EEEN40130: Advanced Signal Processing

Prof. Le-Nam Tran

Conor Igoe

13360786

Fixed-point FIR Filter Design

To understand the particular challenge in designing quantized filters, we consider the simple question: in general, what happens to an FIR filter as we change the coefficients of the kernel? Understanding how arbitrary changes affect the filter may give an insight into how we should consider coefficient changes of a particular form, that of quantization.

We start answering this motivating question by taking a closer look at the defining equations of digital filters in equation (1):

$$y[n] = a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + ... + b_1 y[n-1] + b_2[n-2] + ... \tag{1}$$

Taking the z-transform of both sides of equation (1), and manipulating, we obtain the transfer function of our digital filter shown in equation (2), expressed in terms of negative powers of z:

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + ...}{1 - b_1 z^{-1} - b_2 z^{-2} - ...} \tag{2}$$

We note that, for an FIR filter, all feedback coefficients take on a value of zero, implying that the filter depends only on the input, and is not dependent on its past output. Thus, for an FIR filter we obtain a simplified equation for the transfer function of the digital filter, shown in equation (3):

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + ...}{1} \tag{3}$$

We now consider a simple filter to make progress towards understanding general FIR filters. Suppose we have a 2nd order FIR filter, with a simple 3 coefficient kernel. Such a filter has a transfer function given by equation (4):

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1} \tag{4}$$

We now ask: what are the zeros of this filter, and how do they relate to the coefficients in the kernel? Multiplying the numerator and denominator of the transfer function in equation (4) by z^2, we obtain the transfer function of the FIR filter in standard form, with positive powers of z in the denominator and numerator:

$$H(z) = \frac{a_0 z^2 + a_1 z^1 + a_2}{z^2} \tag{5}$$

We note that in doing so, we reveal that the FIR filter has 2 repeated poles at the origin of the z-plane. We also note that the numerator of equation (5) is a 2nd order polynomial, and therefore can be factorized as in equation (6):

$$H(z) = \frac{(z - z_1)(z - z_2)}{z^2} \tag{6}$$

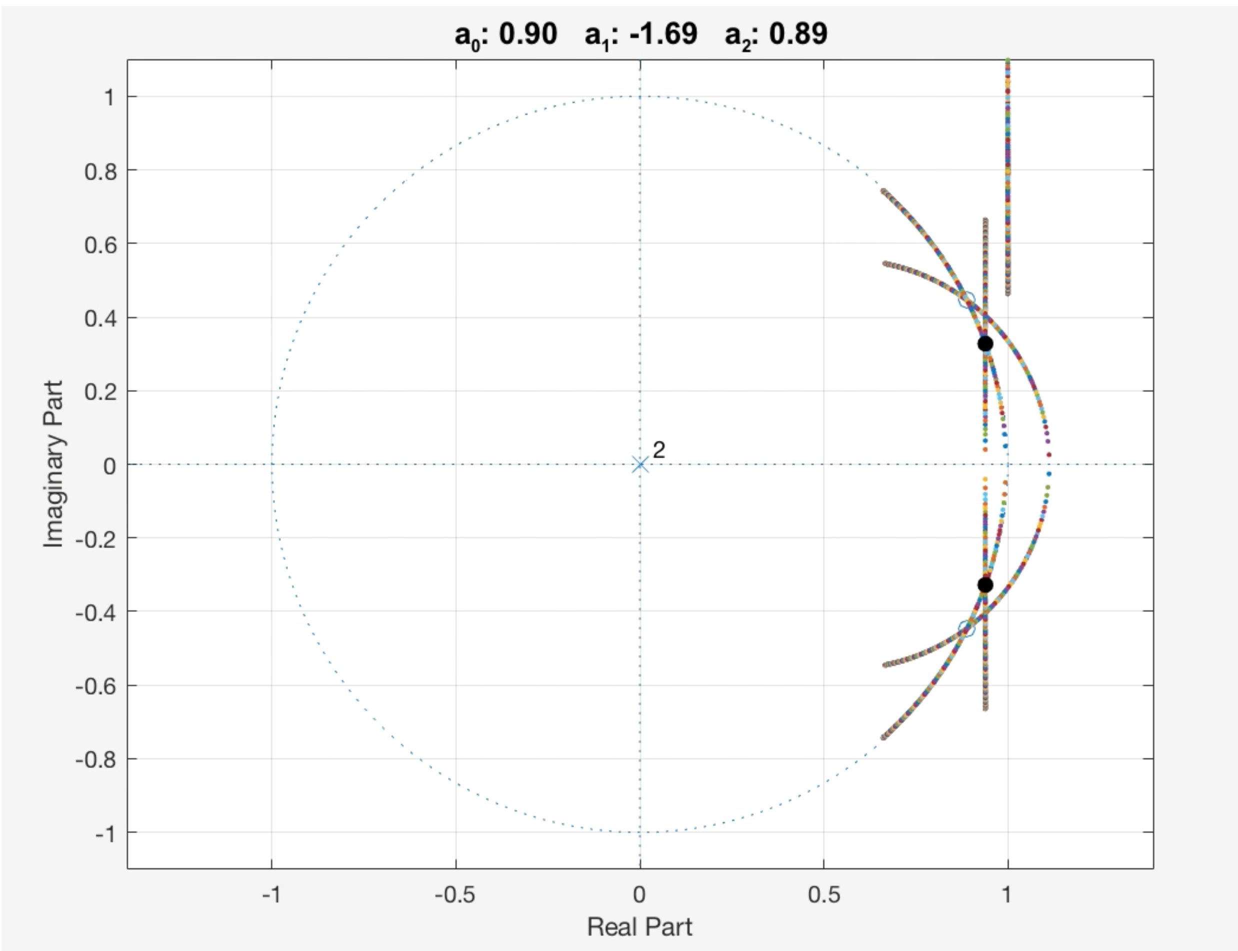
$$\text{where } z_1 = \frac{-a_1 + \sqrt{a_1^2 - 4a_0 a_2}}{2a_0},\ z_2 = \frac{-a_1 - \sqrt{a_1^2 - 4a_0 a_2}}{2a_0}$$

