Campus Ciudad de México

School of Engineering and Sciences
Department of Mechatronics

TE2019 Digital Signal Processing Laboratory

Student's 1 name and ID: Alfredo Zhu Chen - A01651980
Student's 2 name and ID: Luis Arturo Dan Fong - A016506720
Student's 3 name and ID: Miguel Alejandro Cruz Banda - A01656527

Lab # 6 LTI systems and the z-transform
Date: 16/04/2020

1 INTRODUCTION

Transformations are processes that are really helpful when treating different mathematical operations for several purposes. The z-transform origins date from 1947 when W. Hurewicz developed a method to solve a linear constant-coefficient difference equation by first transforming a signal. Regardless of W. Hurewicz being the first one to implement this new method to solve a constant-coefficient difference equation, the name "z-transform" was given until 1952 when mathematicians and scientists J.Ragazzini and L. Zadeh named the transformation at Columbia University [1].

The z-transform is a very powerful tool used to describe and analyze digital systems. It supports the techniques for digital filter design and frequency analysis of digital signals [2].It is also the equivalent of LaPlace transform for discrete time and its applications are many; from data comprehension, analysis to economics and control theory [3].

OBJECTIVES

1. To understand the usefulness of the z-transform for the analysis of discrete LTI systems.
2. To apply the z-transform in MATLAB.

2 MATERIALS & METHODS

The materials used in this laboratory were:

- 1 PC or laptop with at least 4GB RAM and and 500kB of space available
- MATLAB R2019a or newer versions

2.1 LTI systems: difference equations and the convolution sum

Given the following:

$$x(n) = 4cos(n\pi/8) \tag{1}$$
$$y(n) = y(n-1) + 2y(n-2) + x(n-2) \tag{2}$$
$$y(0) = 1 \tag{3}$$
$$y(1) = 1 \tag{4}$$

Signal (1) was directly defined in the interval of n from 0 to 10. An iterative method was used for indexing the *n* values for signal (2), while y(0) and y(1) were directly defined. Since MATLAB does not have index 0, the index *n* was shifted one place. Both functions (1) and (2) were plotted using MATLAB functions *subplot(m,n,p),* which plots the graph in p position of m by n grid of the figure, and *stem(X,Y),* which plots the discrete data sequence *Y* with the interval of *X,* for them to be in the same image.
Given the input signal (5) and impulse response (6):

$$x(n) = 0.8^{n}[u(n) - u(n-5)] \tag{5}$$

$$h(n) = 0.5^{n}[u(n) - u(n - 10)] \tag{6}$$

An interval *n2* was created from 0 to 10. Both of the signals were defined directly with this interval. The MATLAB functions *ones(m,n)* and *zeros(m,n*) are useful to create matrices of ones or zeros of size *m* by *n,* so they are used to create the unit step function according to (5) and (6). After this, the output signal y(n) is obtained by doing the convolution of the input signal (5) with the impulse response (6), this can be done with MATLAB function *conv(u,v)*, where u is (5) and v is (6) in this experiment. Then, the input signal, impulse response and output signal were plotted using *subplot(m,n,p)* and *stem(X,Y) as before.*

2.2 Convolution sum and the z-transform.

Given the function (7), it was defined with the help of MATLAB function *heaviside(x),* which evaluates the heaviside step function at *x*. So the heaviside function evaluated at *n* minus the heaviside evaluated at *n-3* could represent (7).

$$x(n) = u(n) - u(n - 3) \tag{7}$$

In order to find the z-transform of (7), the z-transform pair (8) and the time shifting property (9) are used. It can be applied z-transform directly in the first term of (7), however the second term needs the time shifting property before the z-transform, in this case the time is shifted by $n_0 = 3$.

$$u(n) \Leftrightarrow \frac{1}{1-z^{-1}} \tag{8}$$

$$x[n - n_0] \Leftrightarrow z^{-n_0}X(z) \tag{9}$$

After applying the z-transform to each term, thanks to the linearity property (10), the result is as (11).

$$a_1x_1[n] + a_2x_2[n] \Leftrightarrow a_1X_1(z) + a_2X_2(z) \tag{10}$$

$$X(z) = \frac{1}{1-z^{-1}} - \frac{z^{-3}}{1-z^{-1}} = \frac{1-z^{-3}}{1-z^{-1}} \tag{11}$$

