example has a single hidden layer which routes the input nodes to the output nodes. The nodes are shown as circular regions while the blue lines indicate the interconnections.

In such a network, each interconnection is associated with a parameter called a weight. These weights indicate how much a particular node influences a

particular output. Some neural networks also incorporate convolution—they are known as convolutional neural networks and they are powerful tools for image processing.

APPLICATION OF CORRELATION

1. Signal processing related to human

hearing: *The human ear interprets signals that are nearly periodic signals to be exactly periodic. This is just like the case where an autocorrelated signal exhibits slightly different maxima-values at regular intervals of time.*

2. Vocal processing: *Correlation can help to determine the tempo or pitch associated with musical signals. The reason is the fact that the autocorrelation can effectively be used to identify repetitive patterns in any given signal.*

3. Determining synchronization pulses: *The synchronization pulses in a received signal, which in turn facilitates the process of data retrieval at the receiver's end. This is because the correlation of the known synchronization pulses with the*

incoming signal exhibits peaks when the sync pulses are received in it. This point can then be used by the receiver as a point of reference, which makes the system understand that the part of the signal following from then on (until another peak is obtained in the correlated signal indicating the presence of sync pulse) contains data.

4. Radar engineering: *Correlation can help determine the presence of a target and its range from the radar unit. When a target is present, the signal sent by the radar is scattered by it and bounced back to the transmitter antenna after being highly attenuated and corrupted by noise. If there is no target, then the signal received will be just noise. Now, if we correlate the arriving signal with the signal sent, and if we obtain a peak at a certain point, then we can conclude that a target is present. Moreover, by knowing the time-delay (indicated by the time-instant at which the correlated signal exhibits a peak) between the sent and received signals, we can even determine the distance between the target and the radar.*

5. Interpreting digital communications through noise: *As demonstrated above,*

correlation can aid in digital communications by retrieving the bits when a received signal is corrupted heavily by noise. Here, the receiver correlates the received signal with two standard signals which indicate the level of '0' and '1', respectively. Now, if the signal highly correlates with the standard signal which indicates the level of '1' more than with the one which represents '0', then it means that the received bit is '1' (or vice versa).

6. Impulse response identification: *As demonstrated above, cross-correlation of a system's output with its input results in its impulse response, provided the input is zero mean unit variance white Gaussian noise.*

7. Image processing: *Correlation can help eliminate the effects of varying lighting which results in brightness variation of an image. Usually this is achieved by cross-correlating the image with a definite template wherein the considered image is searched for the matching portions when compared to a template (template matching). This is further found to aid the processes like facial*

recognition, medical imaging, navigation of mobile robots, etc.

8. **Linear prediction algorithms:** *In prediction algorithms, correlation can help guess the next sample arriving in order to facilitate the compression of signals.*
9. **Machine learning:** *Correlation is used in branches of machine learning, such as in pattern recognition based on correlation clustering algorithms. Here, data points are grouped into clusters based on their similarity, which can be obtained by their correlation.*
10. **SONAR:** *Correlation can be used in applications such as water traffic monitoring. This is based on the fact that the correlation of the signals received by various shells will have different time-delays and thus their distance from the point of reference can be found more easily.*

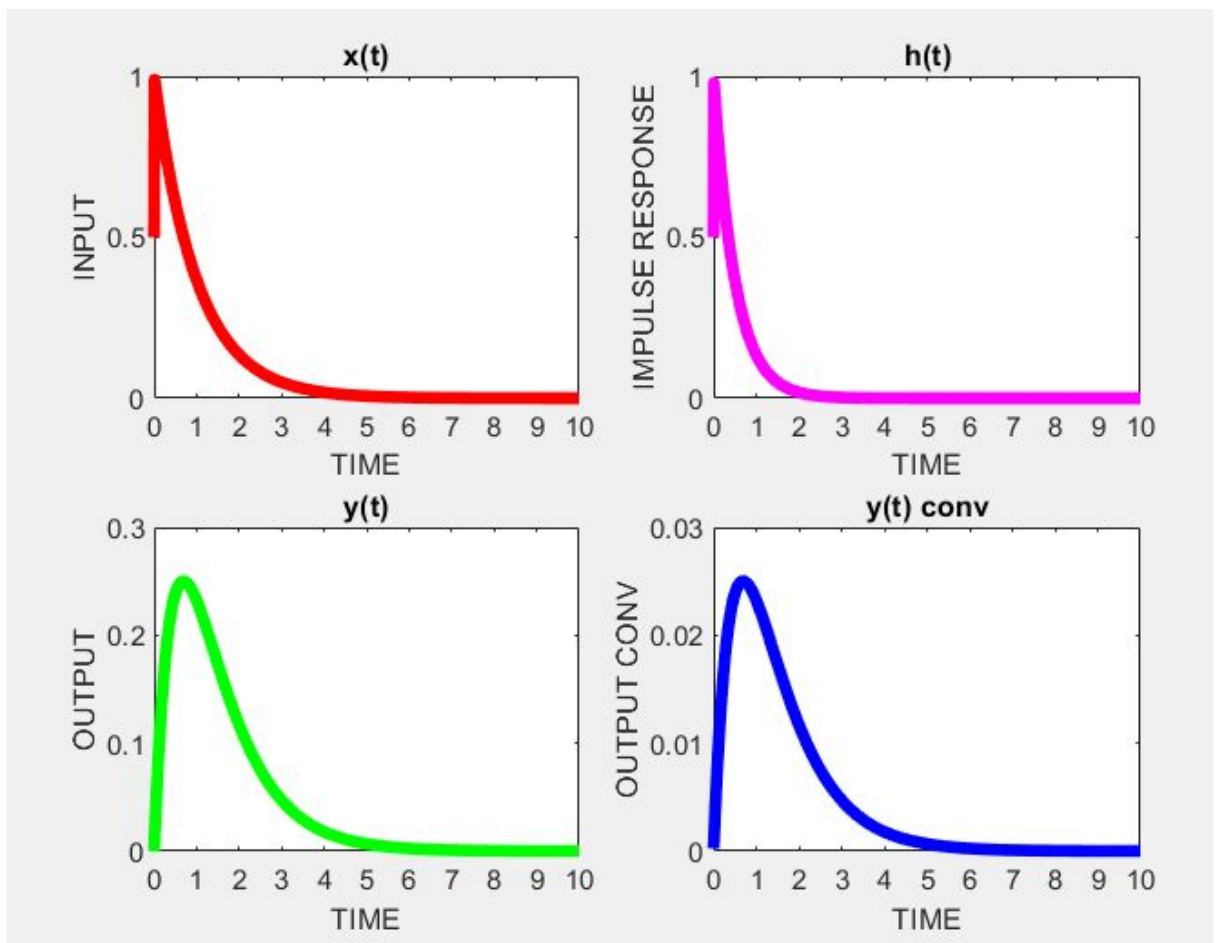
CONCLUSION

The discussion presented in this report reinforces the fact that the convolution and correlation is an inevitable part of many signal processing applications.