Equation (6) implies that, as well as two poles at the origin of the z-plane, our simple FIR filter also has 2 complex conjugate zeros. Importantly, ***both** zeros have a **non-linear** relationship with **all three** coefficients of the original kernel.* This observation will prove to be of significant importance shortly.

Our simple FIR filters poles and zeros in equation (6) completely define the topology of the z-space, with hills at pole locations and troughs at zeros, and a flat relief far away from both[1]. Moreover, this topology completely defines the frequency response: an "ant" crawling along the unit circle in the z-domain will rise and fall as it travels over the varying "terrain", commensurate with the magnitude of the frequency response at any frequency along the unit circle.

We are now in a position to explore how changes in the coefficients influences the position of the zeros, the z-plane topology, and ultimately the frequency response of the filter. Figure (1) shows a z-domain plot of the resulting zeros of our simple 2nd order FIR filter as the individual coefficients of the kernel are varied about their original values of $a_0, a_1 a_2 = [0.90, -1, 6, 0.89]$. [2]

Figure 1: Plot of zero location in z-domain for small changes in kernel coefficient values



[1]This intuition holds for causal, non-delayed kernels (where the number of origin-poles and zeros is matched). The intuition is slightly more complex for non-causal or delayed kernels.

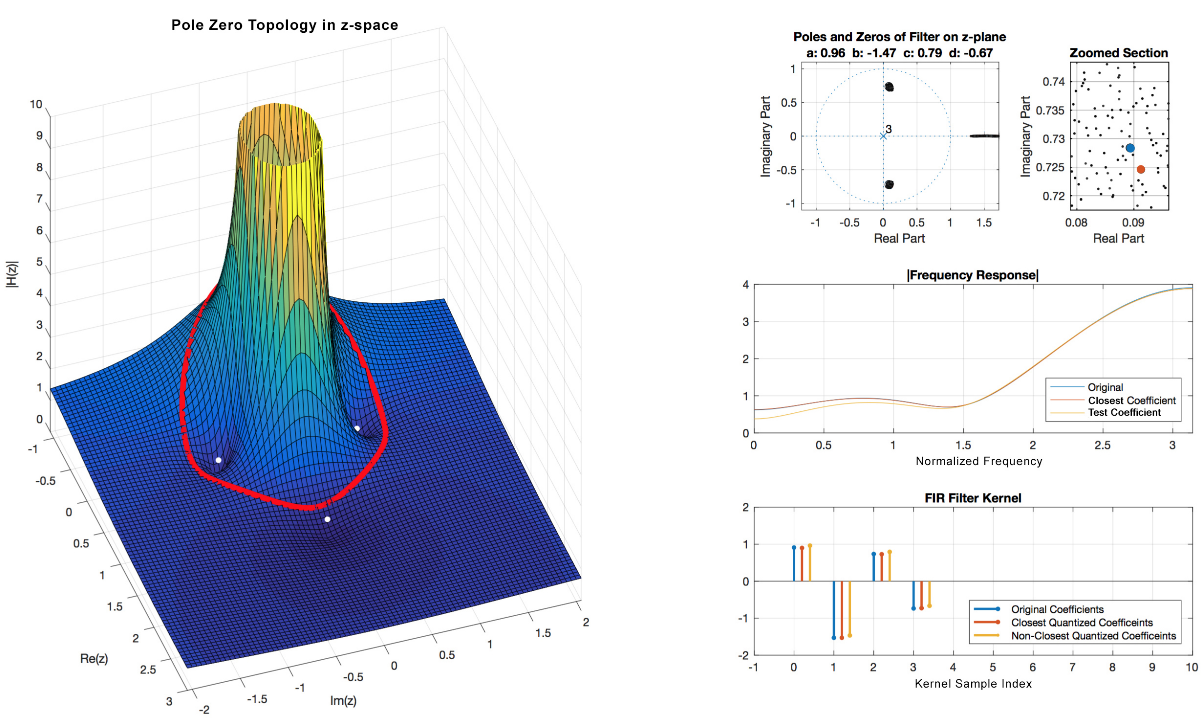
[2]A video animation of this variation is available in the link provided at the end of this report.

We note a few important observations:

- As expected, changing one coefficient results in changing the position of *all* zeros.
- Defined paths are traced out in the z-domain (the zero version of the root locus of our filter).
- Some parts of the z-domain are more sensitive to changes in the coefficients than other areas.

In the case of filter quantization, we are concerned with understanding how discrete changes in coefficient values varies the position of the zeros. Figure (2) shows the result of plotting the zeros for all 625 possible kernels that can arise from quantizing the kernel of a 3rd order FIR filter to coefficients that are up to ± 2 quantization steps from their closest quantization level. We start with plotting the zero positions for the original coefficients, the points marked in large blue in the upper right two plots of Figure (2). We then plot zeros that arise from using the coefficients that have values closest to their original values while being represented by a given level of bit resolution; these zeros are marked in large red in the same upper right two plots. Next, we consider the $5 \times 5 \times 5 \times 5$ kernels that arise from choosing quantization levels 2 steps below, 1 step below, unchanged, 1 step above and 2 steps above for each coefficient. Each of these zero locations is marked by a small black dot. Notably, we observe that setting the coefficients to their nearest quantized level does *not* correspond to a zero position that is the closest possible value to its original position. Specifically, other zero locations exist nearer to the original zero locations that can be represented by a set of quantized coefficients that is different to the closest quantized set. Although the zero locations that arise from using the closest quantized coefficients have the minimum "overall" distance from their original positions when *all* zeros are considered of equal importance, when designing for a particular spec, it is not in general true that we care equally about all zeros.

