ECE 3640
Lecture 2 – Z Transforms

Objective: Z-transforms are to difference equations what Laplace transforms are to differential equations. In this lecture we are introduced to Z-transforms, their inverses, and their properties.

We will also solve difference equations using transform techniques.

Introduction to the Z-transform

As you recall, we talked first about differential equations, then difference equations. The methods of solving difference equations was in very many respects parallel to the methods used to solve differential equations. We then learned about the Laplace transform, which is a useful tool for solving differential equations and for doing system analysis on continuous-time systems. Our development now continues to the Z-transform. This is a transform technique used for discrete time signals and systems. As you might expect, many of the tools and techniques that we developed using Laplace transforms will transfer over to the Z-transform techniques.

The Z-transform is simply a power series representation of a discrete-time sequence. For example, if we have the sequence $x[0], x[1], x[2], x[3]$, the Z-transform simply multiplies each coefficient in the sequence by a power of z corresponding to its index. In this example

$$X(z) = x[0] + x[1]z^{-1} + x[2]z^{-2} + x[3]z^{-3}.$$

Note that negative powers of z are used for positive time indexes. This is by convention. (Comment.)

For a general causal sequence $f[k]$, the Z-transform is written as

$$F[z] = \sum_{k=0}^{\infty} f[k]z^{-k}.$$

For a general (not necessarily noncausal) sequence $f[k]$,

$$F[z] = \sum_{k=-\infty}^{\infty} f[k]z^{-k}.$$

As for continuous time systems, we will be most interested in causal signals and systems.

The inverse Z-transform has the rather strange and frightening form

$$f[k] = \frac{1}{2\pi j}\oint F[z]z^{k-1}dz$$

where $\oint$ is the integral around a closed path in the complex plane, in the region of integration. (Fortunately, this is rarely done by hand, but there is some very neat theory associated with integrals around closed contours in the complex plane. If you want the complete scoop on this, you should take complex analysis.)

Notationally we will write

$$F[z] = \mathcal{Z}\{f[k]\} \qquad f[k] = \mathcal{Z}^{-1}\{F[z]\}$$

or

$$f[k] \leftrightarrow F[z]$$

Example 1 Find the Z-transform of $f[k] = \gamma^k u[k]$.

$$F[z] = \sum_{k=0}^{\infty} \gamma^k z^{-k} = \sum_{k=0}^{\infty} (\gamma/z)^k = \frac{1}{1-\gamma/z}$$

(Recall the formula for the sum of an infinite series. Remember it on your deathbed!) When does this converge?

$$F[z] = \frac{z}{z-\gamma}$$

The ROC is $|z| > |\gamma|$. (Compare with ROC for causal Laplace transform.) □

Example 2 Find the following Z-transforms.

1. $f[k] = \delta[k]$. $F[z] = 1$. ROC: all z.
2. $f[k] = u[k]$. $F[z] = \frac{z}{z-1}$. ROC: $|z| > 1$.
3. $f[k] = \cos(\beta k)$.

$$F[z] = \frac{1}{2}\left[\frac{z}{z-e^{j\beta}} + \frac{z}{z-e^{-j\beta}}\right] = \frac{z(z-\cos\beta)}{z^2 - 2z\cos\beta + 1}$$

□

Important and useful functions have naturally been transformed and put into tabular form.

Inverse Z-transforms

Given the nature of the inverse Z-transform integral, we look for other ways of computing the inverse. The approach is very similar to what we did for Laplace transforms: We break the function down into pieces that we can recognize from the table, then do table lookup. As for Laplace transforms, there will be a variety of properties (delay, convolution, scaling, etc.) that will help us. The most important tool, however, remains the Partial Fraction Expansion. However, we will find it convenient to modify it slightly for our purposes here.

Example 3 Find the inverse Z-transform of $F[z] = \frac{z}{z-.2}$.

