ECE503 Homework Assignment Number 9

1. 3 points. Mitra 11.12. If we measure "operations" as real multiplications, what is the asymptotic complexity of the DFT? What if we measure "operations" as real additions?
   Solution: From the definition of the DFT we see that the computation of $X[k]$ requires N complex multiplications and $N-1$ complex additions for each $k = 0, \dots, N-1$. Recall, for two complex numbers $x = a + jb$ and $y = c + jd$, the product $xy = ac - bd + j(ad + bc)$. Hence, there are four real multiplications and two real additions. The sum $x + y$ requires two real additions. Hence, the N complex multiplications result in $4N$ real multiplications and $2N$ real additions. The $N-1$ complex additions in the computation of $X[k]$ result in $2N-2$ real additions. The totals for each $k = 0, \dots, N-1$ are then

   - $4N$ real multiplications
   - $4N-2$ real additions

   which means that the computation of the DFT requires $4N^2$ real multiplications and $4N^2 - 2N$ real additions. The asymptotic complexity is $\mathcal{O}(N^2)$ irrespective of whether we use complex multiplies, complex additions, real multiplies, or real additions as our measure of operations.

2. 4 points. Mitra 11.27.
   Solution to part (a):

$$\mathbf{V}_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & W_8^0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & W_8^1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & W_8^2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & W_8^3 \\ 1 & 0 & 0 & 0 & W_8^4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & W_8^5 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & W_8^6 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & W_8^7 \end{bmatrix}, \quad \mathbf{V}_4 = \begin{bmatrix} 1 & 0 & W_8^0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & W_8^2 & 0 & 0 & 0 & 0 \\ 1 & 0 & W_8^0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & W_8^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & W_8^0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & W_8^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & W_8^0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & W_8^2 \end{bmatrix},$$

$$\mathbf{V}_2 = \begin{bmatrix} 1 & W_8^0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & W_8^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & W_8^0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & W_8^4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & W_8^0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & W_8^4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & W_8^0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & W_8^4 \end{bmatrix}, \quad \mathbf{E} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

As can be seen from the above, multiplication by each matrix $\mathbf{V}_{k}, k = 1,2,3$, requires at most 8 complex multiplications.

Solution to part (b):

(b) The transpose of the matrices given in Part **(a)** are as follows:

$$\mathbf{V}_8^t = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ W_8^0 & 0 & 0 & 0 & W_8^0 & 0 & 0 & 0 \\ 0 & W_8^1 & 0 & 0 & 0 & W_8^1 & 0 & 0 \\ 0 & 0 & W_8^2 & 0 & 0 & 0 & W_8^2 & 0 \\ 0 & 0 & 0 & W_8^3 & 0 & 0 & 0 & W_8^3 \end{bmatrix},$$

$$\mathbf{V}_4^t = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ W_8^0 & 0 & W_8^0 & 0 & 0 & 0 & 0 & 0 \\ 0 & W_8^2 & 0 & W_8^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & W_8^0 & 0 & W_8^0 & 0 \\ 0 & 0 & 0 & 0 & 0 & W_8^2 & 0 & W_8^2 \end{bmatrix},$$

$$\mathbf{V}_2^t = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ W_8^0 & W_8^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & W_8^0 & W_8^4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & W_8^0 & W_8^4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & W_8^0 & W_8^4 \end{bmatrix},$$

$$\mathbf{E}^t = \mathbf{E} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

It is easy to show that the flow-graph representation of $\mathbf{D}_8 = \mathbf{E}^t\mathbf{V}_2^t\mathbf{V}_4^t\mathbf{V}_8^t$ is precisely the 8-point DIF FFT algorithm of Figure 11.28.

3. 3 points. Mitra 11.32.
Solution to part (a): The number of zero-valued samples to be added is $256 - 197 = 59$.

Solution to part (b): The DFT will require $N^2 = 38809$ complex multiplications and $N^2 - N = 38612$ complex additions, where $N = 197$ in this case.

