EE-384: Digital Signal Processing **Spring 2023**

Laboratory 3: z –transform

Instructor: Mr Ammar Naseer *EE UET New Campus*

Aims

This lab session provides signal and system description in the complex frequency domain. MATLAB techniques are introduced to analyze z-transforms, inverse z-transforms and system representation in terms of their transfer functions. Solutions of difference equations using the z-transform and MATLAB are also provided.

Pre-Lab:

The z-Transform

The z-transform of a sequence x(n) is given by

$$X(z) = \mathcal{Z}[x(n)] = \sum_{n=-\infty}^{\infty} x(n) z^{-n} \qquad (4.1)$$

where z is a complex variable. The set of z values for which X(z) exists is called the region of convergence (ROC) and is given by

$$R_1 < |z| < R_2 \qquad (4.2)$$

for some non-negative numbers R_1 and R_2.

The inverse z-transform of a complex function X(z) is given by

$$x(n) = \mathcal{Z}^{-1}[x(n)] = \frac{1}{2\pi j} \oint_C X(z) z^{n-1} dz \qquad (4.3)$$

where C is a counterclockwise contour encircling the origin and lying in the ROC.

Comments:

1. The complex variable z is called the complex frequency given by $z = |z|e^{j\omega}$!, where $|z|$ is the magnitude and ω is the real frequency.

2. Since the ROC (4.2) is defined in terms of the magnitude $|z|$ the shape of the ROC is an open ring. Note that R_1 may be equal to zero and/or R_2 could possibly be ∞.
3. If $R_2 < R_1$, then the ROC is a null space and the z-transform does not exist.
4. The function $|z| = 1$ (or $z = e^{j\omega}$) is a circle of unit radius in the z-plane and is called the unit circle. If the ROC contains the unit circle, then we can evaluate X(z) on the unit circle.

$$X(z)|_{z=e^{j\omega}} = X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{j\omega n} = \mathcal{F}[x(n)]$$

Therefore the discrete-time Fourier transform $X(e^{j\omega})$ may be viewed as a special case of the z-transform X(z).

Properties of the ROC

Note that the ROC is a distinguishing feature that guarantees the uniqueness of the z- transform. Hence it plays a very important role in system analysis.

1. The ROC is always bounded by a circle since the convergence condition is on the magnitude $|z|$.
2. The sequence $x(n) = a^n u(n)$ is a special case of a right-sided sequence, defined as a sequence x(n) that is zero for some $n < n_0$. The ROC for right-sided sequences is always outside of a circle of radius R_1. If $n_0 \geq 0$, then the right-sided sequence is also called a causal sequence.
3. The sequence $x(n) = b^n u(-n-1)$ is a special case of a left-sided sequence, defined as a sequence x(n) that is zero for some $n > n_0$. If $n_0 \geq 0$, the resulting sequence is called an anticausal sequence. The ROC for left-sided sequences is always inside of a circle of radius R_2.
4. The ROC for two-sided sequences is always an open ring $R_1 < |z| < R_2$, if it exists.
5. The sequences that are zero for $n < n_1$ and $n > n_2$ are called finite-duration sequences. The ROC for such sequences is the entire z-plane. If $n_1 < 0$, then $z = 1$ is not in the ROC. If $n_2 > 0$, then $z = 0$ is not in the ROC.
6. The ROC cannot include a pole since X(z) converges uniformly in there.
7. There is at least one pole on the boundary of a ROC of a rational X(z).
8. The ROC is one contiguous region; that is, the ROC does not come in pieces. In digital signal processing, signals are assumed to be causal since almost every digital data is acquired in real time. Therefore the only ROC of interest to us is the one given in statement 2.

Important Properties of the z-Transform

We state the following important properties of the z-transform without proof.