Knowing what we know, it is straightforward to write

$$f[k] = (.2)^k u[k]$$

□

Example 4 Find the inverse Z-transform of

$$F[z] = \frac{8z-19}{(z-2)(z-3)}$$

In order to do this, we need to expand using PFE into something like

$$F[z] = \frac{k_1 z}{z-2} + \frac{k_2 z}{z-3}$$

because this is the form that we know about. But recall that the PFE we have come to know and love always just has constants in the numerator. So we will create a new function: Let

$$G[z] = \frac{F[z]}{z} = \frac{8z-19}{z(z-2)(z-3)}$$

Now do the PFE on $G[z]$:

$$\frac{F[z]}{z} = G[z] = \frac{k_1}{z} + \frac{k_2}{z-2} + \frac{k_3}{z-3}$$

Using CUPI,

$$k_1 = \frac{-19}{6} \qquad k_2 = \frac{3}{2} \qquad k_3 = 5/3$$

so we can write

$$G[z] = \frac{F[z]}{z} = \frac{-(19/6)}{z} + \frac{(3/2)}{z-2} + \frac{(5/3)}{z-3}.$$

Now we solve for $F[z]$:

$$F[z] = -\frac{19}{6} + \frac{(3/2)z}{z-2} + \frac{(5/3)z}{z-3}.$$

Note that by our little trick we have put this into the form we need. Now we can read off the inverse directly:

$$f[k] = -\frac{19}{6}\delta[k] + [\frac{3}{2}(2)^k + \frac{5}{3}(3)^k]u[k].$$

□

The point is: Computing inverse Z-transforms using PFE is exactly analogous to computing inverse Laplace transforms, **provided that you form $F[z]/z$ first.**

There is another method of obtaining inverse Z-transforms which is useful if you only need a few terms. Recall that the Z-transform is simply a power series. All you need to do is find the coefficients of the power series. One way to do this is by long polynomial division. This gives you a numerical expression for as many terms of $f[k]$ as you choose to compute, not a closed-form mathematical expression.

Example 5 If $F[z] = z/(z - .5)$, find $f[k]$. Work through the first few. □

Properties of Z-transforms

In the descriptions of these properties, take

$$f[k] \leftrightarrow F[z].$$

Delay property This is very analogous to the differentiation property of Laplace transforms, and will similarly allow us to solve differential equations.

$$\boxed{f[k-1]u[k-1] \leftrightarrow z^{-1}F[z]}$$

So z^{-1} is the delay operator. (As s is the differentiation operator.) Also

$$\boxed{f[k-1]u[k] \leftrightarrow z^{-1}F[z] + f[-1].}$$

Note the difference between these two!

This property is used to introduce the initial conditions when we use transforms to solve difference equations.

Proof For the more general case of shifting by m,

$$\mathcal{Z}\{f[k-m]u[k-m]\} = \sum_{k=0}^{\infty} f[k-m]u[k-m]z^{-k} = \sum_{k=m}^{\infty} f[k-m]z^{-k}$$

Now make a a change of variable: let $r = k - m$, so that $k = r + m$.

$$\mathcal{Z}\{f[k-m]u[k-m]\} = \sum_{r=0}^{\infty} f[r]z^{-(r+m)} = z^{-m}F[z].$$

For the other case,

$$\begin{aligned}\mathcal{Z}\{f[k-m]u[k\} &= \sum_{k=0}^{\infty} f[k-m]z^{-k} = \sum_{r=-m}^{\infty} f[r]z^{-(r+m)} \\ &= z^{-m}\left[\sum_{r=-m}^{-1} f[r]z^{-r} + \sum_{r=0}^{\infty} f[k]z^{-r}\right] \\ &= z^{-m}\sum_{k=1}^{m} f[-k]z^{k} + z^{-m}F[z]\end{aligned}$$

So

$$\boxed{f[k-1] \leftrightarrow z^{-1}F[z] + f[-1]}$$

$$\boxed{f[k-2] \leftrightarrow z^{-2}F[z] + z^{-1}f[-1] + f[-2]}$$

$$\boxed{f[k-3] \leftrightarrow z^{-3}F[z] + z^{-2}f[-1] + z^{-1}f[-2] + f[-3]}$$

□

Left Shift (Advance) Similar to the last property,

$$\boxed{f[k+m]u[k] \leftrightarrow z^{m}F[z] - z^{m}\sum_{k=0}^{m-1} f[k]z^{-k}}$$

