The Cooper Union Department of Electrical Engineering
Prof. Fred L. Fontaine
ECE310 Digital Signal Processing
Problem Set IV Quantization
October 30, 2023

1. In this problem, we define "truncation" to mean simply discarding the LSBs on the right, without adjusting the other bits that are kept. Also, we assume rounding operations are performed first, followed by overflow operations. For example, first round up if necessary, then test for overflow.

   Consider the following two's complement fixed point values:

   00101101.0110100
   111110010.1001011

   Do not compute the numerical values. Just produce the two's complement codes as specified.

   (a) Produce the (5).(4) codes assuming: roundoff by rounding with two's complement overflow; and roundoff by truncation with saturation overflow.

   (b) Produce the (4).(3) codes assuming roundoff by rounding with two's complement overflow; and roundoff by truncation with saturation overflow.

2. For each of the following two's complement values, write the code with the fewest number of bits that can represent the same value exactly. Again, don't actually compute the values:

   0001011.011100
   1110000.10010

3. For the following two's complement code, compute the value by hand EFFICIENTLY!! MINIMAL WORK!! Show work. (The spacing is just to make it easy to read)

   11111 11111 11101 . 01100

4. Here you will explore the *sos* structure in MATLAB, and L^∞-scaling. First, design an 8^{th} order bandpass digital elliptic filter with $1.5dB$ ripple in the passband, $30dB$ ripple in the stopband, and passband from 0.3 to 0.6 on a scale where 1 is the Nyquist bandwidth. Specifically, obtain the *zpk* form (NOT the transfer function form). **Careful:** When you use the *ellip* function in MATLAB, check what value of n it needs!

   (a) Convert to *tf* form and obtain a graph of the magnitude response.

   (b) Convert the *zpk* form to *sos* form, with 'up' ordering, and again with 'down' ordering, with L^∞-scaling.

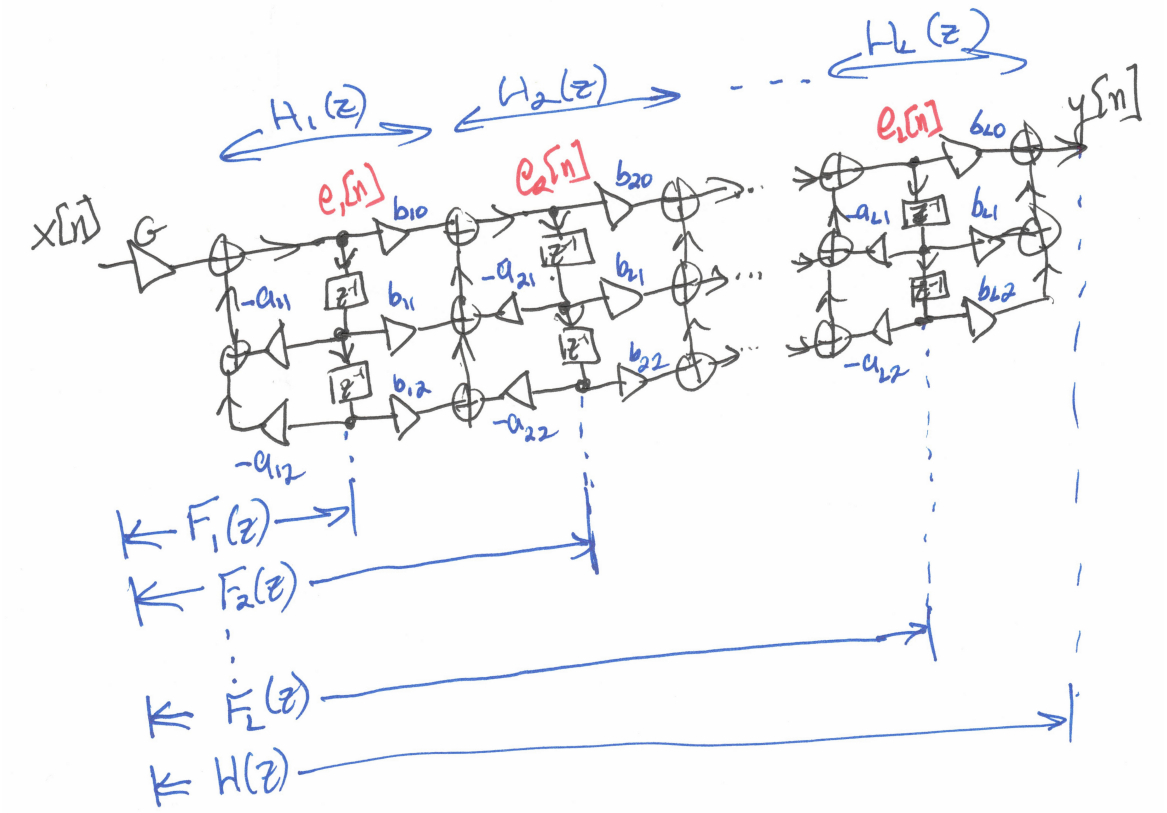
   (c) Compute the magnitudes of the poles of each of the stages of the *sos* realizations, as well as the magnitudes of the zeros, and check that they are indeed increasing (or decreasing) as they should be.