1. **Linearity:**

$$\mathcal{Z}[a_1x_1(n) + a_2x_2(n)] = a_1X_1(z) + a_2X_2(z); \quad ROC : ROC_{x_1} \cap ROC_{x_2} \qquad (4.4)$$

2. **Sample Shifting**

$$\mathcal{Z}[x(n - n_0)] = z^{n_0}X(z); \quad ROC : ROC_x \qquad (4.5)$$

3. **Frequency Shifting**

$$\mathcal{Z}[a^n x(n)] = X\left(\frac{z}{a}\right); \quad ROC : ROC_x \text{ scaled by } |a| \qquad (4.6)$$

4. Folding:

$$\mathcal{Z}[x(n)] = X(1/z); \quad ROC: \text{ Inverted } ROC_x \qquad (4.7)$$

5. Complex Conjugation:

$$\mathcal{Z}[x^*(n)] = X^*(z^*); \quad ROC: ROC_x \qquad (4.8)$$

6. Differentiation in z-domain:

$$\mathcal{Z}[nx(n)] = -z\frac{dX(z)}{dz}; \quad ROC: ROCx \qquad (4.9)$$

7. Multiplication:

$$\begin{aligned}\mathcal{Z}[x_1(n)x_2(n)] &= \frac{1}{2\pi j}\oint_C X_1(\nu)X_2(z/\nu)\nu^{-1}d\nu; \qquad (4.10)\\ &= ROC: \quad ROC_{x_1} \cap \text{ Inverted } ROC_{x_2}\end{aligned}$$

where C is a closed contour that encloses the origin and lies in the common ROC.

8. Convolution

$$\mathcal{Z}[x_1(n) * x_2(n)] = X_1(z)X_2(z); \quad ROC: ROC_{x_1} \cap ROC_{x_2} \qquad (4.11)$$

This last property transforms the time-domain convolution operation into a multiplication between two functions. It is a significant property in many ways. First, if $X_1(z)$ and $X_2(z)$ are two polynomials, then their product can be implemented using the **conv** function in MATLAB.

Let $X_1(z) = 2 + 3z^{-1} + 4z^{-2}$ and $X_2(z) = 3 + 4z^{-1} + 5z^{-2} + 6z^{-3}$. Determine $X_3(z) = X_1(z)X_2(z)$.

From the definition of the z-transform, we observe that

$$\mathrm{x}_1[\mathrm{n}] = [\underset{\uparrow}{2}, 3, 4] \text{ and } x_2[\mathrm{n}] = [\underset{\uparrow}{3}, 4, 5, 6]$$

Then the convolution of these two sequences will give the coefficients of the required polynomial product.

```
>> x1 = [2,3,4];
>> x2 = [3,4,5,6];
>> x3 = conv(x1,x2)
x3 =
    6   17   34   43   38   24
```

Hence

$$X_2(z) = 6 + 17z^{-1} + 34z^{-2} + 43z^{-3} + 38z^{-4} + 24z^{-5}$$

Note that using the **conv_m** function developed in Lab 3, we can also multiply two z-domain polynomials corresponding to noncausal sequences.

3A: Let $\mathrm{X}_1(\mathrm{z}) = \mathrm{z} + 2 + 3\mathrm{z}^{-1}$ and $\mathrm{X}_2(\mathrm{z}) = 2\mathrm{z}^2 + 4\mathrm{z} + 3 + 5\mathrm{z}^{-1}$. Determine $X_3(z) = X_1(z)X_2(z)$.

In passing we note that to divide one polynomial by another one, we would require an inverse operation called deconvolution. In MATLAB **[p,r] = deconv(b,a)** computes the result of dividing b by a in a polynomial part p and a remainder r. For example, if we divide the polynomial $X_3(z)$ by $X_1(z)$, as follows

```
>> x3 = [6,17,34,43,38,24];
>> x1 = [2,3,4];
```

```
>> [x2,r] = deconv(x3,x1)
x2 =
    3    4    5    6
r =
    0    0    0    0    0    0
```

then we obtain the coefficients of the polynomial $X_2(z)$ as expected. To obtain the sample index, we will have to modify the **deconv** function as we did in the **conv_m** function. This operation is useful in obtaining a proper rational part from an improper rational function. Using z-transform properties and the z-transform table, let us determine the z-transform of