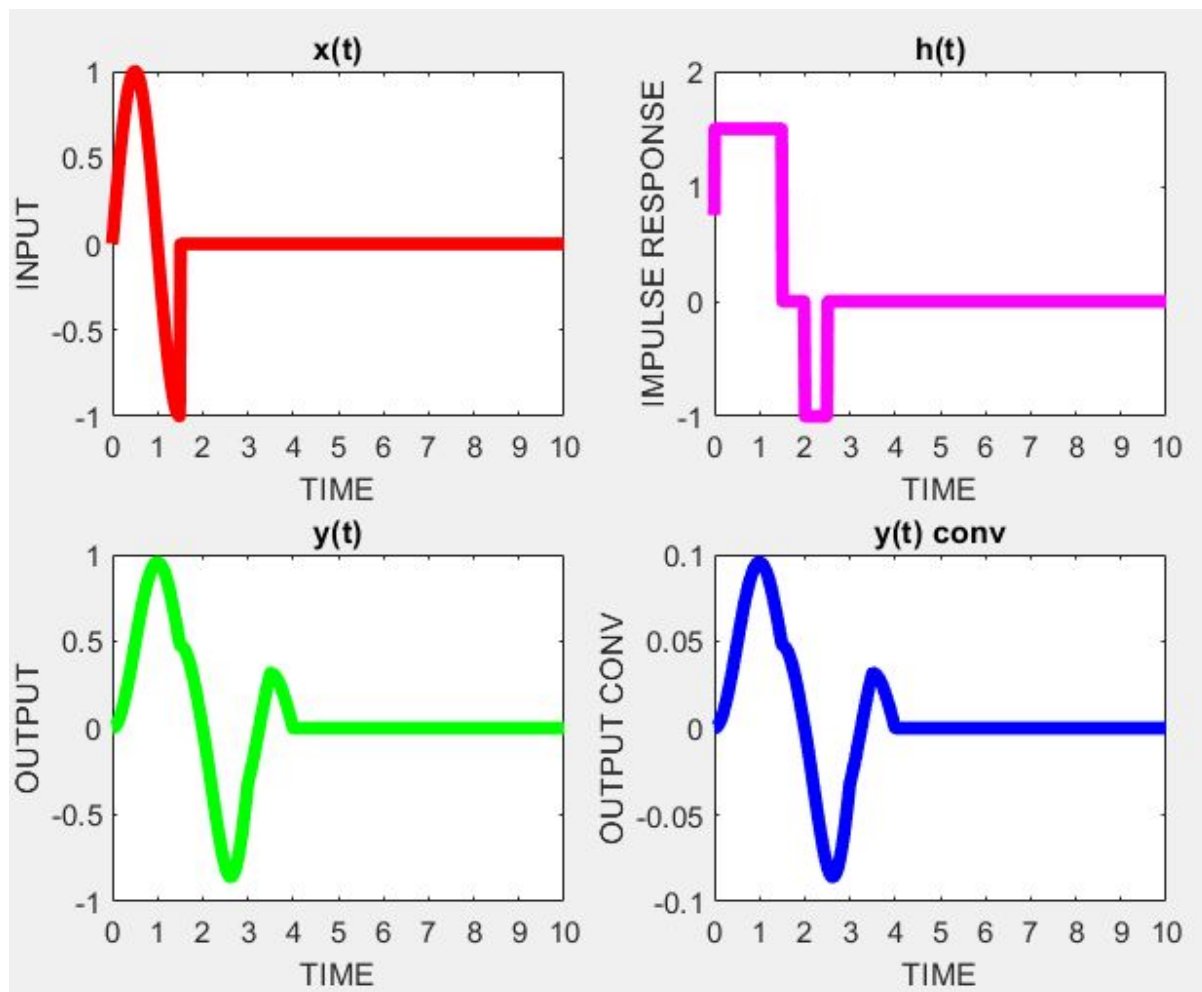
RESULTS

1. PART A

1.

