TECH 350: DSP

Class VII: Transfer Functions, Complex Numbers, Prerequisites for Fourier Transform

But first, *REVIEW?*

Review: z-Transform

We define ***Z*** such that powers of Z correspond to the number of delays or advances in a difference equation.

One sample of delay = Z. Two samples of delay = Z^2.

One sample of advance = Z^{-1}. Two samples of advance = Z^{-2}.

Ex.:

If we're given a sequence x(t) = [1 2 3 4], its z-Transform, $X(Z) = 1 + 2Z + 3Z^2 + 4Z^3$

Ex.:

If we're given a sequence x(t) = [1 3 0 -1 -5], its z-Transform, $X(Z) = 1 + 3Z - Z^3 - 5Z^4$

Review: The Convolution Theorem

The convolution of f and g is written f*g.

An intuitive way to compute convolution by hand on 1-dimensional vectors is to think of it as a sliding window of multiplication and summation by a flipped 'kernel'. For example:

$x_1 = (1, 5)$

$x_2 = (1, 2, 3)$

$x_1 * x_2 = (1, 7, 13, 15)$

```
Convolution by hand (metaphor of sliding window):

5 1
  1 2 3
= 1 x 1 = 1

5 1
1 2 3
= 5 x 1 + 1 x 2 = 7

  5 1
1 2 3
= 5 x 2 + 1 x 3 = 13

    5 1
1 2 3
= 5 x 3 = 15
```

Review: The Convolution Theorem

$x_1 = (1, 5)$

$x_2 = (1, 2, 3)$

z-Transforms of these:

$x_1(Z) = 1 + 5Z$

$x_2(Z) = 1 + 2Z + 3Z^2$

Multiplying their z-Transforms together:

$x_1(Z)x_2(Z) = 1 + 7Z + 13Z^2 + 15Z^3$

… which is the z-Transform of the convolution of x_1 and X_2!!

Review: The Convolution Theorem

The Convolution Theorem: *convolution in the time domain is equivalent to multiplication in the z domain.*

How does this help us?

Well, remember impulse responses of LTI systems?

A very important application of the convolution theorem: any signal x(n) that is input to an LTI system, the system's output y(n) is equal to the discrete convolution of the input signal x(n) and the system's impulse response (IR) h(n).

So, if x_1 = input signal and x_2 = a systems's impulse response, we just figured out its output!

LTI System: Using Input + FIR to get Output

System S is defined by the diff. equation:

S: y(n) = 1 x(n) + 2 x(n-1) + 3 x(n-2)

The FIR of S, IR(S) is: [1 2 3]

If we want the output of S, Y, given the input X = [1 2 3]:

IR(S) * X = [1 4 10 12 9]

Checking that this is equal to the output of S given X as input…

LTI System: Using Input + FIR to get Output

System S is defined by the diff. equation:

S: y(n) = 1 x(n) + 2 x(n-1) + 3 x(n-2)

Checking that this is equal to the output of S given X as input:

X(1) = 1, Y(1) = 1 * 1 + 0 + 0 = 1

X(2) = 2, Y(2) = 2 * 1 + 1 * 2 + 0 = 4

X(3) = 3, Y(3) = 3 * 1 + 2 * 2 + 1 * 3 = 10

X(4) = ø, Y(4) = 0 * 1 + 3 * 2 + 2 * 3 = 12

X(5) = ø, Y(5) = 0 * 1 + 0 * 2 + 3 * 3 = 9

Y = [1 4 10 12 9], woo!

Transfer Functions

Give x(n) as input and y(n) as output, the ***transfer function,*** H(z), may be written as:

$$H(z) = \frac{Y(z)}{X(z)}$$

Where X(z) = Z(x(n)) and Y(z) = Z(y(n)) (z-Transforms of x(n) and y(n))

This follows from the fact that the impulse response convolved with the input is the output:

H(z) is equal to the z-Transform of the impulse response h(n) (neat!)

Transfer Functions

Transfer Function*: provides an algebraic representation of an LTI filter in the *frequency domain

Before we look at an example:

There are two different conventions for denoting delays and advances:

Laplace: uses positive powers (e.g. Z^2) for delay and negative powers (e.g. Z^{-2})

Engineering: uses negative powers (e.g. Z^{-2}) for delay and positive powers (e.g. Z^2)

I felt that positive powers was useful for explaining the z-Transform, but we will proceed with the engineering convention moving forward

Transfer Function: Example 1

Difference Equation: y(n) = a x(n) + b x(n-1) + c y(n-1) (remember a, b, and c are coefficients)

Transfer Function:

first, get the z-Transform of the part of the diff. equation with x(n) (for the denominator):

Z(a x(n) + b x(n-1)) = 1a + bZ^{-1} = a + bZ^{-1}

second, get the z-Transform of the part of the diff. equation with y(n) (for the numerator):

Z(c y(n-1)) = 1 - cZ^{-1} (y(n) is on the left side, we move c y(n-1) to the left)

third, combine these:

H(n) = 1 - cZ^{-1} / a + bZ^{-1}

Transfer Function: Example 2

Difference Equation (biquad): $y(n) = a_0x(n) + a_1x(n-1) + a_2x(n-2) - b_1y(n-1) - b_2y(n-2)$

first, get the z-Transform of the part of the diff. equation with x(n) (for the denominator)

second, get the z-Transform of the part of the diff. equation with y(n) (for the numerator)

third, combine these

Zeros + Poles (after JOS)