Using z-transform properties and the z-transform table, let us determine the z-transform of

$$x(n) = (n-2)(0.5)^{(n-2)}\cos\left[\frac{\pi}{3}(n-2)\right]u(n-2)$$

Applying sample shift property

$$X(z) = \mathcal{Z}[(n)] = z^{-2}\mathcal{Z}\left[n(0.5)^n\cos\left(\frac{\pi n}{3}\right)u(n)\right]$$

with no change in the ROC. Applying the multiplication by a ramp property

$$X(z) = z^{-2}\left\{-z\frac{d\mathcal{Z}\left[n(0.5)^n\cos\left(\frac{\pi n}{3}\right)u(n)\right]}{dz}\right\}$$

with no change in the ROC. Now the z-transform of $(0.5)^n\cos(\frac{\pi n}{3})u(n)$ from Table is

$$\begin{aligned}\mathcal{Z}\left[(0.5)^n\cos\left(\tfrac{\pi n}{3}\right)u(n)\right] &= \frac{1-(0.5\cos\pi/3)z^{-1}}{1-2(0.5\cos\pi/3)z^{-1}+0.25z^{-2}}; \quad |z|>0.5\\ &= \frac{1-0.25z^{-1}}{1-0.5z^{-1}+0.25z^{-2}}; \quad |z|>0.5\end{aligned}$$

or

$$\begin{aligned}X(z) &= -z^{-1}\frac{d}{dz}\left\{\frac{1-0.25z^{-1}}{1-0.5z^{-1}+0.25z^{-2}}\right\}\\ &= \frac{0.25z^{-3}-0.5z^{-4}+0.0625z^{-5}}{1-z^{-1}+0.75z^{-2}-0.25z^{-3}+0.0625z^{-4}}; \quad |z|>0.5\end{aligned}$$

MATLAB verification: To check that this X(z) is indeed the correct expression, let us compute the first 8 samples of the sequence x(n) corresponding to X(z), as discussed before

```
>> b = [0,0,0,0.25,-0.5,0.0625];
>> a = [1,-1,0.75,-0.25,0.0625];
>> [x,n]=impseq(0,7,0);
>> x=filter(b,a,x)
x =
       0        0        0   0.2500   -0.2500   -0.3750   -0.1250    0.0781
>> x = [(n-2).*(1/2).^(n-2).*cos(pi*(n-2)/3)].*stepseq(0,7,2)
```

```
x =
     0     0     0   0.2500  -0.2500  -0.3750  -0.1250   0.0781
```

This approach can be used to verify the z-transform computations.

Inversion of the z-Transform

From equation (4.3), the inverse z-transform computation requires an evaluation of a complex contour integral that, in general, is a complicated procedure. The most practical approach is to use the partial fraction expansion method. It makes use of the z-transform table. The z-transform, however, must be a rational function. This requirement is generally satisfied in digital signal processing.

A MATLAB function **residuez** is available to compute the residue part and the direct (or polynomial) terms of a rational function inz^{-1}. Let

$$\begin{aligned} X(z) &= \frac{b_0 + b_1 z^{-1} + \dots + b_M z^{-M}}{a_0 + a_1 z^{-1} + \dots + a_N z^{-N}} = \frac{B(z)}{A(z)} \\ &= \sum_{k=1}^{N} \frac{R_k}{1 - z^{-1} p_k} + \sum_{k=0}^{M-N} C_k z^{-k} \end{aligned}$$

be a rational function in which the numerator and the denominator polynomials are in ascending powers of z^{-1}. Then **[R,p,C]=residuez(b,a)** computes the residues, poles, and direct terms of X(z) in which two polynomials B(z) and A(z) are given in two vectors b and a, respectively. The returned column vector R contains the residues, column vector p contains the pole locations, and row vector C contains the direct terms. If p(k)=...=p(k+r-1) is a pole of multiplicity r, then the expansion includes the term of the form

$$\frac{R_k}{1 - p_k^{-1}} + \frac{R_{k+1}}{(1 - p_k^{-1})^2} + \dots + \frac{R_{k+r-1}}{(1 - p_k^{-1})^r}$$