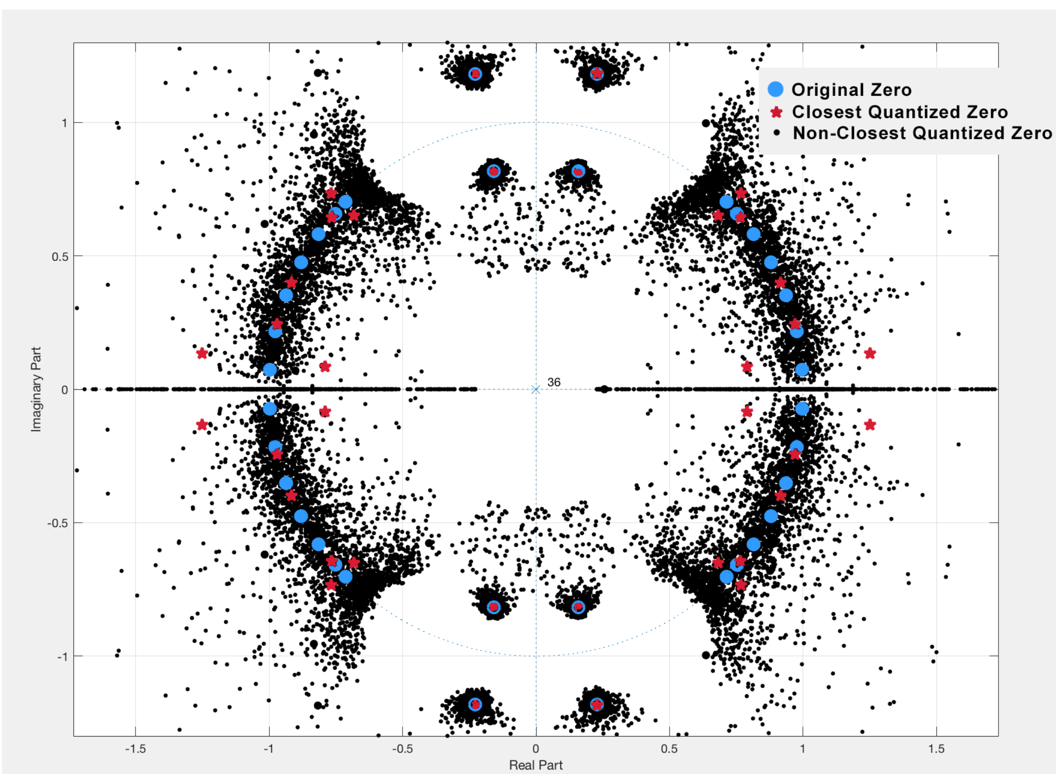
Figure 2: Illustration of the effect of using different quantized kernels other than the closest for a 3rd order FIR filter



This understanding offers a valuable suggestion as to how to approach the original filter design question: one way to keep the frequency response in-spec, is to choose quantized filter coefficients that position the most important zeros nearest to their original, non-quantized positions. This raises the question: how does one determine "the most important" zeros of a filter? The answer to such a question depends on the exact specification of the filter: some specs may require a very aggressive stopband attenuation, some may care more about having fast rolloff, some may care about minimizing ripple in the passband. For a given spec, a filter may fail to reach the desired performance for a particular level of quantization due to a breach of spec at a particular portion of the frequency response. For example, after quantizing the coefficients, the resulting filter may break the spec somewhere in the middle of the stopband. This implies that we "care" more about the zeros that are influencing this region of the z-plane, and in turn, this "failed" portion of the unit circle. Instead of just using the closest quantized coefficients (and, in effect, giving equal "importance" to all zeros of the filter) if we could try and redistribute our quantized zeros to other valid positions such that they reflect our filter "priorities", it may be possible to find a new set of quantized coefficients that "rebalance" the frequency response of the filter.

To illustrate this hypothesis, consider figure (3). Here we find that the original zero of a length 37 filter using a non-quantized kernel occurred at the point marked in blue. After quantizing the kernel and using coefficients that have values closest to their original values, the resulting zero positions are plotted in red. The numerous other smaller points mark the range of zero positions that are reachable in the z-domain using quantized kernel coefficients that take on values other than the immediately closest quantization. Note that there are significantly closer zeros to the original zeros than the "closest quantization" zero positions. The caveat here is that while certain zeros are closer to the original, the same is not true for other zeros (with exception of the conjugate pairs, which experiences a similar improvement in proximity to the original zero).

Figure 3: Illustration of the effect of using different non-closest quantized 37 coefficient kernels