We may write a general form of the transfer function (with the leading coefficient in the numerator called g) as such:

$$H(z) = g\frac{1 + \beta_1 z^{-1} + \cdots + \beta_M z^{-M}}{1 + a_1 z^{-1} + \cdots + a_N z^{-N}}$$

We may then factor the numerator and denominator to obtain:

$$H(z) = g\frac{(1 - q_1 z^{-1})(1 - q_2 z^{-1})\cdots(1 - q_M z^{-1})}{(1 - p_1 z^{-1})(1 - p_2 z^{-1})\cdots(1 - p_N z^{-1})}.$$

If z is set to any of the numbers q_1, q_2, …q_M (what are called the *zeros*), the transfer function evaluates to 0.

As z approaches any of p_1, p_2, …p_M (what are called the *poles*), the transfer function approaches infinity.

Zeros + Poles (after JOS)

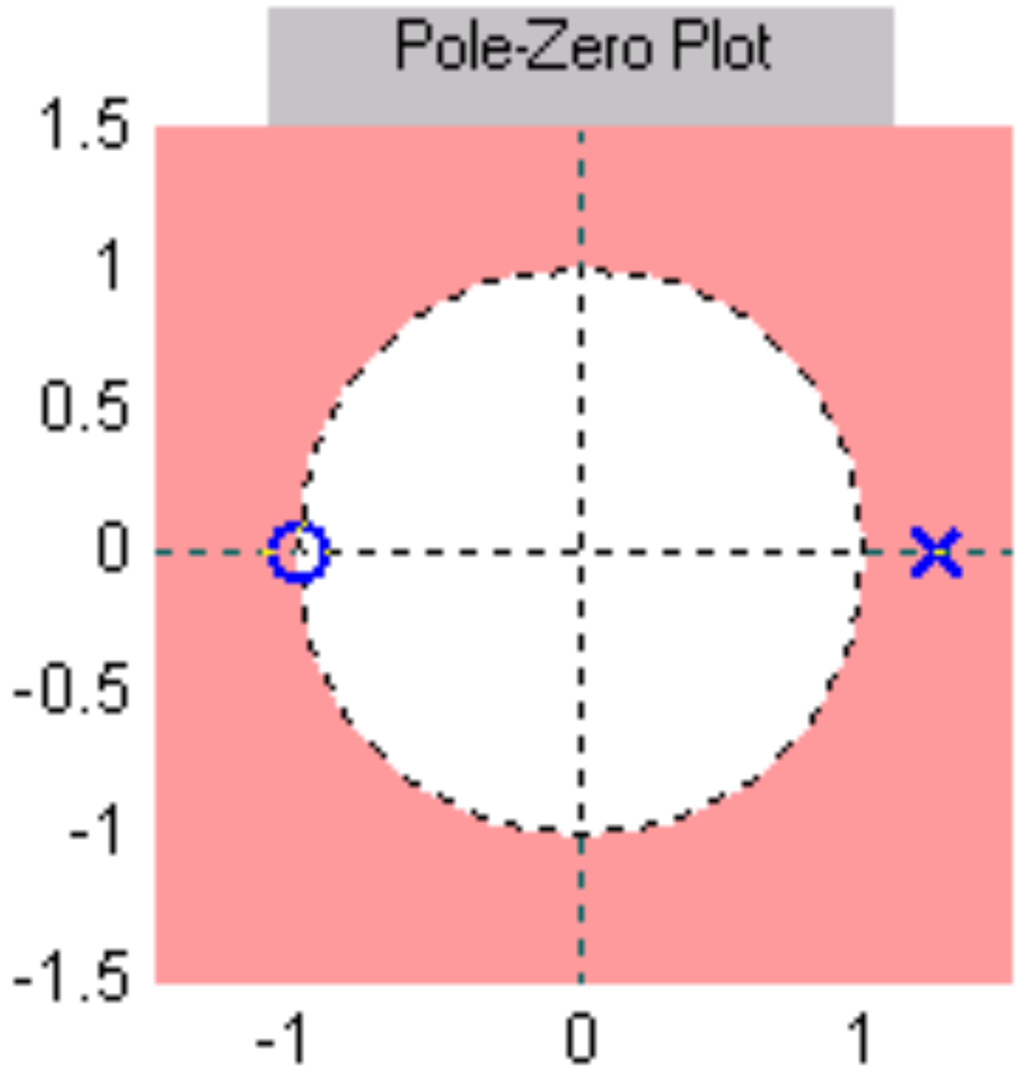
For Computer Musicians:

That's why, if you've screwed around with filters and coefficients, they've either suddenly gone silent (and need some coddling to get sounding again) (**zeros**) or they explode into deafening feedback (**poles**)

We can additionally use zeroes and poles to describe filters, e.g.:

"The biquad filter is a two-pole, two-zero filter"

$$H(z) = g\frac{1 + \beta_1 z^{-1} + \beta_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$



Plot in the z-Domain that shows where the filter goes to zero (zeros, Os) and trends to infinity (poles, Xs)