Solution to part (c): The FFT will require $N \log_2 N = 2048$ complex multiplications and $N \log_2 N = 2048$ complex additions, where $N = 256$ in this case. This $19\times$ less than direct DFT computation, even though the number of points used to compute the FFT is greater than the number of points used to compute the FFT. Note: The Mitra textbook also mentions a symmetry condition (p. 625 and Figure 11.23) that can be used to cut the number of multiplications required in half. With this clever trick, the FFT will require $\frac{N}{2} \log_2 N =$

1024 complex multiplications, which is even better. But the asymptotic complexity remains $\mathcal{O}(N \log_2 N)$ even with this trick.

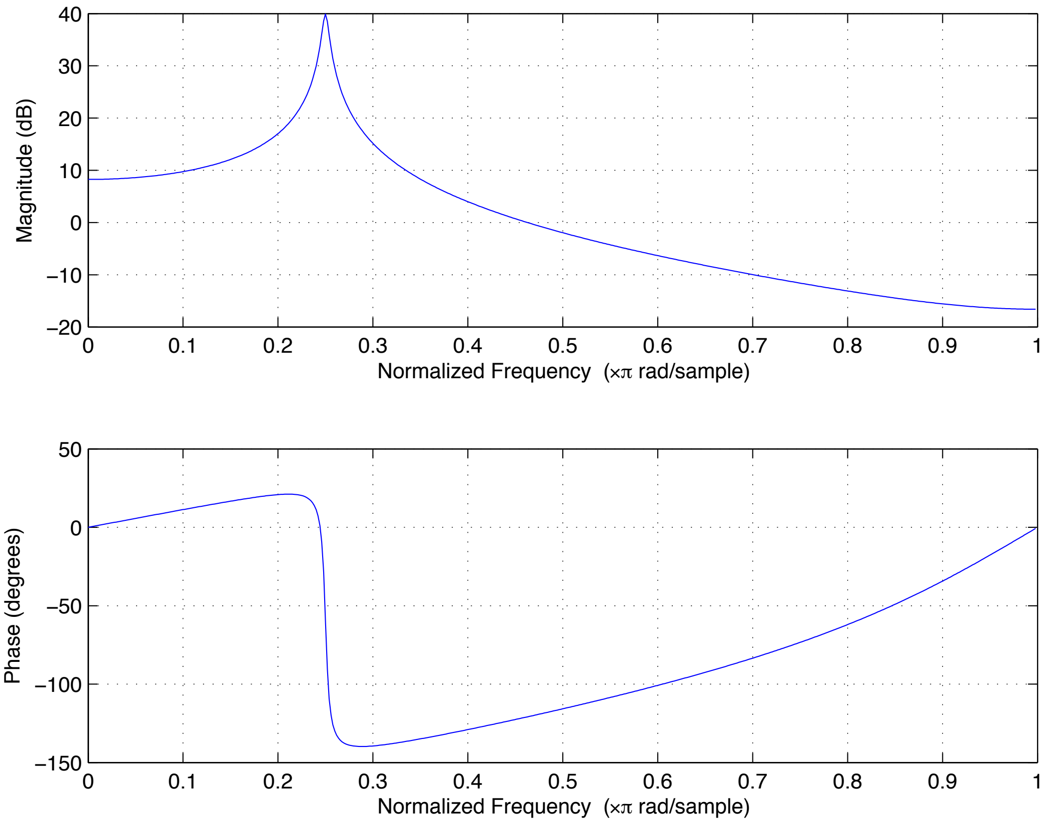
4. 7 points total. Suppose you have a causal IIR digital filter with transfer function

$$H(z) = \frac{1 + 0.5z^{-1}}{1 - 1.4z^{-1} + 0.98z^{-2}}$$

   (a) 1 point. Use Matlab to plot the magnitude response of this filter. Is this filter stable?
   Solution to part (a):

```
b = [1 0.5 0]
a = [1 -1.4 0.98]
freqz(b,a)
```

   Since `abs(roots(a)) = [0.9899,0.9899]` and the filter is causal, the ROC extends outward from $|z| > 0.9899$. Hence, the ROC contains the unit circle and the filter is stable. The magnitude and phase responses of this filter are plotted below.