   (d) Do the following for the 'up' and 'down' cases. Let's call the successive stages $H_1,\ H_2,\ \cdots,\ H_L$, where H_1 is the first stage (where the input is applied). We have $H_i(z) = B_i(z)/A_i(z)$ where:

$$\begin{aligned} B_i(z) &= b_{i0} + b_{i1}z^{-1} + b_{i2}z^{-2} \\ A_i(z) &= 1 + a_{i1}z^{-1} + a_{i2}z^{-2} \end{aligned}$$

   The realization is a cascade of second order sections, with MATLAB interpreting it as each stage is direct form II. This is illustrated in the **Figure** below. You will note the transfer function from the input to the top of the first delay chain is $1/A_1(z)$, from the input to the top of the second delay chain is $B_1/(A_1A_2)$, then $B_1B_2/(A_1A_2A_3)$, and so forth. It is THESE transfer functions that are successively L^∞-scaled. This ensures the state-variables are scaled (each entry in the chain is a delayed version of the value on top, hence the dynamic ranges of the state variables are controlled properly). Superimpose magnitude response plots of these successive cumulative transfer functions (one graph for the 'up' case, another graph for the 'down' case). They should appear to get successively sharper and sharper, but in all cases the peak gain should remain 0dB (that is the L^∞-scaling in action).

   (e) If you look at the 'up' and 'down' realizations, the denominator polynomials should appear in reverse order. The numerator polynomials are also in reverse order but are scaled differently. For example, $B_{1,up}(z)$ and $B_{4,down}(z)$ should match up to a constant scaling factor. Check all this! The reversal of the orders requires rescaling (which is done cumulatively, not by individual stages).


H1(z)
H2(z)
HL(z)
x[n]
G
e1[n]
e2[n]
eL[n]
y[n]
b10
b11
b12
b20
b21
b22
bL0
bL1
bL2
-a11
-a12
-a21
-a22
-aL1
-aL2
F1(z)
F2(z)
FL(z)
H(z)

5. **Sensitivity Properties of Parallel Allpass Realizations**

When a filter is decomposed as the sum of two lower order filters, e.g., $H(z) = H_1(z) + H_2(z)$, it is called a parallel configuration; this contrasts with the cascade configuration which corresponds to $H(z) = H_1(z) H_2(z)$. It turns out that a large class of digital filters (including Butterworth, Chebyshev and elliptic filters) can be realized by a parallel pair of allpass filters. In general, the allpass filters will have complex coefficients, even if the overall filter transfer function has real coefficients. However, when the filter has certain symmetries, the allpass functions can have purely real coefficients, and that is the case you will explore here. It turns out using lossless building blocks in DSP structures, as in this case, can provide significant advantages in terms of quantization effects. Here you will explore the sensitivity advantages of parallel allpass realizations.