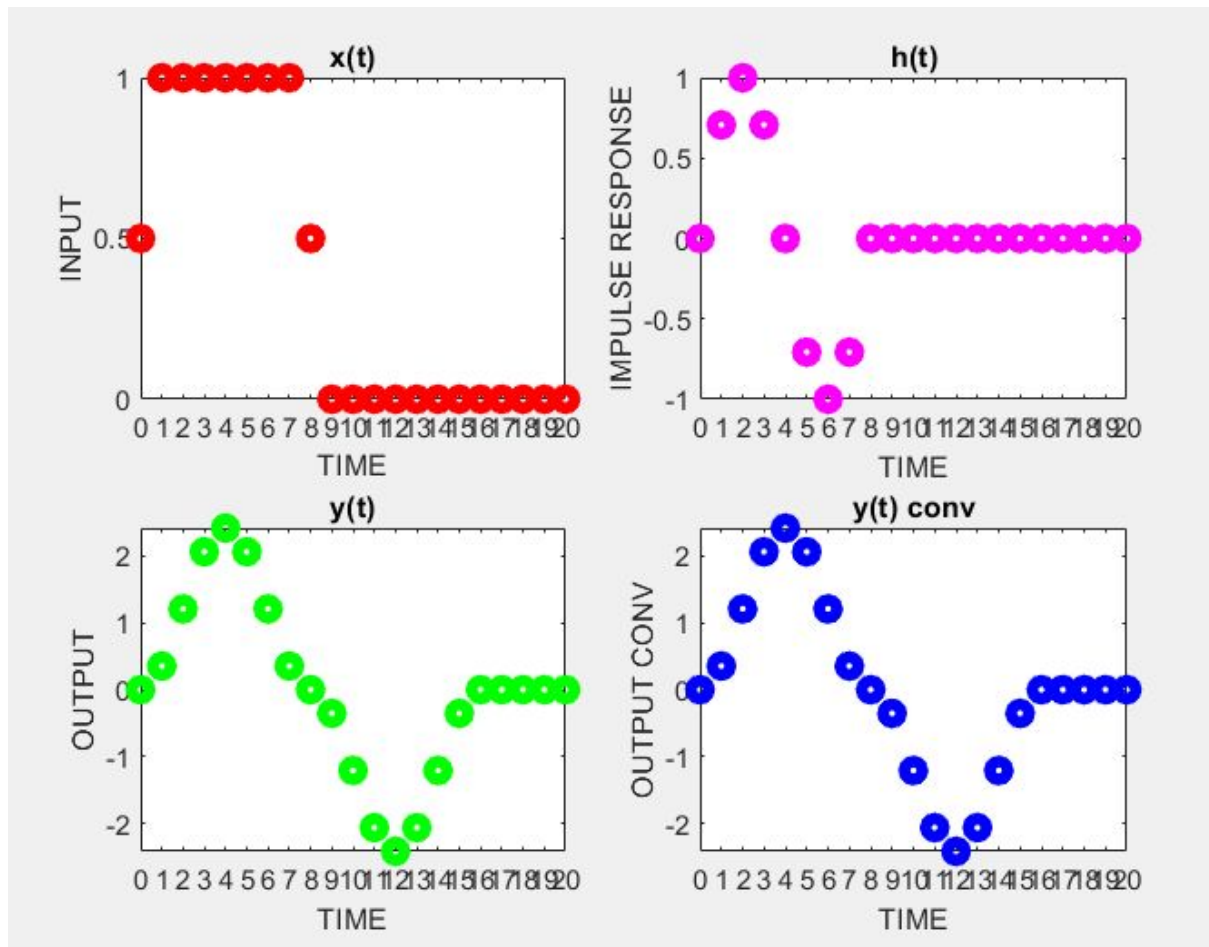


2.