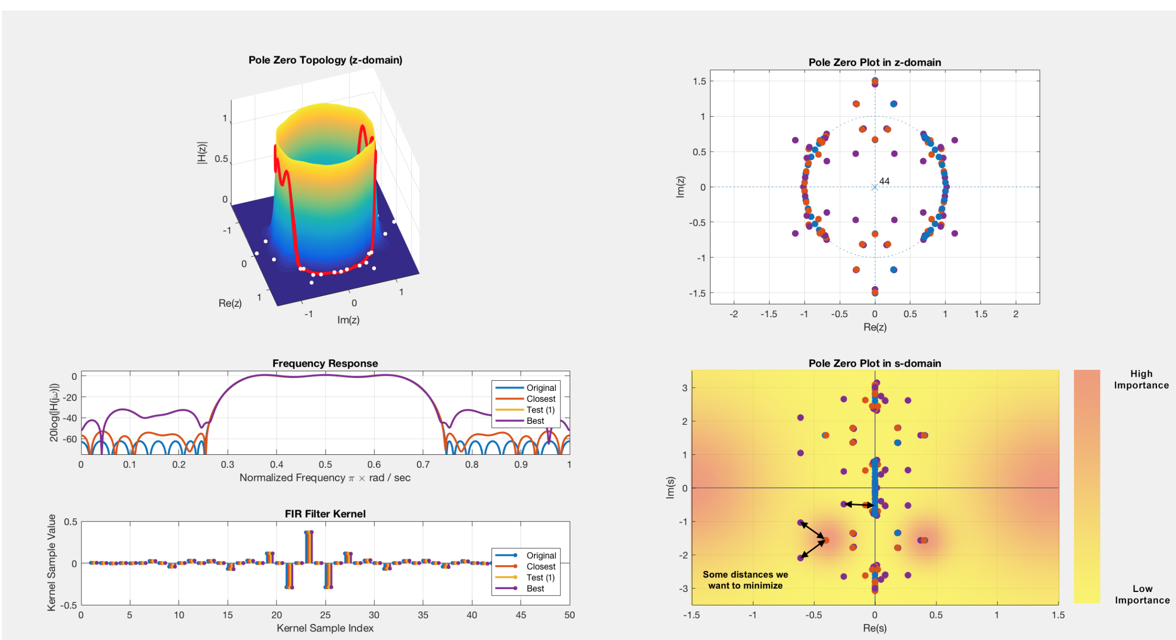


The original problem has now presented itself in a new form: using quantized coefficients other than the closest values to the original, how much closer can we get the most important zeros to the original zero positions? How much closer can we move these zeros to back to their original positions until the other zeros become the problematic ones? Depending on the exact nature of the filter spec, this may result in different degrees of freedom. If a particular filter spec is comparably strict in all regions of the frequency response, then the closest filter may be the best one can do. If, however, there is an imbalance in the specification severity, then it may be possible to find better quantized coefficient values other than the closest ones.

We have now developed a potential strategy to find optimal quantized filter coefficients. Starting from the original, non-quantized kernel coefficients, find the position of the resulting zeros in the z-domain. Next, based on the design spec, derive weights to mathematically describe which regions of the z-plane are of more importance than others in terms of how much we care about divergences from original zero positions. Finally, find a set of quantized coefficients that minimizes the sum of the weighted