Example 6 Find the Z-transform of the sequence

$$f[k] = k\{u[k] - u[k-6]\}$$

(Plot this). Write this as

$$f[k] = ku[k] - ku[k-6] = ku[k] - ((k-6)u[k-6] + 6u[k-6])$$

Then

$$ku[k] \leftrightarrow \frac{z}{(z-1)^2}$$

and

$$u[k-6] \leftrightarrow z^{-6}\frac{z}{z-1}$$

so

$$(k-6)u[k-6] \leftrightarrow z^{-6}\frac{z}{(z-1)^2}$$

Combining,

$$F[z] = \frac{z}{(z-1)^2} + z^{-6}\frac{z}{(z-1)^2} + 6z^{-6}\frac{z}{z-1} = \frac{z^6 - 6z + 5}{z^5(z-1)^2}$$

□

Convolution Like the convolution property for Laplace transforms, the convolution property for Z-transforms is very important for systems analysis and design. In words: The transform of the convolution is the product of the transforms. This holds for both Laplace and Z-transforms.

If $f_1[k] \leftrightarrow F_1[z]$ and $f_2[k] \leftrightarrow F_2[z]$ then

$$f_1[k] * f_2[k] \leftrightarrow F_1[z]F_2[z]$$

where $*$ denotes convolution (in this case, discrete-time convolution).

Proof This is somewhat easier (and more general) to prove for noncausal sequences.

$$\begin{aligned}
\mathcal{Z}[f_1[k] * f_2[k]] &= \mathcal{Z}\left[\sum_{m=-\infty}^{\infty} f_1[m]f_2[k-m]\right] \\
&= \sum_{k=-\infty}^{\infty}\sum_{m=-\infty}^{\infty} f_1[m]f_2[k-m]z^{-k} \\
&= \sum_{m=-\infty}^{\infty} f_1[m] \sum_{k=-\infty}^{\infty} f_2[k-m]z^{-k} \\
&= \sum_{m=-\infty}^{\infty} f_1[m] \sum_{r=-\infty}^{\infty} f_2[r]z^{-(r+m)} \\
&= \sum_{m=-\infty}^{\infty} f_1[m]z^{-m} \sum_{r=-\infty}^{\infty} f_2[r]z^{-r} = F_1[z]F_2[z].
\end{aligned}$$

$\square$

Multiplication by γ^k

$$\gamma^k f[k]u[k] \leftrightarrow F[z/\gamma]$$

Multiplication by k

$$kf[k]u[k] \leftrightarrow -z\frac{d}{dz}F[z]$$

Initial Value theorem For a causal $f[k]$,

$$f[0] = \lim_{z\to\infty} F[z]$$

Final Value theorem If $F[z]$ has no poles outside the unit circle (i.e. it is stable),

$$\lim_{n\to\infty} f[n] = \lim_{z\to 1}(z-1)F(z).$$

Solution of difference equations

Example 7 Solve

$$y[k+2] - 5y[k+1] + 6y[k] = 3f[k+1] + 5f[k]$$

with $y[-1] = \frac{11}{6}$ and $y[-2] = \frac{37}{36}$ and input $f[k] = 2^{-k}u[k]$. First, shift the equation so that we can take advantage of the form of the initial conditions. We replace $k \to k-2$ to obtain

$$y[k] - 5y[k-1] + 6y[k-2] = 3f[k-1] + 5f[k-2]$$

Now take the Z-transform of each part.

$$y[k] \Leftrightarrow Y[z]$$

We are interested in what happens from $k = 0$ and onward, so $y[k-1]$ is to be interpreted as $y[k-1]u[k]$, and not $y[k-1]u[k-1]$. In this way we introduce the initial condition information:

$$y[k-1]u[k] \Leftrightarrow z^{-1}Y[z] + y[-1] = z^{-1}Y[z] + \frac{11}{6}.$$

$$y[k-2]u[k] \Leftrightarrow z^{-2}Y[z] + z^{-1}y[-1] + y[-2] = z^{-2}Y[z] + z^{-1}\frac{11}{6} + \frac{37}{36}.$$

Also take the Z-transform of the input sequence:

$$f[k] \Leftrightarrow \frac{z}{z-0.5}$$

For delayed versions of the input function

$$f[k-r] \Leftrightarrow z^{-r}\frac{z}{z-0.5}$$

since the function is causal.