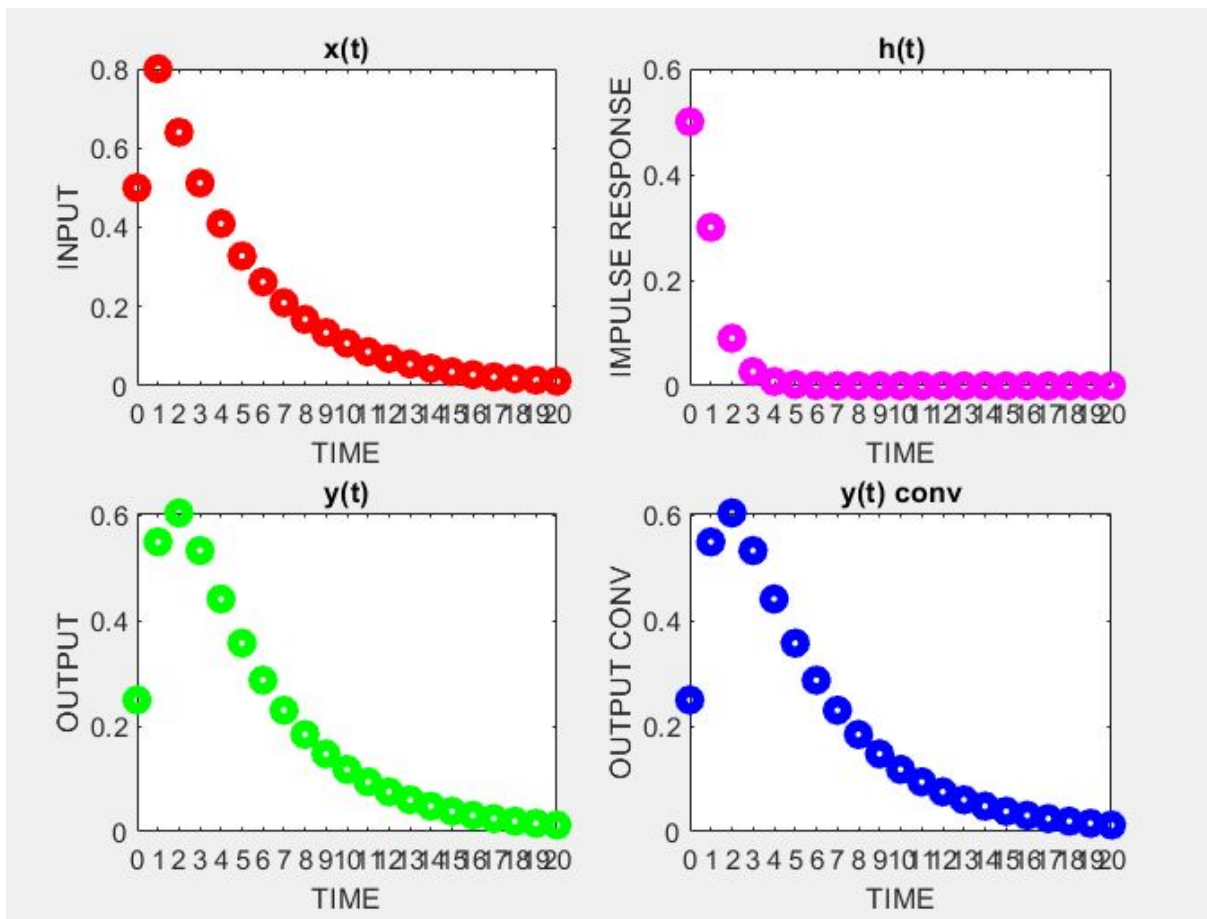


2. PART B

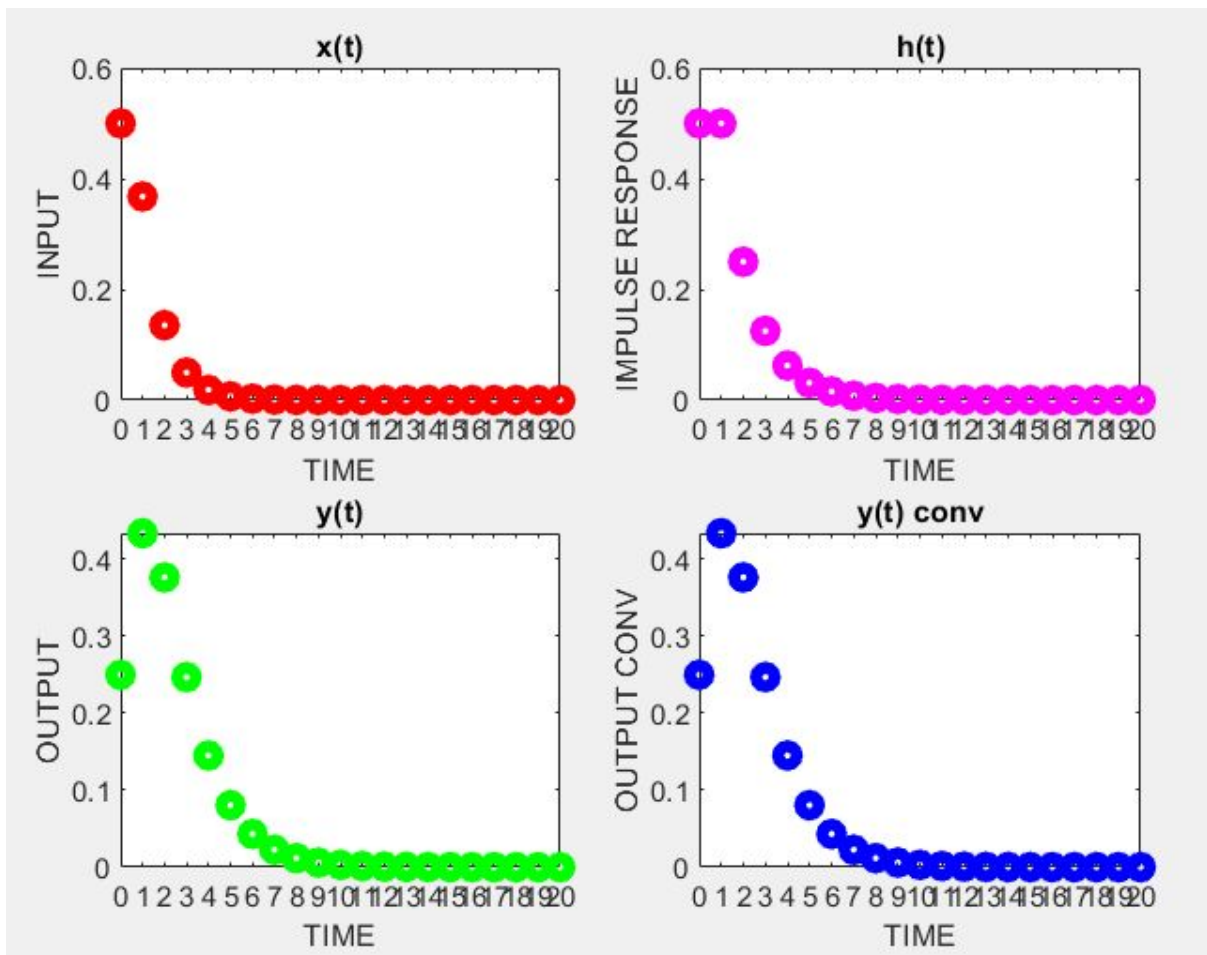
1.



2.

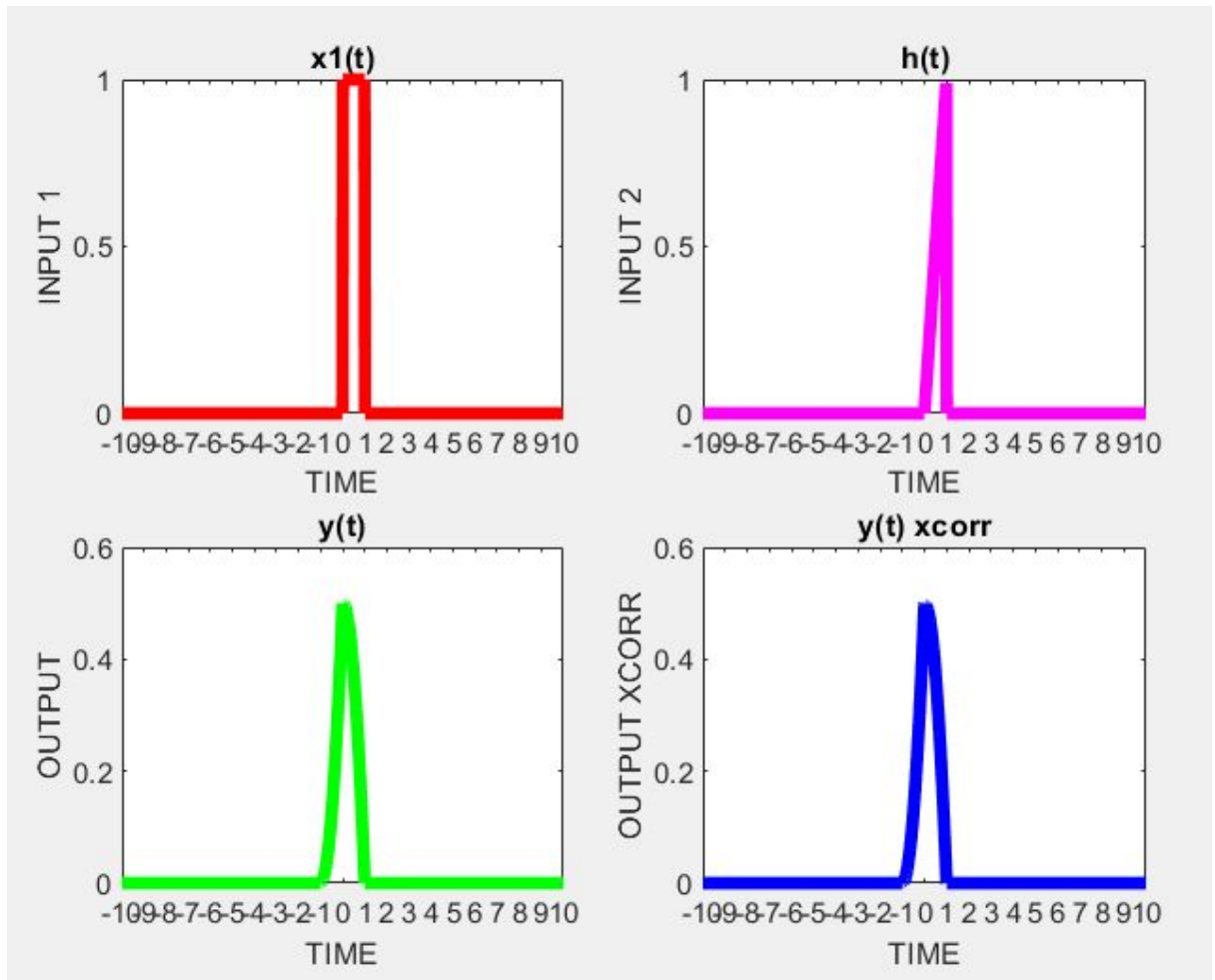


3.