Y(z) is obtained by multiplying *X(z)* by itself as it can be seen in (12). The main idea is to obtain the coefficients of Y(z), therefore some algebra manipulations were required to take expression into a power series. (13) is calculated by multiplying the numerator and denominator inside the parentheses of (12) by z^3, (14) shows the factorization of the numerator inside the brackets since it is a difference of cubes, cancellation of terms in the fraction leads to (15), the square is applied to the numerator and denominator in (16), (17) is obtained after grouping common terms and a final expression of the power series (18) is determined by dividing the numerator by the denominator.

$$Y(z) = X(z)X(z) = \left(\frac{1-z^{-3}}{1-z^{-1}}\right)^2 \tag{12}$$

$$Y(z) = \left[\frac{z^3-1}{z^2(z-1)}\right]^2 \tag{13}$$

$$Y(z) = \left[\frac{(z-1)(z^2+z+1)}{z^2(z-1)}\right]^2 \tag{14}$$

$$Y(z) = \left[\frac{(z^2+z+1)}{z^2}\right]^2 \tag{15}$$

$$Y(z) = \frac{(z^4+z^3+z^2+z^3+z^2+z+z^2+z+1)}{z^4} \tag{16}$$

$$Y(z = \frac{(z^4+2z^3+3z^2+2z+1)}{z^4}) \tag{17}$$

$$Y(z) = 1 + 2z^{-1} + 3z^{-2} + 2z^{-3} + z^{-4} \tag{18}$$

The convolution of (7) with itself is done by using the table method as it can be seen in the following table 1. In this case, the impulse response in the convolution sum is the same as (7), it is time-reversed to obtain *x(-k)* and then shifted by *n* to form *h(n-k)*. The two sequences *x(k)* and *x(n-k)* are multiplied together for all values of *k* with *n* fixed at some value. Then, the product is summed over all *k* to produce a single example *y(n)*, which the results will be shown later in section 3.2.

Table 1Table method to obtain the convolution

k	-2	-1	0	1	2	3	4
x[k]	0	0	1	1	1	0	0
x(-k)	1	1	1	0	0	0	0
x(1-k)	0	1	1	1	0	0	0
x(2-k)	0	0	1	1	1	0	0
x(3-k)	0	0	0	1	1	1	0
x(4-k)	0	0	0	0	1	1	1

Finally, the result has to be verified by making the convolution of (7) with itself using the MATLAB function *conv(u,v)*. The coefficients must be the same as the ones obtained in *Y(z)*, which will be compared in section 3.2.

2.3 Inverse z-transform

By using symbolic MATLAB the inverse z-transform of the signal (19) has to be found.

$$X(z) = \frac{(2-z^{-1})}{\left[2(1+0.5z^{-1})(1+0.5z^{-1})\right]} \tag{19}$$

Syms was used to define both *n* and *z* as symbolic variables for then applying an inverse z-transform with the function *iztrans(f),* considering *f* to be function (19) and finally evaluating the limit for the MATLAB function using *limit(f,var,a),* being that the limit of *f* when *var* approaches to *a.* In our case, the result of the inverse z-transform x(n) required to be evaluated in an infinite value, so this function was helpful to do so.

2.4 MATLAB partial fraction expansion

This experiment requires to find analytically the partial expansion of (*20)* and then obtain *x(n)*. The final result should match the result by using MATLAB. *x(n)* can be obtained similarly as previous section 2.3 with MATLAB function *iztrans(f).*

$$X(z) = \frac{(2z^{-1})}{\left[(1-z^{-1})(1-2z^{-1})\right]} \tag{20}$$

To find the partial fraction expansion of (20), the denominator is separated as (21), where *A* and *B* are coefficients required to be determined. Then the denominator is put together again so this helps to find the values of the coefficients. This can be done directly since in (20) the order of the denominator is greater than the order of the numerator and that makes it a proper rational function.

$$X(z) = \frac{A}{1-z^{-1}} + \frac{B}{1-2z^{-1}} = \frac{A(1-2z^{-1})+B(1-z^{-1})}{(1-z^{-1})(1-2z^{-1})} \quad (21)$$

To find the values of the coefficients, the numerator of (20) should be equal to the numerator of (21), as it is shown in (22). According to (22), (23) and (24) are obtained by matching the coefficients of variable *z* depending on the order of the exponent.

$$(2z^{-1}) = A(1 - 2z^{-1}) + B(1 - z^{-1}) = A - 2Az^{-1} + B - Bz^{-1} \quad (22)$$

$$2 = - 2A - B \quad (23)$$

$$0 = A + B \quad (24)$$

By adding (23) with (24), *B* from both equations cancel each other out and A can be obtained as (25) and (26). B is then obtained by substitution of *A* in (24).