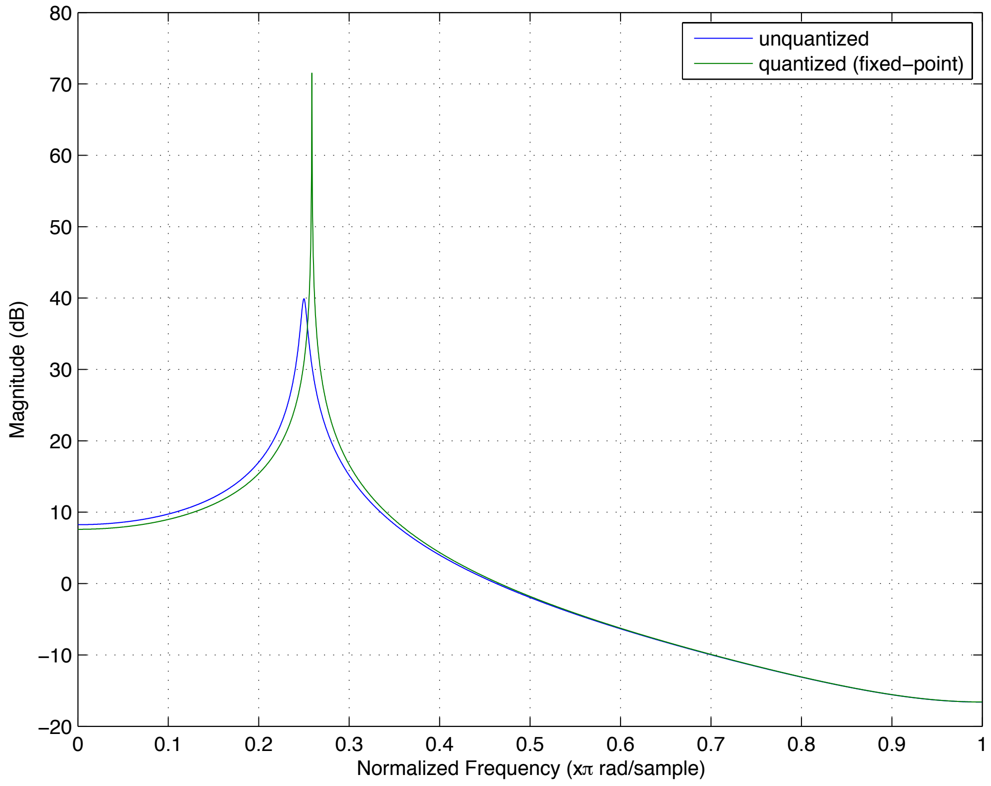
   (b) 3 points. Now suppose the filter coefficients are quantized to 8-bit signed fixed-point data-types with 4 fractional bits. Compute the quantized coefficients and write them in base-10 and binary representations, showing the binary decimal point explicitly. Re-plot the magnitude response of your filter with these quantized coefficients, comparing the results with the unquantized filter. Is the quantized filter stable?
   Solution to part (b):

```
% compute quantized filter coefficients with 4 fractional bits
% we don't need to worry about overflow here since the 8-bit/4-frac signed
% fixed-point number is large enough to capture the numerator and denominator coeffs
bq = round(b*2^4)/2^4
aq = round(a*2^4)/2^4
[hq,wq] = freqz(bq,aq,4096);
[h,w] = freqz(b,a,4096);
plot(w/pi,20*log10(abs(h)),wq/pi,20*log10(abs(hq)))
grid on
xlabel('Normalized Frequency (x\pi rad/sample)');
ylabel('Magnitude (dB)');
legend('unquantized','quantized (fixed-point)');
```

The numerator is unchanged by the conversion to 8-bit signed fixed-point data-types with 4 fractional bits. In binary, the first numerator coefficient is $0001_\Delta 0000$ and the second numerator coefficient is $0000_\Delta 1000$. There is no quantization error.

The denominator is changed by the conversion to 8-bit signed fixed-point data-types with 4 fractional bits. In decimal, the denominator coefficients become $[1, -1.375, 1]$. In binary, the first denominator coefficient is $0001_\Delta 0000$, the second numerator coefficient is $1110_\Delta 1010$, and the first denominator coefficient is $0001_\Delta 0000$.