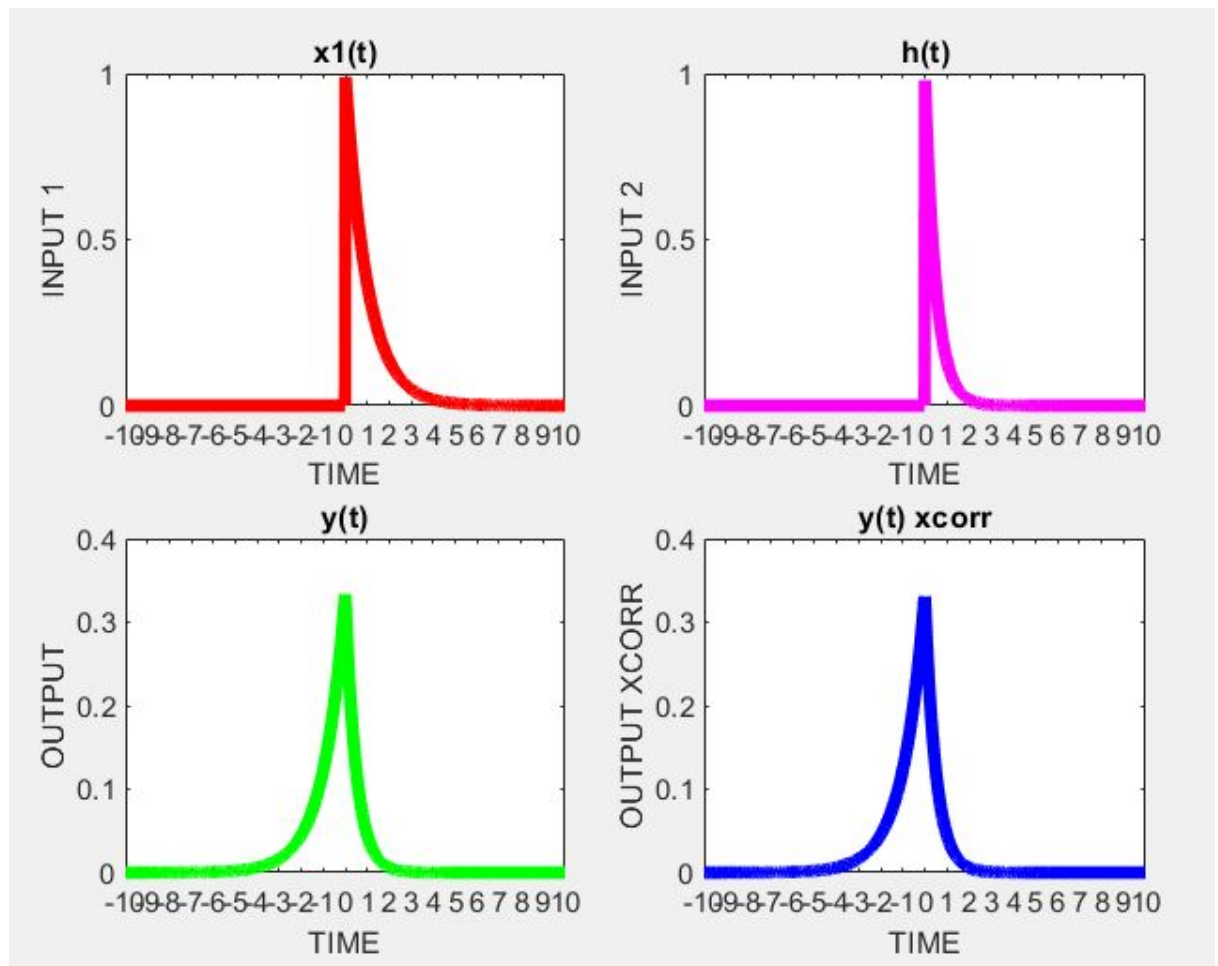


3.PART C

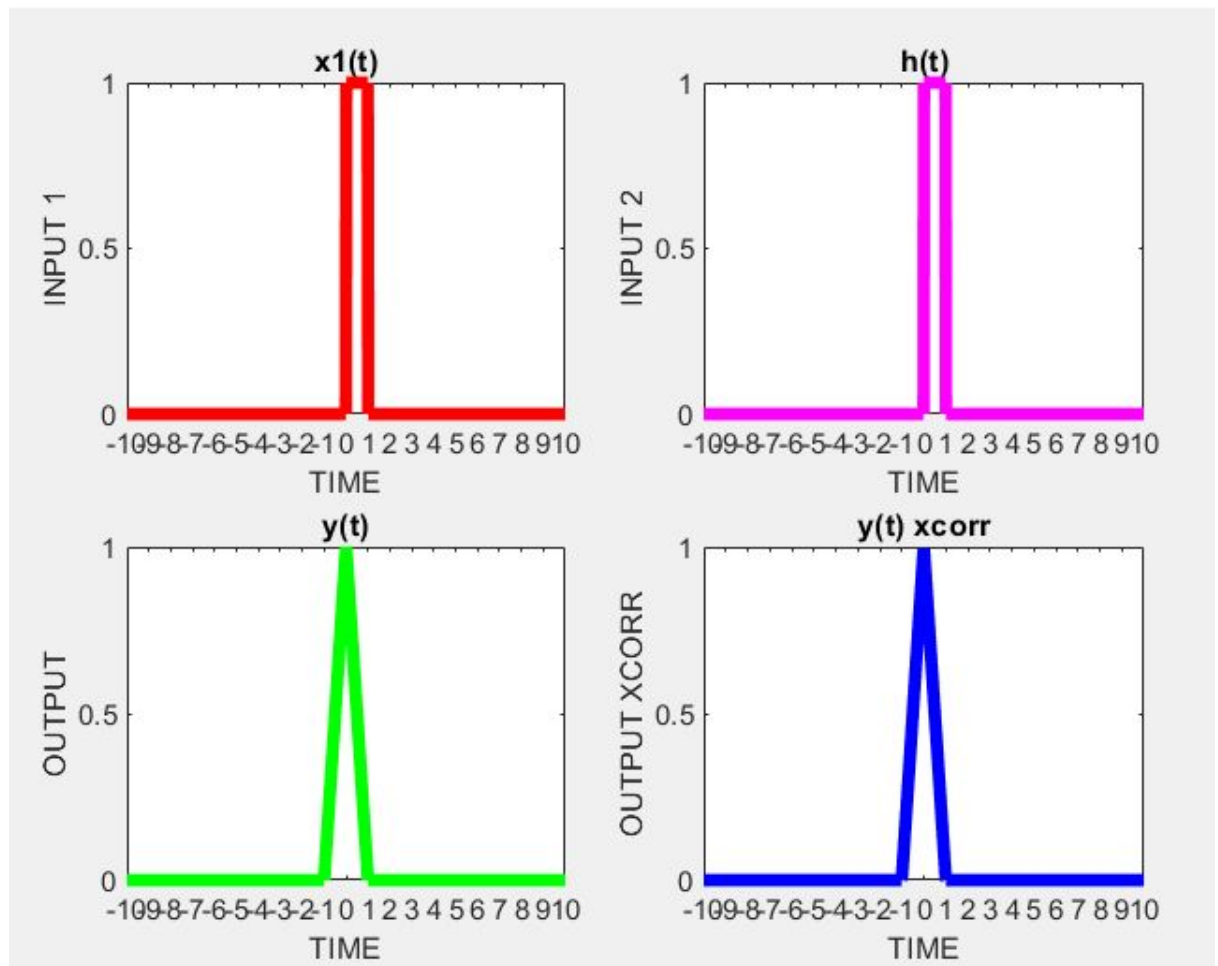