$$2 + 0 = (- 2A - B) + (A + B) = - A \quad (25)$$

$$A = - 2 \quad (26)$$

$$B = 2 \quad (27)$$

Finally, having (21) as a linear combination of two terms and applying linearity property (10), the inverse z-transform of *X(z)* can be obtained by adding the inverse z-transform of each term with the z-transform pair (8) and (29) .

$$a^{n}u(n) \Leftrightarrow \frac{1}{1-az^{-1}} \quad (29)$$

3 RESULTS AND DISCUSSION

3.1 LTI systems: difference equations and convolution sum

The input signal *x(n)* and the output signal *y(n) evaluated from 0 to 10* are shown in figure 3.1.1. The first one looks like a cosine wave as it is supposed to be. The second signal seems to be increasing very quickly as *n* increases, the reason for this behaviour is that it accumulates previous values of the output signal as *n* increases. It also depends on the first signal, but the dependency is not very evident since it is rapidly increasing after *n=5* and the negative values of *x(n)* do not affect that much after that point to y(n).

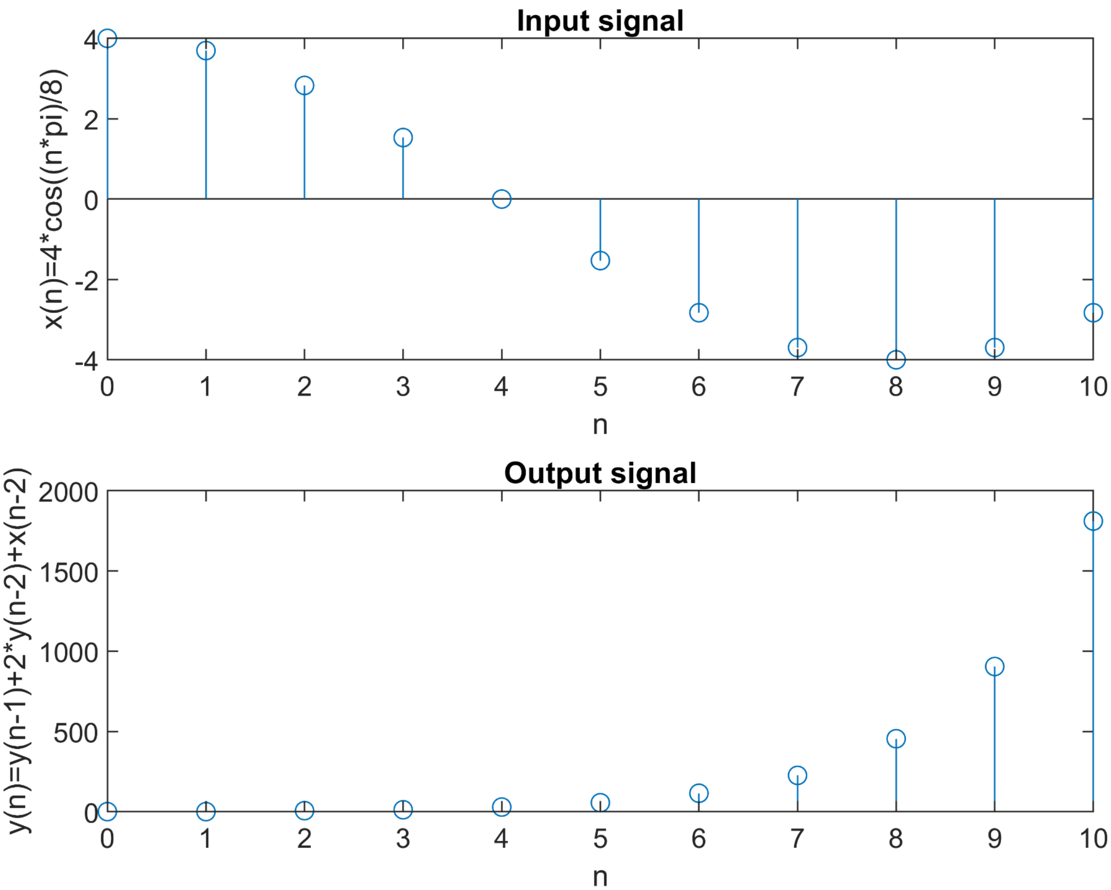

Figure 3.1.1 Plot of input signal *x(n)* and output signal *y(n)*

Figure 3.1.2 shows the graphs of the input signal *x(n)*, impulse response *h(n)* and the convolution of *x(n)* with *h(n)* as output signal *y(n).*