Similarly, **[b,a]=residuez(R,p,C)**, with three input arguments and two output arguments, converts the partial fraction expansion back to polynomials with coefficients in row vectors b and a.

To check our residue calculations, let us consider the rational function

$$X(z) = \frac{z}{3z^2 - 4z + 1}$$

First rearrange X(z) so that it is a function in ascending powers of z^{-1}.

$$X(z) = \frac{0 + z^{-1}}{3 - 4z^{-1} + z^{-2}}$$

Now using the MATLAB script

```
>> b = [0,1];
>> a = [3,-4,1];
>> [R,p,C] = residuez(b,a)
```

```
R =
   0.5000
  -0.5000
p =
   1.0000
   0.3333
C =
    []
```

we obtain

$$X(z) = \frac{\frac{1}{2}}{1 - z^{-1}} - \frac{\frac{1}{2}}{1 - \frac{1}{3}z^{-1}}$$

Similarly, to convert back to the rational function form,

```
>> [b,a] = residuez(R,p,C)
b =
  -0.0000    0.3333
a =
   1.0000   -1.3333    0.3333
```

so that

$$X(z) = \frac{0 + \frac{1}{3}z^{-1}}{1 - \frac{4}{3}z^{-1} + \frac{1}{3}z^{-2}} = \frac{z^{-1}}{3 - 4z^{-1} + z^{-2}}$$

Compute the inverse z -transform of

$$X(z) = \frac{1}{(1 - 0.9z^{-1})^2(1 + 0.9z^{-1})} \qquad |z| > 0.9$$

We will evaluate the denominator polynomial as well as the residues using the MATLAB script:

```
>> b = 1;
>> a = poly([0.9,0.9,-0.9])
a =
   1.0000   -0.9000   -0.8100    0.7290
>> [R,p,C]=residuez(b,a)
R =
  0.2500 + 0.0000i
  0.5000 - 0.0000i
  0.2500 + 0.0000i
p =
  0.9000 + 0.0000i
  0.9000 - 0.0000i
 -0.9000 + 0.0000i
C =
```

```
[]
```

Note that the denominator polynomial is computed using MATLABs polynomial function **poly**, which computes the polynomial coefficients, given its roots. We could have used the c**onv** function, but the use of the **poly** function is more convenient for this purpose. From the residue calculations and using the order of residues we have

$$X(z) = \frac{0.25}{(1-0.9z^{-1})} + \frac{0.5}{(1-0.9z^{-1})^2} + \frac{0.25}{(1-0.9z^{-1})} \qquad |z| > 0.9$$

$$X(z) = \frac{0.25}{(1-0.9z^{-1})} + \frac{0.5}{0.9} z \frac{0.9z^{-1}}{(1-0.9z^{-1})^2} + \frac{0.25}{(1-0.9z^{-1})} \qquad |z| > 0.9$$

Using the z-transform property of time-shift,

$$x(n) = 0.25(0.9)^n u(n) + \frac{5}{9}(n+1)(0.9)^{n+1} u(n+1) + 0.25(0.9)^n u(n)$$

which, upon simplification, becomes

$$x(n) = 0.75(0.9)^n u(n) + 0.5n(0.9)^n u(n) + 0.25(0.9)^n n u(n)$$

MATLAB verification:

```
[x,n] = impseq(0,7,0);
>> x = filter(b,a,x)
x =
   1.0000    0.9000    1.6200    1.4580    1.9683    1.7715    2.1258    1.9132
>> x = (0.75)*(0.9).^n + (0.5)*n.*(0.9).^n + (0.25)*(-0.9).^n
x =
   1.0000    0.9000    1.6200    1.4580    1.9683    1.7715    2.1258    1.9132
```

Determine the inverse z-transform of

$$X(z) = \frac{1 + 0.4\sqrt{2}z^{-1}}{(1 - 0.8\sqrt{2}z^{-1} + 0.64z^{-2})}$$

so that the resulting sequence is causal and contains no complex numbers.