differences between original and quantized zero positions. In essence, we have translated the original problem into a non-linear mixed integer programming problem, whereby our decision variables are the zero locations in the z-domain. Our objective function can be expressed as the weighted sum of the differences between the quantized and non-quantized zeros, weighted based on some priority function that assigns a weight to a region of the z-plane derived from the filter spec. Our understanding of how zero locations relate to coefficient values (through algebraic multiplication) then gives us our non-linear integer constraints (when the resulting coefficient values are scaled appropriately to take on integer values).

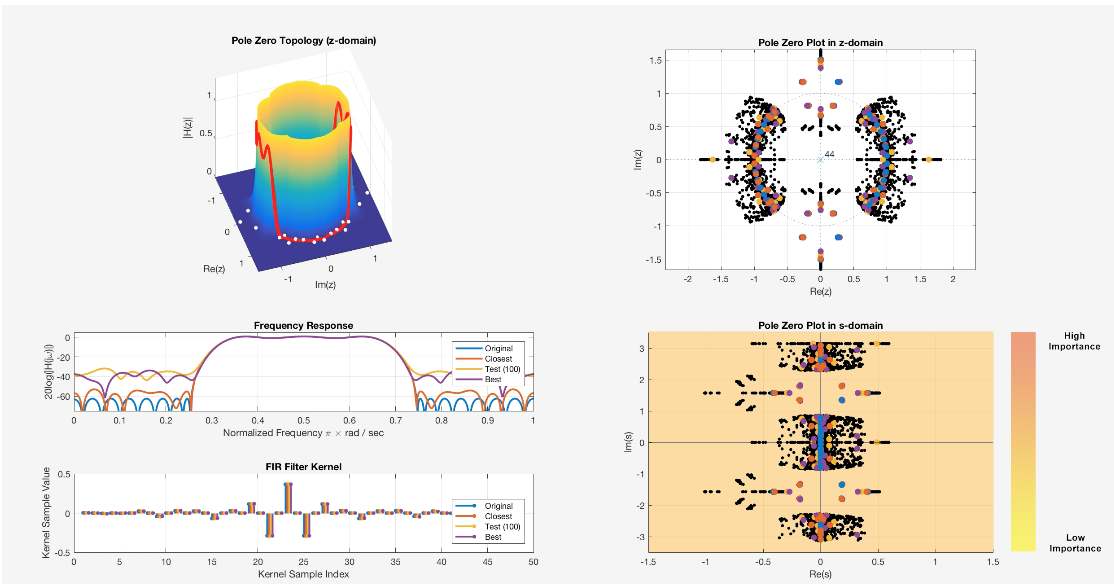
Figure (4) illustrates an example of this approach, showing how a "test" set of (non-closest) quantized kernel coefficients is assessed. The positions of the zeros that arise from using this alternative quantized kernel are shown in the z-plane. To derive the mathematical expression for the "importance" weighting, we transform our zeros into the s-domain (to "undo" to the non-linear co-ordinate system of the z-domain such that we can see more clearly which zeros have greater influence on the frequency response of the filter).