Now combine all of the pieces into the difference equation:

$$Y[z] - 5[z^{-1}Y[z] + \frac{11}{6}] + 6[z^{-1}Y[z] + z^{-1}\frac{11}{6} + \frac{37}{36}] = \frac{3}{z-.5} + \frac{5}{z(z-.5)}$$

Combining terms

$$\left(1 - \frac{5}{z} + \frac{6}{z^2}\right)Y[z] - \left(3 - \frac{11}{6}\right) = \frac{3}{z-.5} + \frac{5}{z(z-.5)}$$

Identify portions due to input and initial conditions.

$$\left(1 - \frac{5}{z} + \frac{6}{z^2}\right)Y[z] = \frac{3z^2 - 9.5z + 10.5}{z(z-.5)}$$

Multiply by z^2:

$$(z^2 - 5z + 6)Y[z] = \frac{z(3z^2 - 9.5z + 10.5)}{z-.5}$$

Solve for $Y[z]$:

$$Y[z] = \frac{z(3z^2 - 9.5z + 10.5)}{(z-.5)(z^2 - 5z + 6)}$$

At this point, finding the solution is done by PFE. Remember to divide by z:

$$\frac{Y[z]}{z} = \frac{(3z^2 - 9.5z + 10.5)}{(z-.5)(z^2 - 5z + 6)} = \frac{k_1}{z-.5} + \frac{k_2}{z-2} + \frac{k_3}{z-3}$$

Using CUPI we find that

$$\frac{Y[z]}{z} = \frac{(26/15)}{z-.5} - \frac{(7/3)}{z-2} + \frac{(18/5)}{z-3}.$$

We can now solve for $y[n]$:

$$y[n] = [\frac{26}{15}(.5)^k - \frac{7}{3}2^k + \frac{18}{5}3^k]u[k]$$

□

Note that by keeping the portions due to the initial conditions separate from the portions due to the input we can find both the zero-state response and the zero-input response by this method.

Transfer Functions

Under the assumption of zero initial conditions (the zero-state response) the general LTI difference equation

$$Q[E]y[k] = P[e]f[k]$$

$$(E^n + a_{n-1}E^{n-1} + \cdots + a_1 E + a_0)y[k] = (b_n E^n + b_{n-1}E^{n-1} + \cdots + b_1 E + b_0)f[k]$$

may be transformed to

$$(z^n + a_{n-1}z^{n-1} + \cdots a_1 z + a_0)Y[z] = (b_n z^n + b_{n-1}z^{n-1} + \cdots b_1 z + b_0)F[z]$$

Solving for the output,

$$Y[z] = \left(\frac{b_n z^n + b_{n-1}z^{n-1} + \cdots b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + \cdots a_1 z + a_0}\right) F[z]$$

We define

$$H[z] = \frac{z^n + b_{n-1}z^{n-1} + \cdots b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + \cdots a_1 z + a_0} = \frac{P[z]}{Q[z]}$$

as the **transfer function**. Note that

$$H[z] = \frac{Y[z]}{F[z]} = \frac{\mathcal{Z}[\text{zero-state response}]}{\mathcal{Z}[\text{input}]}$$

and the output is obtained by

$$Y[z] = F[z]H[z].$$

The poles of the transfer function are the roots of the characteristic equation, and we can determine the stability of the system by examination of the transfer function.

Example 8 Unit delay: $y[k] = f[k-1]u[k-1]$. $Y[z] = \frac{1}{z}F[z]$. $H[z] = z^{-1}$. (Remember this!) □

Example 9 Find the transfer function for the system

$$y[k+2] + y[k+1] + .16y[k] = f[k+1] + .32f[k]$$

In operator notation

$$(E^2 + E + .16)y[k] = (E + .32)f[k]$$

$$H[z] = \frac{z + .32}{z^2 + z + .16}$$

Now, if $f[k] = (-2)^{-k}u[k]$ find the zero-state response (all initial conditions zero).