1.



2.

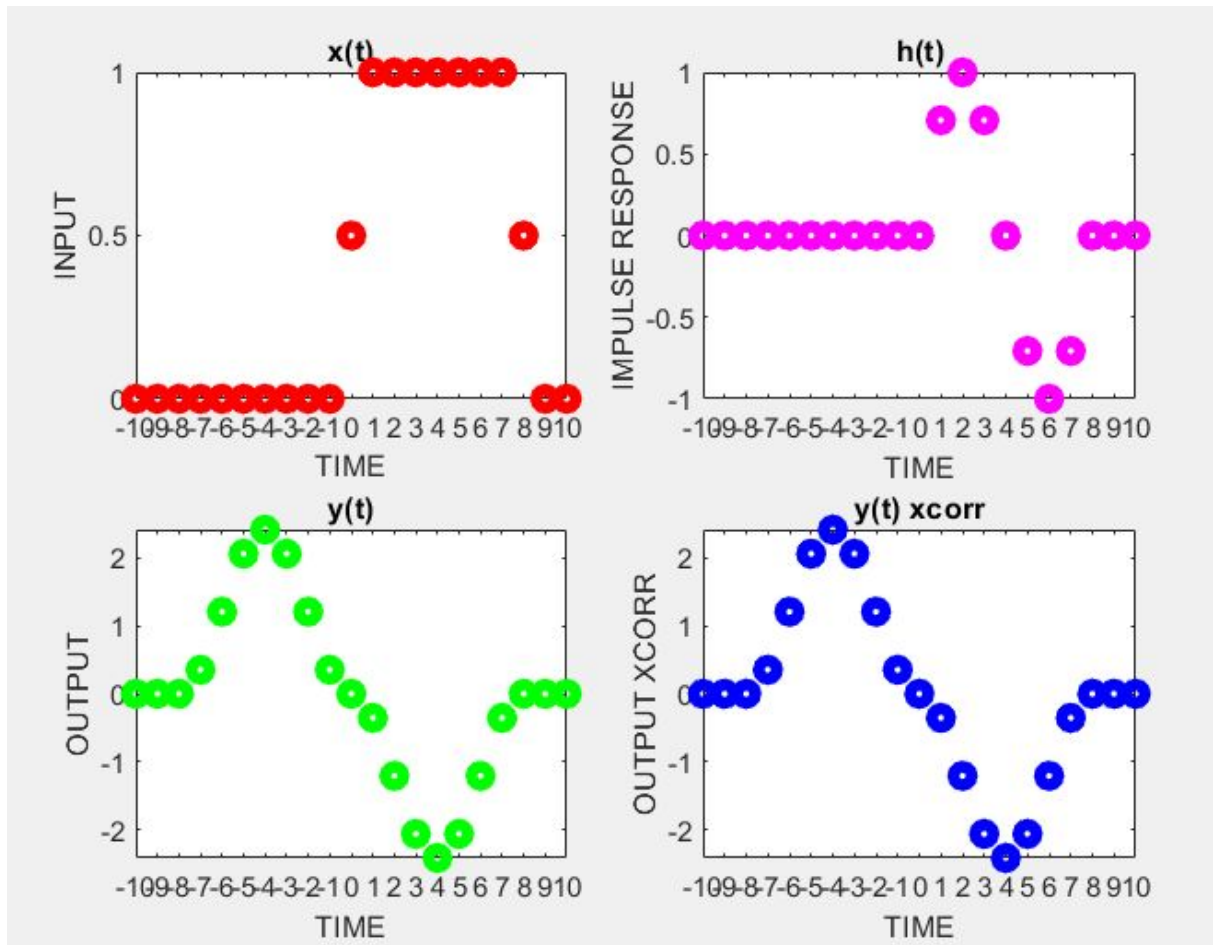


3.