Figure 4: Illustration of optimisation process



We then observe that zeros nearer to the unit circle have a greater effect on the frequency response than zeros further away. Thus, we account for this in our "weighting" function, giving higher importance to zeros originally nearer to the unit circle. Next, we try to model the requirements of our filter, and specifically where our filter has the strictest requirements. This is illustrated by the yellow-orange gradient in the backgroun of the bottom right plot of Figure (4). Bear in mind that Figure (4) shows the first "test" kernel, and as such, the best filter found is also the only filter tested thus far. Figure (5) shows the result of testing 100 different non-closest quantized kernels, using an "equal-importance everywhere" weighting for simulation simplicity.

Figure 5: Testing 100 non-closest quantized kernels with equal importance weighting for all regions of s-plane



Unfortunately, as a lowly undergraduate, I do not possess the mathematical / optimization expertise, nor the necessary time to continue this approach any further. However, I have gained an appreciation for the difficulty of filter design under quantization constraints. Ultimately, I have achieved a filter design that meets the spec that was derived from random trial and error of filter kernels that uses the kernel obtained from non-quantized design and proceeds to find the lowest number of bits that can implement the kernel whilst still meeting the spec. This is very regrettable, as I am well aware that more efficient approaches are possible, but time does not permit any more work on this assignment. I hope I have demonstrated that I have thoroughly attempted to engage with the challenge despite the less than impressive bit total described below.

The most significant saving in the number of coefficients comes from ensuring that the filter spec is symmetric, resulting in the setting of every even sample of the kernel to zero. Such savings can be realized in practice by using a shifting unit in the filter implementation architecture that skips the element on the input that is to be multiplied by zero. Thus, instead of storing the value of zero, we need only shift our signal when appropriate.

In conclusion, it was empirically found that the filter specification can be met using 13 bits to represent the kernel designed using the equiripple command in matlab, making sure to keep the filter specification symmetric. The unquantized filter kernel contains 45 elements. 22 of these coefficients are zero, and taking advantage of the sample shifting property, we do not need to represent these coefficients. Thus, the total number of bits achieved in this report is

$$45 - 22 \text{ coefficients} \times 13 \text{ bits per coefficient} = 299 \text{ bits}$$

This kernel is described by the coefficients showin in Figure (6). Figure (7) shows the performance achieved using this kernel, compared to the kernel of the original, non-quantized kernel.