The magnitude response of the quantized filter is plotted below and compared to the unquantized filter.



The quantized/fixed-point filter is **not stable** since the magnitude of the poles is one, hence the ROC can't contain the unit circle. We see this instability manifested as a very high gain peak at normalized frequency $\approx 0.26\pi$.

(c) 3 points. Given your filter coefficients must be quantized to 8-bit signed fixed-point data-types, what is the optimum number of fractional bits to use here? Can you improve the

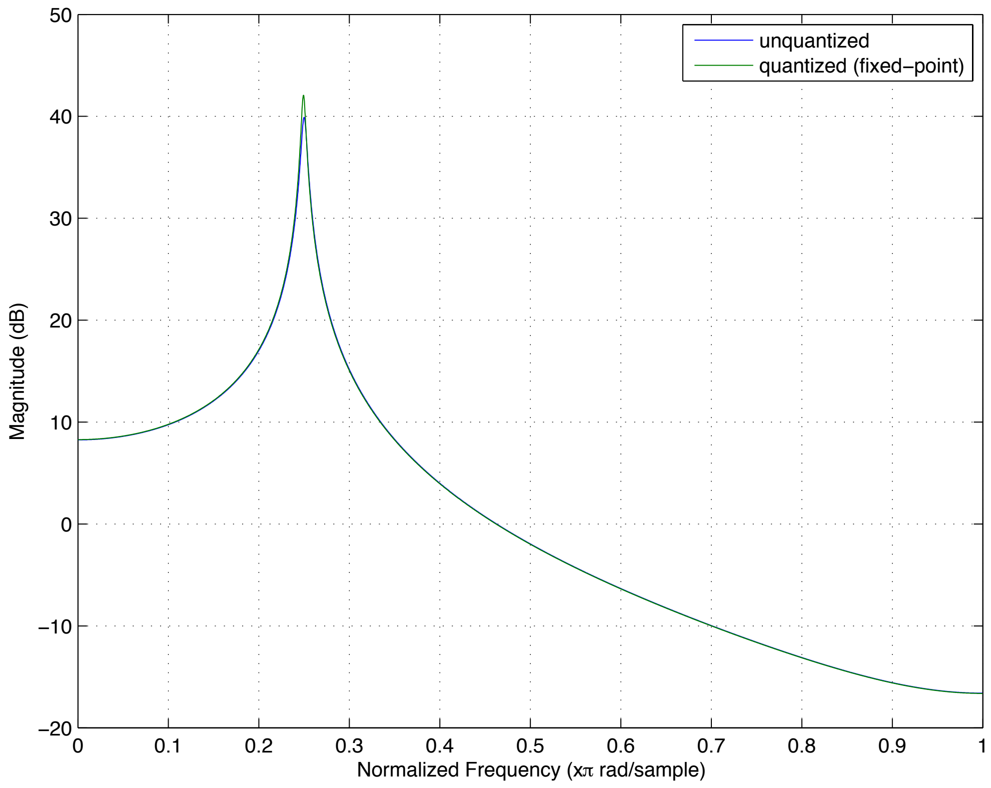
frequency response of your quantized-coefficient filter by using less/more fractional bits?
Solution to part (c): The largest magnitude filter coefficient we need to quantize is the -1.375 coefficient, hence we only need one non-fractional bit (in addition to the sign bit). Hence, we can have 6 fractional bits and avoid overflow. Here is my Matlab code:

```
% compute quantized filter coefficients with 6 fractional bits
% we don't need to worry about overflow here since the 8-bit/6-frac
% signed fixed-point number is large enough to capture the numerator
% and denominator coeffs
bq = round(b*2^6)/2^6;
aq = round(a*2^6)/2^6;
[hq,wq] = freqz(bq,aq,4096);
[h,w] = freqz(b,a,4096);
plot(w/pi,20*log10(abs(h)),wq/pi,20*log10(abs(hq)))
grid on
xlabel('Normalized Frequency (x\pi rad/sample)');
ylabel('Magnitude (dB)');
legend('unquantized','quantized (fixed-point)');
```

In this case, the deniminator coefficients become $a_q = [1, -1.4062, 0.9844]$ (or

$$[01_{\triangle}000000, 10_{\triangle}100110, 00_{\triangle}111111]$$

in binary) and the quantized filter is stable. The magnitude response of the quantized filter with 6 fractional bits for all coefficients is plotted below and compared to the unquantized filter. We see this fixed-point filter is giving a pretty good approximation of the unquantized filter.