$$F[z] = \frac{z}{z + .5}$$

$$Y[z] = F[z]H[z] = \frac{z(z + .32)}{(z^2 + z + .16)(z + .5)}$$

Don't forget to pull over z before the PFE:

$$\frac{Y[z]}{z} = \frac{(z + .32)}{(z^2 + z + .16)(z + .5)} = \frac{2/3}{z + .2} - \frac{8/3}{z + .8} + \frac{2}{z + .5}$$

$$y[k] = [\frac{2}{3}(-.2)^k - \frac{8}{3}(-.8)^k + 2(-.5)^k]u[k]$$

□

If the input is $f[k] = \delta[k]$, then the output is

$$Y[z] = H[z]F[z] = H[z].$$

So

$$h[k] \Leftrightarrow H[z].$$

The transfer function is the Z-transform of the impulse response.
Nomenclature. A discrete-time filter which has only a numerator part (only zeros, except for possible poles at the origin which correspond to delays) is said to be a **finite impulse response** (FIR) filter.
Example 10 What is the impulse response of a filter with

$$H[z] = 1 + 2z^{-1} + 3z^{-3}$$

Note that **all FIR filters are stable**. □

A filter with poles is said to be an **infinite impulse response** (IIR) filter.
Example 11 What is the impulse response of a filter with

$$H[z] = \frac{z}{z - .5}.$$

□

Note that there is no practical way of making an FIR filter for continuous time systems: this is available only for digital filters.

System Realization

All of the block diagram operations we talked about with respect to Laplace transforms still apply. Series (cascade), parallel, and feedback configurations all have the same block diagram simplifications.

The same techniques we used for "system realization" of Laplace transforms still apply for Z-transforms. The only difference is to substitute z^{-1} for $\frac{1}{s}$. Even though the diagram ends up the same, there may be understanding to be gained by working through the steps. As before, we will take an example of a third order transfer function

$$H[z] = \frac{b_3z^3 + b_2z^2 + b_1z + b_0}{z^3 + a_2z^2 + a_1a + a_0}$$

Break this into two pieces: Let

$$X[z] = \frac{1}{z^3 + a_2z^2 + a_1a + a_0}F[z]$$

and let

$$Y[z] = X[z](b_3z^3 + b_2z^2 + b_1z + b_0)$$

From $X[z]$ we can write

$$z^3X[z] + a_2z^2X[z] + a_1zX[z] + z_0X[z] = F[z]$$

$$z^3X[z] = -a_2z^2X[z] - a_1zX[z] - z_0X[z] + F[z]$$

Draw a block diagram. Then we connect the rest of the pieces to get $Y[z]$.

Example 12 Draw a system realization for

$$H[z] = \frac{z+2}{z^2+5z+1}$$

□

Note that in FIR filters the output depends only on the current and previous inputs. In IIR filters, the output depends on these and also on *prior outputs* — there is some kind of feedback. It is this feedback that gives them their infinite response.

This realization is useful in a sort of theoretical sense, and gives us a map of what is going on. But, unlike for continuous-time systems, there is no practical way of doing this using resistors, capacitors, and op-amps. What the diagram really represents is a **computer program**. We will now talk about how this is done. We will start first with an FIR filter. Consider the particular example of

$$H[z] = 2 + 3z^{-1} + 4z^{-2} + 5z^{-3}$$

Then

$$Y[z] = H[z]F[z] = (2 + 3z^{-1} + 4z^{-2} + 5z^{-3})F[z].$$

Transforming back,

$$y[k] = 2f[k] + 3f[k-1] + 4f[k-2] + 5f[k-3]$$

Draw the block diagram. The equation tells us how to program it. First, we need some way of keeping track of the previous values. Let is keep these in an array called `fprevious`, and set it up so that `fprevious[0]` is the current value of f, `fprevious[1]` is the last value of f, and so on. Furthermore, let is keep track of the coefficients in an array called `coefficient`, set up as follows:

```
coefficient[0] = 2;
coefficient[1] = 3;
coefficient[2] = 4;
coefficient[3] = 5;
```

Now we will create a filtering routine, and pass the coefficients into it. **Note: this code is provided for demonstration purposes only. It may have minor problems with it that the student is expected to be able to understand and correct.**

```
/* fir filter, version 1 */
double firfilt(double f, double *coefficient)
{
   static double fprevious[4];
   double sum;

   fprevious[0] = f;   /* assign the current input */

   /* compute the filter output */
   sum = fprevious[0]*coefficient[0] + fprevious[1]*coefficient[1] +
            fprevious[2]*coefficient[2] + fprevious[3]*coefficient[3];

   /* now shift everything down */
   fprevious[3] = fprevious[2];
   fprevious[2] = fprevious[1];
   fprevious[1] = fprevious[0];

   return sum;
}
```