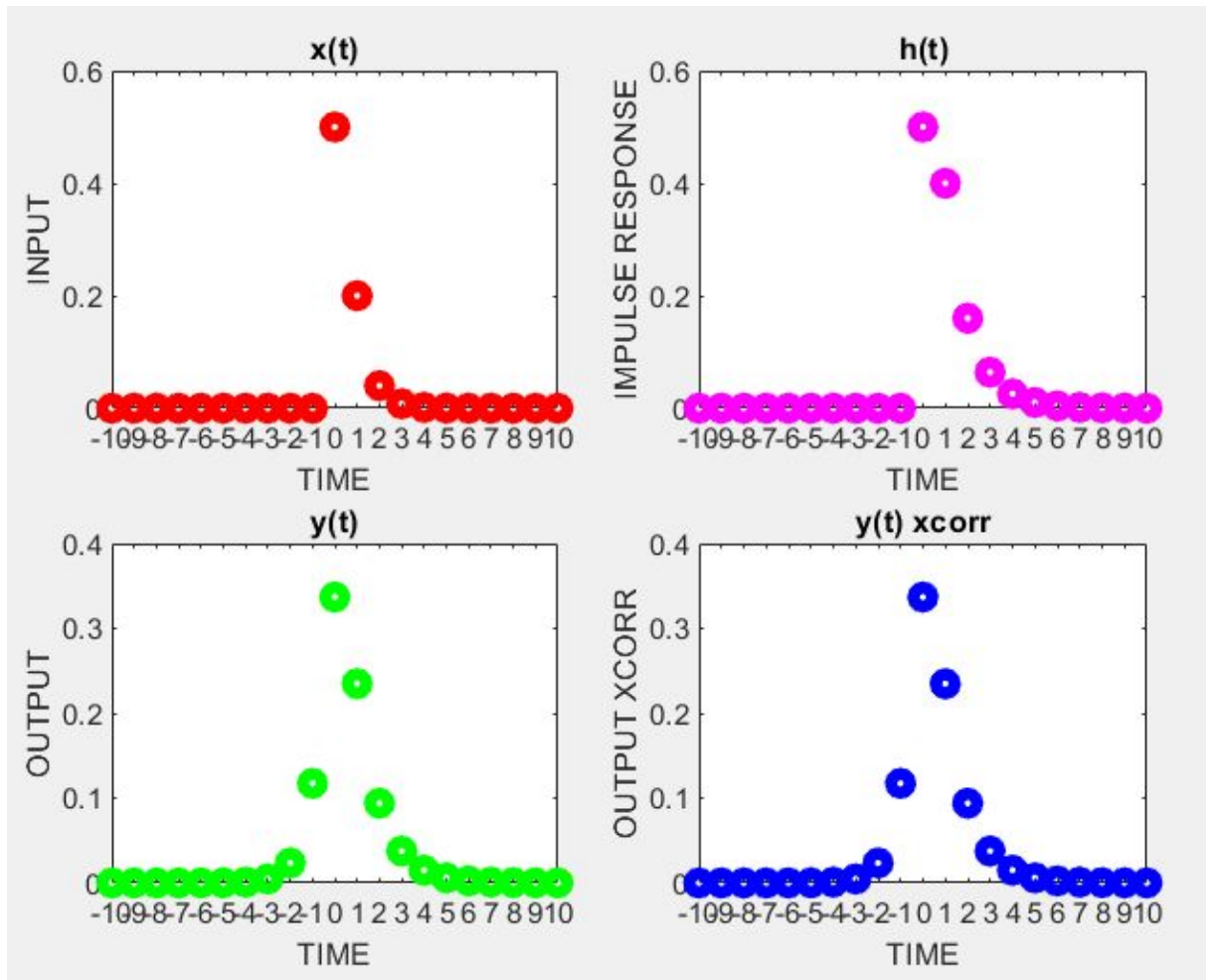


4. PART D

1.



2.



APPENDIX

1. CONVOLUTION A

function convolutionA(x, h)

syms T

syms y(t)

t_p = 0:0.01:10;

n = length(t_p);

x_p = double(x(t_p));

h_p = double(h(t_p));

y_conv = conv(x_p, h_p);

y_conv = y_conv(1, 1:n) / 1000;

subplot(2, 2, 1);

plot(t_p, x_p, '-r', 'LineWidth', 4);

title('x(t)');

xlabel('TIME');

ylabel('INPUT');

xlim([t_p(1) t_p(end)]);

xticks(t_p(1) : t_p(end));

subplot(2, 2, 2);

plot(t_p, h_p, '-m', 'LineWidth', 4);

title('h(t)');

xlabel('TIME');

ylabel('IMPULSE RESPONSE');

```
xlim([t_p(1) t_p(end)]);  
xticks(t_p(1) : t_p(end));
```

```
h(T) = subs(h(t), t, t-T);  
x(T) = subs(x(t), t, T);  
con = x*h;  
y(t) = int(con, T, -inf, inf);
```

```
subplot(2, 2, 3);  
plot(t_p, y(t_p), '-g', 'LineWidth', 4);  
title('y(t)');  
xlabel('TIME');  
ylabel('OUTPUT');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p(1) : t_p(end));
```

```
subplot(2, 2, 4);  
plot(t_p, y_conv, '-b', 'LineWidth', 4);  
title('y(t) conv');  
xlabel('TIME');  
ylabel('OUTPUT CONV');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p(1) : t_p(end));
```

2. A1

syms t

syms x(t)

syms h(t)

*x(t) = exp(-t) * heaviside(t);*

*h(t) = exp(-2*t) * heaviside(t);*

convolutionA(x, h);

3.A2

syms t

syms x(t)

syms h(t)

*x(t) = sin(pi * t) * (heaviside(t) - heaviside(t-1.5));*

h(t) = 1.5(heaviside(t) - heaviside(t-1.5)) -*

heaviside(t-2) + heaviside(t-2.5);

convolutionA(x, h);

4. CONVOLUTION B

function convolutionB(x, h)

% Calculation of y

syms y(n)

*% Manipulation of impulse response and input
signal.*

syms h_cal(n)

syms x_cal(n)

syms T

x_cal(T) = subs(x(n), n, T);

h_cal(n) = h(n);

h_cal(T) = subs(h(n), n, n-T);

t = 0:20;

x_val = x_cal(t);

h_val = h_cal(t);

% Convolving y

*y(n) = sum(x_val.*h_val);*

% Ploting Preparation

t_p = 0:20;

x_p = double(x(t_p));

h_p = double(h(t_p));

y_p = double(y(t_p));

y_conv = conv(x_p, h_p);

y_conv = y_conv(1:21);