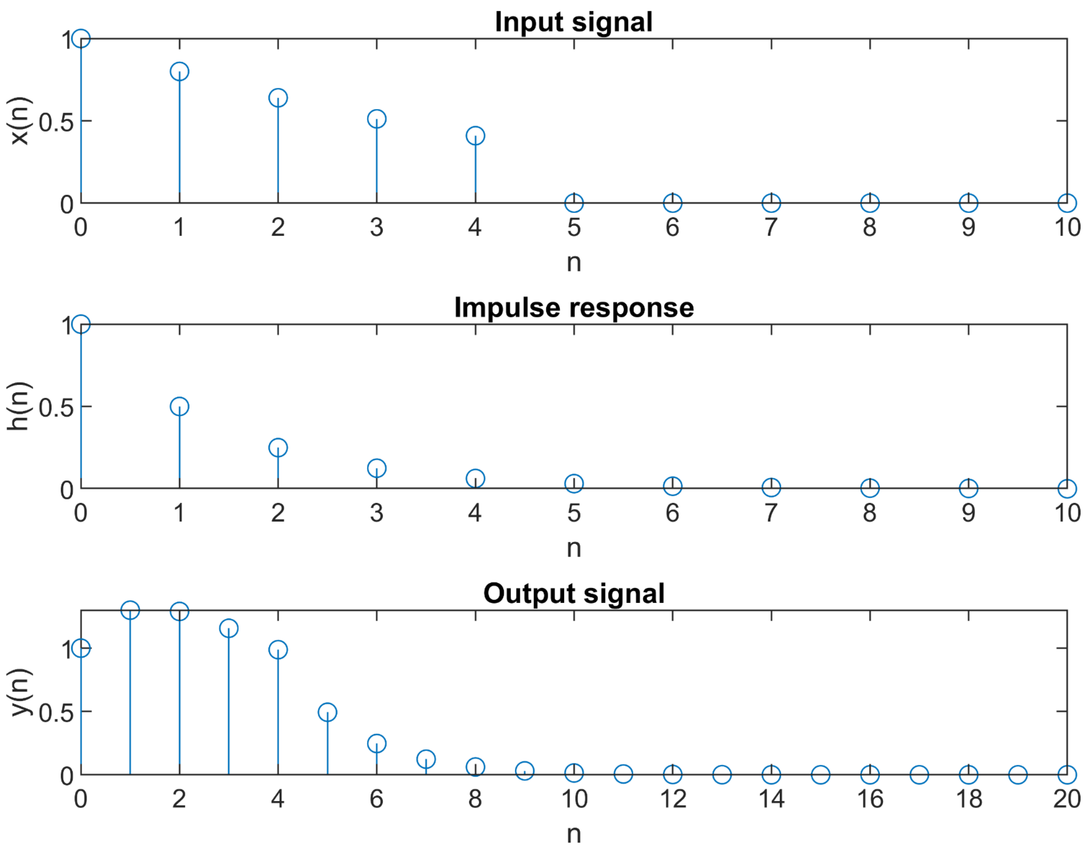

Figure 3.1.2 Input signal *x(n),* impulse response *h(n)* and out signal *y(n)*

3.2 Convolution sum and the z-transform

As it can be seen in figure 3.2.1, the coefficients of *Y(z)* are obtained by using the convolution sum function *conv(u,v)*.