This computes the output and shifts everything down. Note that we have to save the previous values of the outputs so they can be used for the next call to the filter. This has problems — suppose you have more than one filter — how do you keep things from getting messed up. Perhaps a cleaner way to do this would be to pass in the previous values to the filter. The cleanest way is probably to use C++ with a constructor that keeps a separate data set for each instantiation of the filter class. I will leave these finessings to the diligent student.

To generalize our simple filter routine, let us allow different numbers of coefficients to be passed in. This means that we have to allocate sufficient space for the previous values, and add everything up in a loop.

```
#include <stdlib.h>    /* put this at the top so calloc is used right */

.
.
.

double firfilt(double f, double *coefficient,int numcoef)   /* version2 */
{
   static double *fprevious = NULL;
   double sum;
   int i;

   if(fprevious == NULL) { /* first time in allocate enough space */
      fprevious = (double *)calloc(numcoef,sizeof(double));
   }

   fprevious[0] = f;   /* assign the current input */

   sum = 0;
   /* do the filter operations */
   for(i = 0; i < numcoef; i++) {
      sum += fprevious[i]*coefficient[i];
   }

   /* now shift everything down */
   for(i = numcoef-1; i > 0; i--) {
      fprevious[i] = fprevious[i-1];
   }
   return sum;
}
```

For the diligent students interested in speeding things up as much as possible, I pose the following ideas:

1. Can the filter loop and the shift loop be combined, so that only one loop needs to be execute to accomplish both functions?
2. The shifting operation is slow and unnecessary. How could you use a circular queue to store the previous values so that the shifting operation is no longer necessary?

Enough about FIR filters. Implementing IIR filters will also be addressed by means of an example. We want to implement the filter represented by

$$H[z] = \frac{6z^2 + 2z + 3}{z^2 + 4z + 5}$$

$$Y[z] = H[z]F[z]$$
$$(z^2 + 4z + 5)Y[z] = (6z^2 + 2z + 3)F[z]$$

In the time domain,

$$y[k+2] + 4y[k+1] + 5y[k] = 6f[k+2] + 2f[k+1] + 3f[k]$$

Shifting in time and solving for $y[k]$,

$$y[k] = -4y[k-1] - 5y[k-2] + 6f[k] + 2f[k-1] + 3f[k-2]$$

Again, we have a formula for the filter output. Assume that the numerator coefficients are stored in an array `numcoeff` and the denominator coefficients are stored in an array `dencoeff`:

```
numcoeff[0] = 6;
numcoeff[1] = 2;               dencoeff[1] = -4;
numcoeff[2] = 3;               dencoeff[2] = -5;
```

Caution: note that the denominator coefficients are the negative of the coefficients in the original transfer function. We will keep the previous input values in an array `fprevious` and keep previous output values in an array `yprevious`.

```
double iirfilt(double f, double *numcoeff, double *dencoeff)   /* version 1*/
{
   static double fprevious[3];
   static double yprevious[3];
   double y;

   fprevious[0] = f;   /* assign the current input */

   /* compute the filter output */
   y = fprevious[0]*numcoeff[0];  /* get it started */
   y += fprevious[1]*numcoeff[1] + fprevious[2]*numcoeff[2] +
            yprevious[1]*dencoeff[1] + yprevious[2]*dencoeff[2];

   /* now shift everything down */
   fprevious[2] = fprevious[1];
   fprevious[1] = fprevious[0];

   yprevious[2] = yprevious[1];
   yprevious[1] = y;       /* the output */

   return y;
}
```

As before, we will generalize this to arbitrary transfer functions of denominator degree `degree`:

```
/* version 2*/
double iirfilt(double f, double *numcoeff, double *dencoeff, int degree)
{
   static double *fprevious = NULL;
   static double *yprevious = NULL;
   double y;
   int i;