System Representation in the *z*-Domain

The system function H(z) is given by

$$H(z) := \mathcal{Z}[h(n)] = \sum_{\infty}^{\infty} h(n) z^{-n}; \quad r_1 < |z| < r_2 \qquad (4.12)$$

Using the convolution property of the z-transform, the output transform Y (z) is given by

$$Y(z) = X(z)H(z) \quad : ROC_x \cap ROC_h \tag{4.13}$$

provided ROC_x overlaps with ROC_h. On the other hand, when LTI systems are described by a difference equation

$$y(n) + \sum_{k=1}^{N} a_k y(n-k) = \sum_{l=0}^{M} b_l x(n-l) \tag{4.14}$$

the system function H(z) can easily be computed. Taking the z -transform of both sides, and using properties of the z -transform,

$$\begin{aligned} H(z) := \frac{Y(z)}{X(z)} &= \frac{\sum_{l=0}^{M} b_l z^{-l}}{1 + \sum_{k=1}^{N} a_k z^{-1}} \\ &= \frac{b_0 z^{-M}(z^M + \dots + \frac{b_M}{b_0})}{z^{-N}(z^N + \dots + a_N)} \\ &= b_0 z^{N-M} \frac{\prod_{l=1}^{M}(z - z_l)}{\prod_{k=1}^{N}(z - p_k)} \end{aligned} \tag{4.15}$$

where z_l are the system zeros and p_k are the system poles. Thus H(z) (and hence an LTI system) can also be represented in the z-domain using a pole-zero plot. This fact is useful in designing simple filters by proper placement of poles and zeros.

To determine zeros and poles of a rational H(z), we can use the MATLAB function roots on both the numerator and the denominator polynomials. (Its inverse function poly determines polynomial coefficients from its roots, as discussed in the previous section.) It is also possible to use MATLAB to plot these roots for a visual display of a pole-zero plot. The function zplane(b,a) plots poles and zeros, given the numerator row vector b and the denominator row vector a. As before, the symbol o represents a zero and the symbol x represents a pole. The plot includes the unit circle for reference. Similarly, zplane(z,p) plots the zeros in column vector z and the poles in column vector p. Note very carefully the form of the input arguments for the proper use of this function.

Transfer Function Representation

If the ROC of H(z) includes a unit circle ($z = e^{j\omega}$), then we can evaluate H(z) on the unit circle, resulting in a frequency response function or transfer function $H(e^{j\omega})$. Then

$$H(e^{j\omega}) = b_0 e^{jN-M\omega} \frac{\prod_{k=1}^{M}(e^{j\omega} - z_l)}{\prod_{k=1}^{N}(e^{j\omega} - p_k)} \tag{4.16}$$

The factor ($e^{j\omega}$- z_l) can be interpreted as a vector in the complex z-plane from a zero z_l to the unit circle at $z = e^{j\omega}$, while the factor ($e^{j\omega}$ - p_k) can be interpreted as a vector from a pole p_k to the unit circle at $z = e^{j\omega}$. Hence the magnitude response function

$$\left|H(e^{j\omega})\right| = |b_0| \frac{|e^{j\omega} - z_1| \dots |e^{j\omega} - z_M|}{|e^{j\omega} - p_1| \dots |e^{j\omega} - p_N|} \qquad 4.17$$

can be interpreted as a product of the lengths of vectors from zeros to the unit circle divided by the lengths of vectors from poles to the unit circle and scaled by $|b_0|$. Similarly, the phase response function

$$\angle H(e^{j\omega}) = [0 \; or \; \pi] + [(N-M)\omega] + \sum_{1}^{M} \angle(e^{j\omega} - z_k) - \sum_{1}^{N} \angle(e^{j\omega} - p_k) \qquad 4.18$$

can be interpreted as a sum of a constant factor, a linear-phase factor, and a nonlinear-phase factor (angles from the "zero vectors" minus the sum of angles from the "pole vectors")