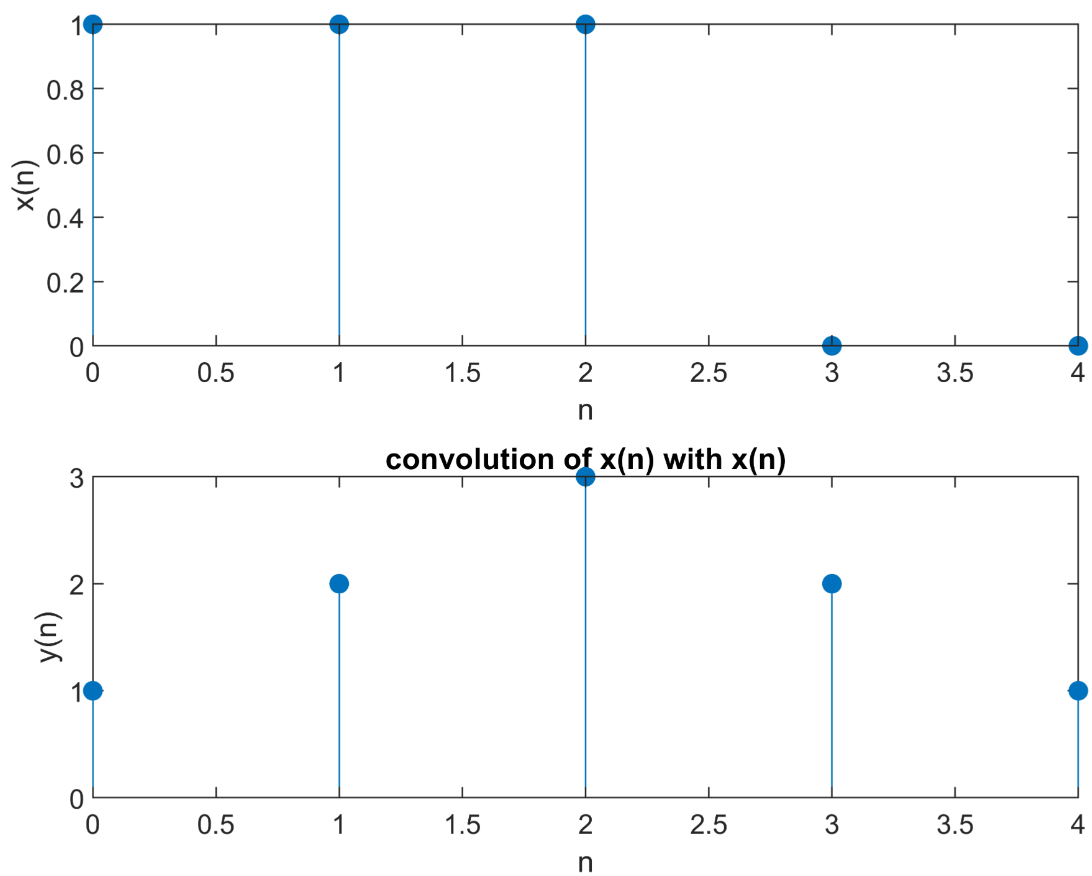

Figure 3.2.1 *x(n)* and coefficients of *Y(z)* obtained by *conv(u,v)* function

As it can be seen in the following table 2, there are three methods implemented to obtain the coefficients of *Y(z).* The second row shows the coefficients calculated by using the z-transform extracted directly from (18), the third row illustrates the calculation of the coefficients from table 1 and the fourth row are the coefficients obtained from figure 3.2.1. The results from these three rows coincides, therefore it is possible to say that the convolution sum is a fast way to find the coefficients of the polynomial resulting from the multiplication of two polynomials. Also, it is also possible to state that the convolution of two signals in the time domain is equivalent to the multiplication of the signals in the z domain as it is in (30).

$$x(n) = x_1(n) * x_2(n) \Leftrightarrow X(z) = X_1(z)X_2(z) \tag{30}$$

Table 2 Coefficients of *Y(z)* obtained by different methods

Coefficients of *Y(z)*	n=0	n=1	n=2	n=3	n=4
By z-transform	1	2	3	2	1
By convolution	1	1+1=2	1+1+1=3	1+1=2	1
By MATLAB function *conv(u,v)*	1	2	3	2	1

3.3 Inverse z-transform

By using the MATLAB function *iztrans(f),* the result of the inverse z-transform of *X(z)* is as equation (31). When evaluating (31) as n goes to infinity in MATLAB with the function *limit(f,var,a)*, it is calculated that the limit of it as *n* approaches infinity is equal to 0 as it can be seen in equation (32).

$$x(n) = 4(-\frac{1}{2})^{n} - 3(-\frac{1}{4})^{n} \tag{31}$$

$$\lim_{n\to\infty} x(n) = 0 \tag{32}$$

3.4 MATLAB partial fraction expansion

With the coefficients obtained in (26) and (27), the partial fraction expansion becomes as (33).

$$X(z) = \frac{-2}{1-z^{-1}} + \frac{2}{1-2z^{-1}} \tag{33}$$

Given equation (33), it is possible to determine its inverse z-transform with the help of the transform pairs (8) and (29). So the analytical result of the *x(n)* is as follows in equation (34).

$$x(n) =- 2u(n) + 2 \cdot 2^{n}u(n) \tag{34}$$

From figure 3.4.1, it can be observed that the analytical result is the same as the result given by MATLAB inverse z-transform function *izstrans(f),* so our answer is verified.