```

```
    if(fprevious == NULL) {   /* first time set up space */
       fprevious = (double )calloc(degree+1,sizeof(double));
       yprevious = (double )calloc(degree+1,sizeof(double));
    }

    fprevious[0] = f;   /* assign the current input */

    /* compute the filter output */
    y = fprevious[0]*numcoeff[0];  /* get it started */
    for(i = 1; i <= degree; i++) {
       y += fprevious[i]*numcoeff[i];
    }
    yprevious[0] = y;

    /* now shift everything down */
    for(i = degree; i > 0; i--) {
       fprevious[i] = fprevious[i-1];
       yprevious[i] = yprevious[i-1];
    }

    return y;
 }
```

Again, speedups are attainable: merge the shift loop into the filter loop, or get rid of shifting entirely by using a circular queue.

Bilateral Z-transform

In the most general case, we have

$$F(z) = \sum_{k=-\infty}^{\infty} f[k] z^{-k}$$

Let us consider the z transform of $f[k] = -\gamma^k u[-(k+1)]$ (draw the picture). We find

$$F(z) = \frac{z}{z-\gamma}.$$

Compare with $g[k] = \gamma^k u[k]$. What gives? Must specify region of convergence for these.

Frequency response

Continuous time with transfer function $H(s)$: $e^{j\omega t} \rightarrow H(j\omega)e^{j\omega t}$. An analogous result holds for discrete time systems.

Let the input to a discrete-time system be $f[k] = z^k$ (everlasting, so we don't have to worry about transients). Then

$$y[k] = h[k] * z^k = z^k \sum_m h[m] z^{-m} = z^k H[z].$$

More particularly, consider when $z = e^{\pm j\Omega}$. We find that

$$e^{j\Omega k} \rightarrow H(e^{j\Omega}) e^{j\Omega k}$$

$$e^{-j\Omega k} \to H(e^{-j\Omega})e^{-j\Omega k}$$

Adding:

$$\cos \Omega k \to \mathrm{Re}(H(e^{j\Omega})e^{j\Omega k})$$

or, in polar form with $H(e^{j\Omega}) = |H(e^{j\Omega})|e^{j\arg H(e^{j\Omega})}$ we find

$$\cos \Omega k \to |H(e^{j\Omega})| \cos(\Omega k + \arg H(e^{j\Omega k})).$$

That is, the cosine is modified in amplitude and phase by the transfer function.

Example 13 For the system $y[k+1] - 0.8y[k] = f[k+1]$, determine the frequency response. We have

$$H(z) = \frac{z}{z-0.8} = \frac{1}{1-0.8z^{-1}}$$

$$H(e^{j\Omega}) = \frac{1}{1-0.8e^{-j\Omega}} = \frac{1}{(1-0.8\cos\Omega) + j0.8\sin\Omega}$$

Then

$$|H(e^{j\Omega})| = \frac{1}{\sqrt{(1-0.8\cos\Omega)^{+}(0.8\sin\Omega)^2}}$$

and

$$\arg H(e^{j\Omega}) = -\tan^{-1}\frac{0.8\sin\Omega}{1-0.8\cos\Omega}.$$

When the input is $f[k] = 1$, determine the output. What about when $f[k] = \cos(\pi/6k - 0.2)$? □

Note that $H(e^{j\Omega})$ is periodic.

Frequency response from pole-zero plot: Rubber sheet geometry

Let us write $H(z)$ in terms of its poles and zeros:

$$H(z) = b_n \frac{(z-z_1)(z-z_2)\cdots(z-z_n)}{(z-\gamma_1)(z-\gamma_2)\cdots(z-\gamma_n)}$$

Consider evaluating this at a point $z = e^{j\Omega}$, which is on the unit circle. We find

$$|H(e^{j\Omega})| = |b_n| \frac{|e^{j\Omega} - z_1| \cdots |e^{j\Omega} - z_1|}{|e^{j\Omega} - \gamma_1| \cdots |e^{j\Omega} - \gamma_1|}$$

Let us write $e^{j\Omega} - z_i = r_i e^{j\phi_i}$ (polar form for the line segment connecting them), and $e^{j\Omega} - z_i = d_i e^{j\theta_i}$ (polar form). Then

$$|H(e^{j\Omega})| = |b_n| \frac{r_1 r_2 \cdots r_n}{d_1 d_2 \cdots d_n} = |b_n| \frac{\text{product of distances from zeros to } e^{j\Omega}}{\text{product of distances from poles to } e^{j\Omega}}$$