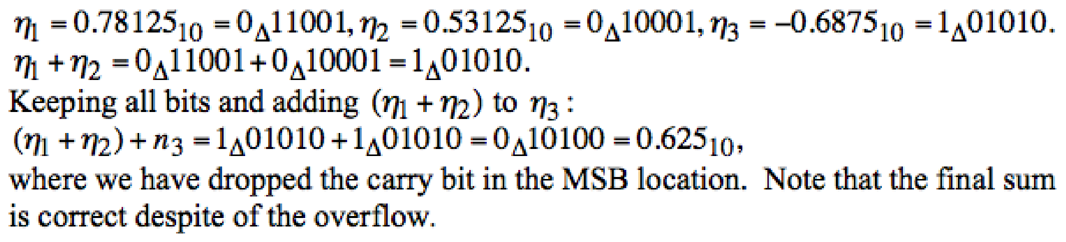


5. 4 points. Represent $x[n] = n(-\pi)^n$ in an 8-bit signed fixed-point data type with two fractional bits for $n = 0, \ldots, 3$ in both saturation overflow and wrapped overflow. Write your results in base-10 and binary and show your binary decimal point explicitly. Also compute the quantization error.
Solution: The unquantized values of $x[n]$ can be computed as $[0, -3.1416, 19.7392, -93.0188]$. The following table provides the 8-bit signed fixed-point data type with two fractional bits binary representations with saturation overflow and wrapped overflow. The largest positive value we can represent with this datatype is 31.75 and the largest negative value we can represent is -32. Hence overflow only occurs when $n = 3$.

Unquantized	Quantized decimal	Quantized binary (saturated)	Quantized binary (wrapped)
0	0.00	$000000_\Delta 00$	$000000_\Delta 00$
-3.1416	-3.25	$111100_\Delta 11$	$111100_\Delta 11$
19.7392	19.75	$010011_\Delta 11$	$010011_\Delta 11$
-93.0188	-93.00	$100000_\Delta 00$	$100011_\Delta 00$

Note $100011_\Delta 00$ is in -29 in decimal. This makes sense because -93 the wrapping effectively adds/subtracts 64 to the value to make it fall between -32 and +31.75. Here, if we add 64 to -93, we get -29. Saturation overflow is slightly better in this case because the saturated value when $n = 3$ is -32.

6. 4 points. Mitra 11.49. This problem assumes wrapped overflow. Also compute the final result assuming saturation overflow.
Solution:



$\eta_1 = 0.78125_{10} = 0_\Delta 11001, \eta_2 = 0.53125_{10} = 0_\Delta 10001, \eta_3 = -0.6875_{10} = 1_\Delta 01010.$
$\eta_1 + \eta_2 = 0_\Delta 11001 + 0_\Delta 10001 = 1_\Delta 01010.$
Keeping all bits and adding $(\eta_1 + \eta_2)$ to η_3:
$(\eta_1 + \eta_2) + n_3 = 1_\Delta 01010 + 1_\Delta 01010 = 0_\Delta 10100 = 0.625_{10},$
where we have dropped the carry bit in the MSB location. Note that the final sum is correct despite of the overflow.

If we were to do this with saturation overflow, then $\eta_1 + \eta_2 = 0_\Delta 11111$, which is the largest positive number we can represent with this fixed-point data type. Then, adding η_3, we get $(\eta_1 + \eta_2) + \eta_3 = 0_\Delta 00101 = \frac{9}{32} \neq 0.625$, hence the result is clearly incorrect. Usually saturation overflow is better, but this is one case where wrapped overflow is better than saturation overflow.