Z-TRANSFORM ANALYSIS OF DISCRETE-TIME FILTERS*

Mark A. Davenport

This work is produced by OpenStax-CNX and licensed under the Creative Commons Attribution License 3.0^{\dagger}

1 z-transform analysis of discrete-time filters

The z-transform might seem slightly ugly. We have to worry about the region of convergence, and we haven't even talked about how to invert it yet (it isn't pretty). However, in the end it is worth it because it is extremely useful in analyzing digital filters with feedback. For example, consider the system illustrated below



Figure 1

We can analyze this system via the equations

$$v[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2]$$
(1)

^{*}Version 1.2: Jul 19, 2010 2:47 pm -0500

[†]http://creativecommons.org/licenses/by/3.0/

 and

$$y[n] = v[n] + a_1 y[n-1] + a_2 y[n-2].$$
(2)

More generally,

$$v[n] = \sum_{k=0}^{N} b_k x[n-k]$$
 (3)

 and

$$y[n] = \sum_{k=1}^{M} a_k y[n-k] + v[n]$$
(4)

or equivalently

$$\sum_{k=0}^{N} b_k x \left[n - k \right] = y \left[n \right] - \sum_{k=1}^{M} a_k y \left[n - k \right].$$
(5)

In general, many LSI systems satisfy linear difference equations of the form:

$$\sum_{k=0}^{M} a_k y \left[n - k \right] = \sum_{k=0}^{N} b_k x \left[n - k \right].$$
(6)

What does the z-transform of this relationship look like?

$$Z\{\sum_{k=0}^{M} a_{k} y [n-k]\} = Z\{\sum_{k=0}^{M} b_{k} x [n-k]\}$$

$$\sum_{k=0}^{M} a_{k} Z\{y [n-k]\} = \sum_{k=0}^{N} b_{k} Z\{x [n-k]\}.$$
(7)

Note that

$$Z\{y[n-k]\} = \sum_{\substack{n=-\infty \\ \infty}}^{\infty} y[n-k] z^{-n} = \sum_{\substack{m=-\infty \\ m=-\infty}}^{\infty} y[m] z^{-m} \cdot z^{-k} = Y(z) z^{-k}.$$
(8)

Thus the relationship above reduces to

$$\sum_{k=0}^{M} a_{k} Y(z) z^{-k} = \sum_{k=0}^{N} b_{k} X(z) z^{-k}
Y(z) \left(\sum_{k=0}^{M} a_{k} z^{-k} \right) = X(z) \left(\sum_{k=0}^{N} b_{k} z^{-k} \right)
\frac{Y(z)}{X(z)} = \frac{\left(\sum_{k=0}^{N} b_{k} z^{-k} \right)}{\left(\sum_{k=0}^{M} a_{k} z^{-k} \right)}$$
(9)

Hence, given a system like the one above, we can pretty much immediately write down the system's transfer function, and we end up with a rational function, i.e., a ratio of two polynomials in z. Similarly, given a rational function, it is easy to realize this function in a simple hardware architecture. We will focus exclusively on such rational functions in this course.