Similarly,

$$\arg H(e^{j\Omega}) = (\phi_1 + \cdots \phi_n) - (\theta_1 + \cdots + \theta_n) = \text{sum of zero angles to } e^{j\Omega} - \text{sum of pole angles to } e^{j\Omega}.$$

Discuss filter design by pole placement, and the rubber sheet idea: poles increase the gain, zeros decrease it. Notch filter. Overhead.

Example 14 Design using "trial and error" techniques a digital bandpass filter which passes at $\omega = 250\pi$ rad/sec and has zero transmission at $\omega = 0$ and $\omega = 1000\pi$. The highest frequency in the system is $f = 400$ Hz.

To must sample at more than twice the highest frequency: $f_s > 2f = 800$ Hz. We will take $f_s = 1000$ samples/second, so $T = 1/f_s = 1/1000$. In order to get zero transmission at the specified frequencies we must place zeros at $e^{j0T} = 1$ and $e^{j1000\pi T} = -1$. (Draw the zeros.) To the the response to peak up at $\omega = 250\pi$ we want to put a pole near it on the unit circle, $e^{j250\pi T} = e^{j\pi/4}$, and also at the conjugate location. Specifically, we will put the poles at

$$p_1 = \gamma e^{j\pi/4} \qquad p_2 = \gamma e^{-j\pi/4}.$$

where $\gamma < 1$ to ensure the poles are inside the unit circle. What is the effect of γ on the response? The transfer function is

$$H(z) = K\frac{(z-1)(z+1)}{(z-\gamma e^{j\pi/4})(z-\gamma e^{-j\pi/4})} = K\frac{z^2-1}{z^2-\sqrt{2}\gamma z+\gamma^2}.$$

□

Example 15 We want to design a second-order notch filter to have zero transmission at 250 Hz and a sharp recovery on each side of the notch. The highest frequency in the system is $f = 500$ Hz. Take $f_s = 1000$ Hz, to $T = 1/1000$. The notch frequency is $\omega T = 2\pi(250)T = \pi/2$. We need a zero there. To get the recovery gain, we need a pole nearby. Let up place the pole at $\pm ja$ with $a < 1$ for stability. The transfer function is

$$H(z) = K\frac{(z-j)(z+j)}{(z-ja)(z+ja)} = K\frac{z^2+1}{z^2+a^2}.$$

Let us choose K to get unity DC gain:

$$H(1) = K\frac{2}{1+a^2}$$

so

$$K = (1+a^2)/2.$$

□

Linear phase FIR filters

We have mentioned several times that FIR filters can have linear phase. Now we will show why. Suppose that $h[k] = h[n-k]$ (the coefficients are symmetric. In particular, consider the example

$$h[k] = h[0]\delta[k] + h[1]\delta[k-1] + h[2]\delta[k-2] + h[3]\delta[k-3] + h[4]\delta[k-4] + h[5]\delta[k-5]$$

then

$$\begin{aligned} H(e^{j\Omega}) &= h[0] + h[1]e^{-j\Omega} + h[2]e^{-j2\Omega} + h[3]e^{-j3\Omega} + h[4]e^{-j4\Omega} + h[5]e^{-j5\Omega} \\ &= e^{-j(5/2)\Omega}\left(h[0]e^{j(5/2)\Omega} + h[1]e^{j(3/2)\Omega} + h[2]e^{j(1/2)\Omega} + h[3]e^{-j(1/2)\Omega} + h[4]e^{-j(3/2)\Omega} + h[5]e^{-j(5/2)\Omega}\right) \\ &= e^{-j(5/2)\Omega}\left(2h[0]\cos((5/2)\Omega) + 2h[1]\cos((3/2)\Omega) + 2h[2]\cos((1/2)\Omega)\right) \end{aligned}$$

Then

$$\arg H(e^{j\Omega}) = -5/2\Omega,$$

a linear function of phase.

We can pull a similar stunt with antisymmetry: $h[k] = -h[n-k]$.