Figure 6: Original (left) and Quantized (right) Kernel coefficients

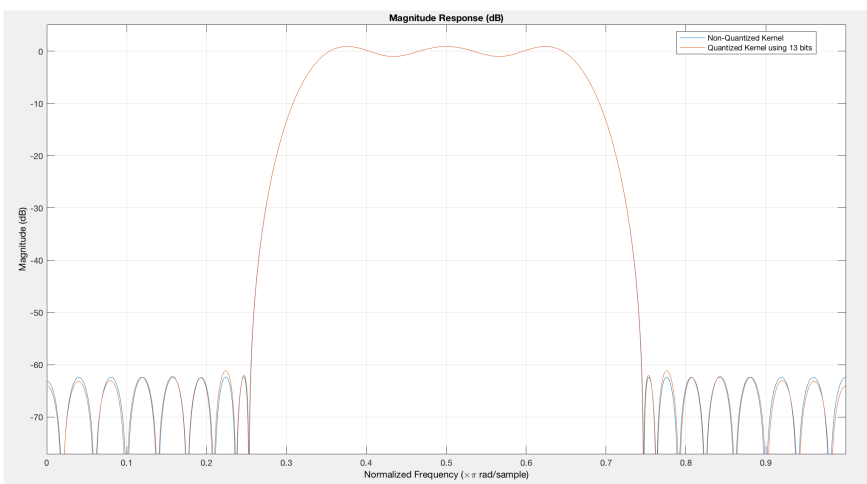
```
% Filter #1
% ------------------------------

Numerator:
 0.002623322998213782446252917424089901033
-0.000010428416743825550744772731603937643
-0.004446089153959903277069098948004466365
 0.000020240329794684833202749255076646762
-0.003262509681509084217415628259573168179
-0.000029117998131078832329578232052114117
 0.024550914465951294979317864886070310604
 0.000023409289825312975603021845016904479
-0.042939369437697223752525843565308605321
-0.00001841415933344545165510804107800169
 0.028343482255732854735441605953383259475
 0.000011339906583509512660455816057680778
 0.025151539974413071598657865024506463669
-0.000012086556577500497328472861546799777
-0.069262005442691801371246640428580576554
 0.000003180660224318265873107722599422154
 0.029868421969463841703085194012601277791
 0.000006016924105158789484813341291724598
 0.114981389007993886619551915373449446633
-0.000015313279516515938155109266549658287
-0.288555859707016248982824890845222398639
 0.000006166119368405007200410750112240521
 0.366620130367493635148434805159922689199
 0.000006166119368405007200410750112240521
-0.288555859707016248982824890845222398639
-0.000015313279516515938155109266549658287
 0.114981389007993886619551915373449446633
 0.000006016924105158789484813341291724598
 0.029868421969463841703085194012601277791
 0.000003180660224318265873107722599422154
-0.069262005442691801371246640428580576554
-0.000012086556577500497328472861546799777
 0.025151539974413071598657865024506463669
 0.000011339906583509512660455816057680778
 0.028343482255732854735441605953383259475
-0.00001841415933344545165510804107800169
-0.042939369437697223752525843565308605321
 0.000023409289825312975603021845016904479
 0.024550914465951294979317864886070310604
-0.000029117998131078832329578232052114117
-0.003262509681509084217415628259573168179
 0.000020240329794684833202749255076646762
-0.004446089153959903277069098948004466365
-0.000010428416743825550744772731603937643
 0.002623322998213782446252917424089901033
```

```
% Filter #2
% ------------------------------

Numerator:
 0.002596333035569551792032250148167804582
 0
-0.00447643626822336482967701343227417965
 0
-0.003223034113120823093701083905671112007
 0
 0.024530870749864042146271003730362281203
 0
-0.042973788174944307916014452075614826754
 0
 0.028380605940536134806917445416729606222
 0
 0.025157571827415313447939837487865588628
 0
-0.069295233432097688708317662076296983287
 0
 0.029902594271732078901049334263007040136
 0
 0.11495488336797601847116823137184837833
 0
-0.28855108184967814821675347047857940197
 0
 0.366620130367493635148434805159922689199
 0
-0.28855108184967814821675347047857940197
 0
 0.11495488336797601847116823137184837833
 0
 0.029902594271732078901049334263007040136
 0
-0.069295233432097688708317662076296983287
 0
 0.025157571827415313447939837487865588628
 0
 0.028380605940536134806917445416729606222
 0
-0.042973788174944307916014452075614826754
 0
 0.024530870749864042146271003730362281203
 0
-0.003223034113120823093701083905671112007
 0
-0.00447643626822336482967701343227417965
 0
 0.002596333035569551792032250148167804582
```

Figure 7: Performance Comparison of Original and Quantized Kernel



Video Animations and Code Repository

A video illustrating the effects of changing kernel coefficients on the zeros of a 2nd order FIR filter is available here:

`https://youtu.be/oL_yuIBm-DI`

A video illustrating the resulting zeros from using various non-closest quantized length 45 FIR kernels is available here:

`https://youtu.be/pdAyZDFFUmU`

The Matlab code used to generate these animations is available on the accompanying GitHub repository for this report at:

`https://github.com/cvigoe`