MATLAB provides a function called **freqz** for this computation, which uses the preceding interpretation. In its simplest form, this function is invoked by

```
>> [H,w] = freqz(b,a,N)
```

which returns the N-point frequency vector w and the N-point complex frequency response vector H of the system, given its numerator and denominator coefficients in vectors b and a. The frequency response is evaluated at N points equally spaced around the upper half of the unit circle. Note that the b and a vectors are the same vectors we use in the filter function or derived from the difference equation representation.

The second form

```
>> [H,w] = freqz(b,a,N,'whole')
```

uses N points around the whole unit circle for computation.

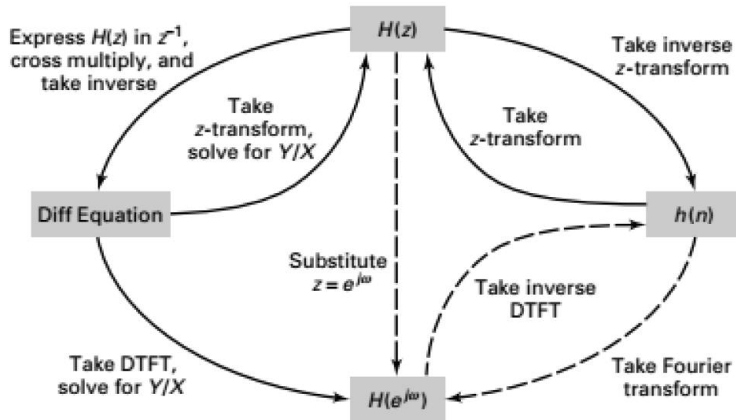
In yet another form

```
>> H = freqz(b,a,w)
```

it returns the frequency response at frequencies designated in vector w, normally between 0 and π. It should be noted that the **freqz** function can also be used for numerical computation of the DTFT of a finite-duration, causal sequence x(n). In this approach, b = x and a =1.

Stability and Causality

For LTI systems, the BIBO stability is equivalent to $\sum_{-\infty}^{\infty}|h(k)| < \infty$. From the existence of the discrete time Fourier transform, this stability implies that H(ej!) exists, which further implies that the unit circle jzj = 1 must be in the ROC of H(z). This result is called the z-domain stability theorem; therefore the dashed paths in Figure 5.1 exist only if the system is stable.



Main Lab

3B: Determine the z -transform of the following sequences using the definition (4.1). Indicate the region of convergence for each sequence and verify the z -transform expression using MATLAB.

1. $\mathrm{x(n)} = (0.8)^n u(n-2)$
2. $\mathrm{x(n)} = [(0.5)^n + (-0.8)^n]u(n)$

3C: Given a causal system

$$y(n) = 0.9y(n-1) + x(n)$$

3. Write a function that would determine H(z) and sketch its pole-zero plot.
4. Plot $|H(e^{j\omega})|$ and $< \ H(e^{j\omega})$

Rubrics

Rubrics	Marks	0	1	2	3	4	5	6
R1	Lab performance	Absent	Could not create any code in Lab	Solve few tasks in Lab	Solve half of the tasks in Lab	Solve most of the task in Lab	Solve all of the task in Lab but with few minor error in code	Formulate all three task during Lab Time
R2	Code Understanding	Absent	Could not make proper logic	Understanding of logic but problem in code creation	Understanding of logic and code but with minor flaws in logic creation	Perfect understand ing of tasks	–	–