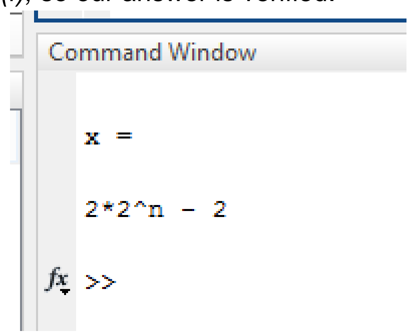

Figure 3.4.1 Result of the inverse z-transform of X(z) using *izstrans(f)* function

4 CONCLUSION

In conclusion, the z-transform was proven to be a powerful mathematical tool for solving and analysing discrete LTI systems since it is a fundamental transform for obtaining a more manageable form of a given function and being able to perform other mathematical operations on the result from the z domain. Once analysing the experiments signals in both MATLAB and conventional methods and obtaining the results, it is confirmed that z-transform is a very useful tool for discrete LTI systems and MATLAB has very powerful functions to obtain z-transform and inverse z-transform of a given signal.

5 REFERENCES

[1] Fadel, A. (2015). On Z-Transform and Its Applications. Retrieved 14 April 2021, from https://scholar.najah.edu/sites/default/files/Asma%20Belal%20Fadel_0.pdf
[2] Tan, L., & Jiang, J. (2013). Digital signal processing (2nd ed.). ELSEVIER.
[3] Jury, E. (1986). Theory and application of the z-transform method. Malabar, Fla.: R.E. Krieger.

6 APPENDIX

1. LTI systems: difference equations and the convolution sum

```
%%LTI systems: difference equations and the convolution sum
clc
clear all
```

```
close all
%% 1.1 Solving the       difference         equation
n=0:10; %interval from 0 to 10
%define y(0) and y(1), matlab does not have index 0, so they are shifted
%one index
y(0+1)=1;
y(1+1)=1;
x=4*cos(n*pi/8); %define x(n)
%define y(n)
for i=(2+1):11
    y(i)=y(i-1)+2*y(i-2)+x(i-2)
end

subplot(2,1,1)
stem(n, x);
title("Input signal");
xlabel("n");
ylabel("x(n)=4*cos((n*pi)/8)");
subplot(2,1,2)
stem(n, y);
title("Output signal");
xlabel("n");
ylabel("y(n)=y(n-1)+2*y(n-2)+x(n-2)");

%% 1.2 Solving the       difference         equation
n2=0:10; %interval of n from 0 to 10

%Define input signal x(n),impuse response h(n) and output signal
%y(n) as the convolution of x(n) and h(n)
x2=(0.8.^n2).*(ones(1,11)-[zeros(1,5) ones(1,11-5)]);
h2=(0.5.^n2).*(ones(1,11)-[zeros(1,10) ones(1,11-10)]);
y2=conv(x2,h2);

figure
subplot(3,1,1)
stem(n2, x2);
title("Input signal");
xlabel("n");
ylabel("x(n)");
subplot(3,1,2)
stem(n2, h2);
title("Impulse response");
xlabel("n");
ylabel("h(n)");
subplot(3,1,3)
stem(0:(length(y2)-1), y2);
title("Output signal");
xlabel("n");
ylabel("y(n)");
```

2. Convolution sum and the z-transform

```
%%Convolution sum and the z-transform
clc
clear all
close all
```

```
%Convolution of x(n) with itself
x=ones(1,3);
yConv=conv(x,x) %show coefficients by convolution

figure
subplot(2,1,1)
stem(0:4,[x 0 0],'filled');
title("Signal x(n)");
xlabel("n");
ylabel("x(n)");
subplot(2,1,2)
stem(0:4,yConv,'filled');
title("convolution of x(n) with x(n)");
xlabel("n");
ylabel("y(n)");
```

3. Inverse z-transform

```
%%Inverse z-transform
clc
clear all
close all

%Use symbolic MATLAB to find the inverse Z-transform x
syms n z
x= iztrans((2-z^(-1))/(2*(1+0.25*z^(-1))*(1+0.5*z^(-1))));
pretty(x)

%evaluate x(n) as n goes to infinity
x_ninf=limit(x,n,inf)
```

4. MATLAB partial fraction expansion

```
%%MATLAB partial fraction expansion
clc
clear all
close all

%Use symbolic MATLAB to find the inverse Z-transform x
syms n z

%obtain inverse transform
x= iztrans((2*z^(-1))/((1-z^(-1))*(1-2*z^(-1))))
```