The Z Transform

A Crash Course

Brian Frost-LaPlante

Spring 2018

This document is to serve as a supplementary document for students in ECE-210 who are not taking/have not taken ECE-211. It walks through only the concepts of discrete-time signal processing necessary for the scope of this course, and furthermore, understanding the applications of MATLAB in engineering industry.

Discrete-Time Signals

Consider the space of real-valued functions with domain $\mathbb{Z}$. We call such functions *discrete-time signals*, but don't get hung up on this name; the input is not necessarily a "time" quantity. To distinguish discrete-time signals from real- valued functions in the usual sense, we denoted signal x evaluated at integer n by $x[n]$. Functions of this type are incredibly useful as, with some basic assumptions made, we can justly represent them with vectors in a computer programming language.

This is a fairly important concept in computer simulation and data processing: computations made with computers cannot, realistically, involve irrational numbers, and as such, the representation of a real-valued function on a computer is quantized in two senses: the domain is discretized and the function value is quantized. The consideration of discrete-time signals is, in effect, the consideration of the first of these quantizations. We see this constantly in MATLAB; the use of the *linspace* function to create an "interval of the real line" as a finite-length vector is exactly this kind of discretization.

The processing of such signals by linear time-invariant (LTI) systems is of great interest, as these days, most designed systems will be running on FPGAs or some other quantized computing device. Put simply, a discrete-time LTI system takes a discrete- time signal as an input and returns a discrete-time signal as an output, and is described by *discrete-time convolution* with a discrete-time signal h. That is to say that a discrete-time LTI system is a mapping from discrete-time signals to discrete-time signals, and for input $x[n]$, the output $y[n]$ is given by

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

The signal h is called the *impulse response* of the system. Note that, if x and h are being represented as, say, vectors in MATLAB, this infinite sum is really a finite sum (as x and h are finite-length) and we make the assumption that they take value 0 outside of the domain on which we have defined them. Convolution can be computed in MATLAB using the function *conv*. Convolution is a binary operation, which maps two discrete-time signals to a discrete-time signal, and is commutative.

The Z Transform

Those of you who have taken Differential Equations should recall a similar operation also called convolution for functions with domain $\mathbb{R}$. You will recall that this operation had a very special property in that its Laplace transform was simply the product of the Laplace transforms of the two argument functions. Presented here is the Z transform, which is analogous to the Laplace transform for signals with domain $\mathbb{Z}$. Given $x[n]$, a discrete-time signal, the Z transform of x is a complex-valued function of a complex variable, X, given by

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

should this series converge in some region of $\mathbb{C}$. Generally, this series may converge to different functions in different annular regions of convergence, and as such, when referring to *the Z transform* of a signal, the region of convergence $R_1 < |z| < R_2$ should be specified. The Z transform is not usually computed by hand from this formula, but rather from some combination of known Z transforms and the properties it has.

Why do we care about the Z transform though? One important property is that it maps convolutions to products. That is to say that for an LTI system with impulse response h, for each input x, the output y has Z transform

$$Y(z) = X(z)H(z)$$

In many cases, this is much simpler to compute than a convolution. We also can gain a lot of insight about a system by looking at the Z transform of its impulse response, $H(z)$. Namely, we take interest in the locations of its poles: points at which it is not defined. The regions of convergence of H are generally annular rings whose inner and outer radii are determined by the positions of poles in the complex plane. A region of convergence contains no poles. The MATLAB function *zplane* makes it easy to see where these poles lie, and thereby where the regions of convergence lie. We say a system is stable (a bounded input yields a bounded output) if the region of convergence contains the unit circle, $|z| = 1$. We say a system is causal if the region of convergence contains the point at ∞, i.e. there is no pole at ∞. Causality simply means that, in a physical sense, the output of the system will not depend on future outputs of the same system.