% PLOTTING

subplot(2, 2, 1);

plot(t_p, x_p, 'Or', 'LineWidth', 4);

title('x(t)');

xlabel('TIME');

ylabel('INPUT');

xlim([t_p(1) t_p(end)]);

xticks(t_p);

subplot(2, 2, 2);

plot(t_p, h_p, 'Om', 'LineWidth', 4);

title('h(t)');

xlabel('TIME');

ylabel('IMPULSE RESPONSE');

xlim([t_p(1) t_p(end)]);

xticks(t_p);

```

subplot(2, 2, 3);
plot(t_p, y_p, 'Og', 'LineWidth', 4);
title('y(t)');
xlabel('TIME');
ylabel('OUTPUT');
xlim([t_p(1) t_p(end)]);
xticks(t_p);

```

```

subplot(2, 2, 4);
plot(t_p, y_conv, 'Ob', 'LineWidth', 4);
title('y(t) conv');
xlabel('TIME');
ylabel('OUTPUT CONV');
xlim([t_p(1) t_p(end)]);
xticks(t_p);

```

5.B1

```
syms n
```

```
syms x(n)
```

```
syms h(n)
```

```
x(n) = heaviside(n) - heaviside(n-8);
```

```
h(n) = sin(2*pi*n/8) * (heaviside(n) -  
heaviside(n-8));
```

```
convolutionB(x, h)
```

6.B2

syms n

syms x(n)

syms h(n)

*x(n) = ((0.8)^(n))*heaviside(n);*

*h(n) = ((0.3)^(n))*heaviside(n);*

convolutionB(x, h)

7.B3

syms n

syms x(n)

syms h(n)

*x(n) = (exp(-n))*heaviside(n);*

*h(n) = (2^(-n))*heaviside(n);*

convolutionB(x, h)

8. CORRELATION C

function correlationC(x, h)

syms T

syms y(t)

t_p = -10:0.01:10;

n = length(t_p);

x_p = double(x(t_p));

h_p = double(h(t_p));

y_corr = xcorr(h_p, x_p);

i1 = (n+1)/2;

*i2 = (3*n-1)/2;*

y_corr = y_corr(i1:i2) / 100;

subplot(2, 2, 1);

plot(t_p, x_p, '-r', 'LineWidth', 4);

title('x1(t)');

xlabel('TIME');

ylabel('INPUT 1');

xlim([t_p(1) t_p(end)]);

xticks(t_p(1):t_p(end));

subplot(2, 2, 2);

plot(t_p, h_p, '-m', 'LineWidth', 4);

title('h(t)');

```
xlabel('TIME');  
ylabel('INPUT 2');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p(1):t_p(end));
```

```
h(T) = subs(h(t), t, t+T);  
x(T) = subs(x(t), t, T);  
y(t) = int(x*h, T, -200, 200);  
y_p = double(y(t_p));
```

```
subplot(2, 2, 3);  
plot(t_p, y_p, '-g', 'LineWidth', 4);  
title('y(t)');  
xlabel('TIME');  
ylabel('OUTPUT');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p(1):t_p(end));
```

```
subplot(2, 2, 4);  
plot(t_p, y_corr, '-b', 'LineWidth', 4);  
title('y(t) xcorr');  
xlabel('TIME');  
ylabel('OUTPUT XCORR');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p(1) : t_p(end));
```

9.C1

```
syms t  
syms x(t)  
syms h(t)
```

```
x(t) = rectangularPulse(t-0.5);  
h(t) = t*rectangularPulse(t-0.5);  
correlationC(x, h);
```

10. C2

```
syms t  
syms x(t)  
syms h(t)
```

```
x(t) = exp(-t)*heaviside(t);  
h(t) = exp(-2*t)*heaviside(t);  
correlationC(x, h);
```

11. Cb

```
syms t  
syms x(t)  
syms h(t)
```

```
x(t) = rectangularPulse(t-0.5);  
h(t) = rectangularPulse(t-0.5);  
correlationC(x, h);
```

12. CORRELATION D

function correlationD(x, h)

% Calculation of y

syms y(n)

% Manipulation of impulse response and input signal.

syms h_cal(n)

syms x_cal(n)

syms T

x_cal(T) = subs(x(n), n, T);

h_cal(n) = h(n);

h_cal(T) = subs(h(n), n, n+T);

t = -10:10;

x_val = x_cal(t);

h_val = h_cal(t);

% Convolving y

```
y(n) = sum(x_val.*h_val);
```

```
% Ploting Preparation
```

```
t_p = -10:10;
```

```
n = length(t_p);
```

```
x_p = double(x(t_p));
```

```
h_p = double(h(t_p));
```

```
y_p = double(y(t_p));
```

```
y_corr = xcorr(h_p, x_p);
```

```
i1 = (n+1)/2;
```

```
i2 = (3*n-1)/2;
```

```
y_corr = y_corr(i1:i2);
```

```
% PLOTTING
```

```
subplot(2, 2, 1);
```

```
plot(t_p, x_p, 'Or', 'LineWidth', 4);
```

```
title('x(t)');
```

```
xlabel('TIME');
```

```
ylabel('INPUT');
```

```
xlim([t_p(1) t_p(end)]);
```

```
xticks(t_p);
```

```
subplot(2, 2, 2);
```

```
plot(t_p, h_p, 'Om', 'LineWidth', 4);
```

```
title('h(t)');
```

```
xlabel('TIME');  
ylabel('IMPULSE RESPONSE');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p);
```

```
subplot(2, 2, 3);  
plot(t_p, y_p, 'Og', 'LineWidth', 4);  
title('y(t)');  
xlabel('TIME');  
ylabel('OUTPUT');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p);
```

```
subplot(2, 2, 4);  
plot(t_p, y_corr, 'Ob', 'LineWidth', 4);  
title('y(t) xcorr');  
xlabel('TIME');  
ylabel('OUTPUT XCORR');  
xlim([t_p(1) t_p(end)]);  
xticks(t_p);
```

13. D1

```
syms n  
syms x(n)  
syms h(n)
```

```
x(n) = heaviside(n) - heaviside(n-8);  
h(n) = sin(2*pi*n/8) * (heaviside(n) - heaviside(n-8));  
correlationD(x, h)
```

14. D2

```
syms n  
syms x(n)  
syms h(n)
```

```
x(n) = ((0.2)^(n))*heaviside(n);  
h(n) = ((0.4)^(n))*heaviside(n);  
correlationD(x, h)
```

15. Db

```
syms n  
syms x(n)  
syms h(n)
```

```
x(n) = n*heaviside(n);  
h(n) = n*heaviside(n);  
correlationD(x, h)
```