The case we take is:

- a third-order elliptic lowpass filter
- passband edge at $\omega_p = 0.3\pi$
- on a linear scale, the passband and stopband ripples have equal amplitude: $0.9 \leq |H(\omega)| \leq 1$ in the passband, and $|H(\omega)| \leq 0.1$ in the stopband (this is one of the symmetries referred to above)
- the passband ripple is $0.92dB$, and the stopband ripple is $20dB$

The original transfer function is:

$$H(z) = \frac{0.1336 + 0.0568z^{-1} + 0.0563z^{-2} + 0.1336z^{-3}}{1 - 1.5055z^{-1} + 1.2630z^{-2} - 0.3778z^{-3}} \tag{1}$$

It is expressed as the sum of two allpass functions as follows:

$$H(z) = \frac{1}{2}\left[\frac{-0.4954 + z^{-1}}{1 - 0.4954z^{-1}} + \frac{0.7626 - 1.0101z^{-1} + z^{-2}}{1 - 1.0101z^{-1} + 0.7626z^{-2}}\right] \tag{2}$$

We are going to quantize the coefficients, and compare the resulting transfer functions with the original ones. In reality, there are several "tricks" that can be employed (e.g., how to handle a coefficient close to 1 like 1.0101). Here we will take a fairly straightforward approach. One comment: the 1/2 in front of (2) can be implemented with a shift, so we don't consider it a multiplier coefficient.

Assume all coefficients are represented with b bit two's complement. You have to decide how many fractional bits versus integer bits to use; the goal is to have as much precision (i.e., fractional bits) as possible as long as none of the coefficients trigger overflow. Don't count pure +1 or -1: of course we don't use a multiplier to represent them! All the coefficients in (1) should have the same format, and all coefficients in each of the two allpass factors in (2) should have the same format, but you can use different formats in each of the two forms.

(a) Use the MATLAB function *fi* to compute fixed-point representations of coefficients for the case of $b = 5$ bits. Specifically, use the command:

$$y = fi\,(x, s, w, f)$$

and then $x0 = y.data$ returns the numeric value. Use *doc fi* to understand the syntax. You are doing this just to get the numeric values for your coefficients. From this point on, use these values as standard double precision to examine the resulting quantized transfer functions, say H_{Q0} for the original form, and H_{QA} for the parallel allpass form.

(b) By plugging in $z = \pm 1$, check that the ideal gain is 1 at $\omega = 0$ and 0 at $\omega = \pi$. Compute the actual gains for the filters with quantized coefficients, and report the error (in dB).

(c) Compute the frequency responses for all three filters for 10^4 points from 0 to π. Compute the maximum of $|H(\omega) - H_{Q0}(\omega)|$ and $|H(\omega) - H_{QA}(\omega)|$. Also, superimpose plots of the magnitude responses of the three filters, on a decibel scale; the vertical axis should go from $0dB$ (or higher to capture the peak gain of all filters, if any exceed $0dB$) down to $-40dB$. Make sure the curves are clearly labeled, e.g. with a legend.

(d) Numerically compute or comment on the following issues;

1. the maximum deviation of the filter gains from the gain of the infinite-precision filter in the passband
2. deviation from the equiripple characteristics (check for the local maxima and minima in the passband and stopband, and observe to what extent they are not uniform)
3. maximum gain in the stopband

(e) Superimpose graphs of the group delay in the passband (only) and comment on whether the parallel all-pass based realization shows relatively better sensitivity properties in this sense.

(f) Compute the poles and zeros of the original filter, and each of the reduced precision realizations (I mean for the whole H_{Q0} and H_{QA}, not for the individual allpass factors). Examine in particular if the poles have moved outside the unit circle. Also, the original filter had a zero exactly on the unit circle (at $z = -1$); where has this zero moved for each of the two reduced precision filters?