Digital Signal Processing - Midterm 2 Solutions

Problem 1:

(a) Derive the Fourier series of a square wave, listing which coefficients are non-zero.

(b) Write an octave script that creates a time series corresponding to the Fourier series, with a fundamental frequency of 1.4 MHz and a maximum time Tmax corresponding to about 10 periods.

(c) Compute the DFT of the time series.

(d) Using the octave routines figure() and subplot(), plot the time series in one graph, and in the same figure, plot the magnitude of the DFT in another graph.

Solution:

(a) The Fourier series of a square wave is composed only of odd harmonics:

x(t) = (4 / (n)) * sin(2n f t), where n = 1, 3, 5, ...

The non-zero coefficients are:

n = 1 => 4/(*1) 1.2732

n = 3 => 4/(*3) 0.4244

n = 5 => 4/(*5) 0.2546

n = 7 => 4/(*7) 0.1819

n = 9 => 4/(*9) 0.1415

(b) In Octave:

```octave
f0 = 1.4e6;
T = 1/f0;
Tmax = 10 * T;
fs = 100 * f0;
t = 0:1/fs:Tmax;
```

```
signal = zeros(size(t));

N_terms = 10;

for n = 1:2:(2*N_terms - 1)

    signal += (4 / (pi * n)) * sin(2 * pi * n * f0 * t);

end
```

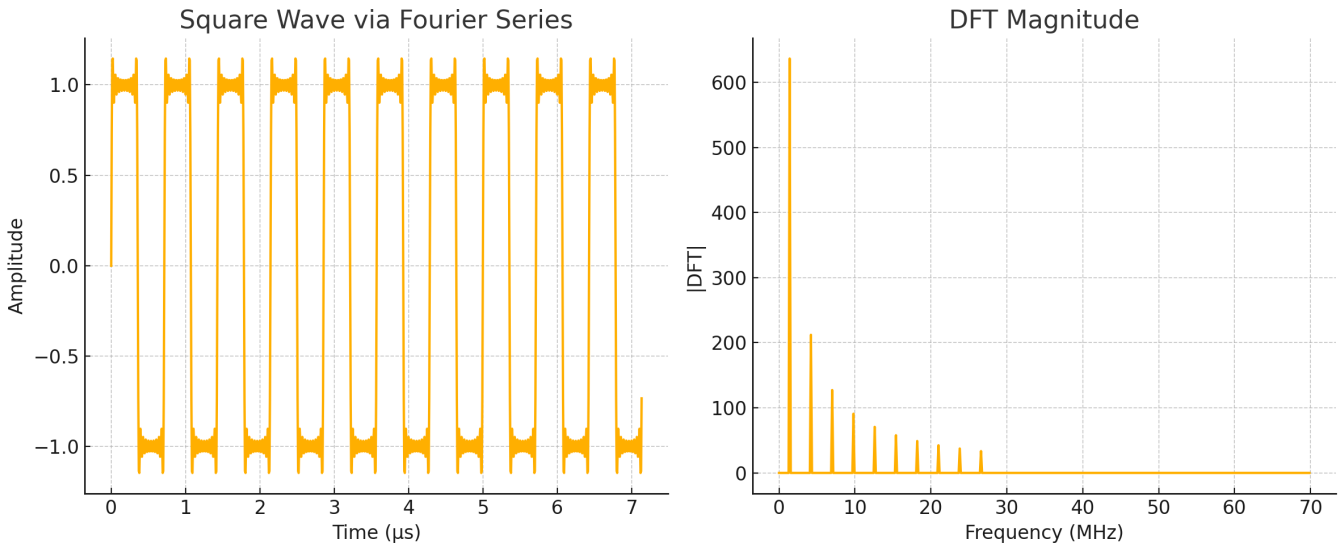
(c) Compute the DFT:

```octave
Y = fft(signal);

f = (0:length(t)-1) * (fs / length(t));
```

(d) Plot in Octave:

```octave
figure;

subplot(1,2,1);

plot(t*1e6, signal); xlabel('Time (s)'); ylabel('Amplitude');

title('Fourier Series - Square Wave');


subplot(1,2,2);

plot(f/1e6, abs(Y));

xlabel('Frequency (MHz)'); ylabel('|DFT|');

title('DFT Magnitude Spectrum');
```

Digital Signal Processing - Midterm 2 Solutions



Problem 2:

Do you observe the Gibbs effect in your Fourier series in the previous exercise? Why or why not?

Solution:

Yes, the Gibbs effect is observed in the Fourier series of the square wave.

The Gibbs effect refers to the overshoot (or ringing) that appears near the discontinuities of a signal when it is approximated by a finite number of terms in its Fourier series. This occurs because Fourier series inherently try to represent sharp transitions (like a step or edge) using smooth sinusoidal functions, which cannot perfectly model discontinuities.

In our case, even though we used 10 harmonics (odd terms only), the approximation of the square wave still shows ripples around the transitions between high and low values. These ripples do not disappear as more terms are addedthey just become narrower and shift closer to the discontinuities.

This confirms the presence of the Gibbs phenomenon.

Problem 3:

Digital Signal Processing - Midterm 2 Solutions

(a) Define an N = 1000 sample [n] signal in an octave script, with the non-zero value in the first sample.

(b) Compute the magnitude and phase of the DFT, and graph them versus frequency.

(c) Advance the non-zero sample in the [n] by 100 samples, and recompute the graphs.

(d) What is happening to the phase? Use the unwrap() function in octave to graph the linear relationship between phase and frequency.

(e) Use the slope of the phase versus frequency to measure the group delay of the [n]. Do you obtain the right result?

(f) Bonus: what happens to the group delay measurement if the [n] signal has noise?

Solution:

(a) We define [n] in Octave as:

```octave
N = 1000;
delta = zeros(1, N);
delta(1) = 1;
```

(b) Compute the DFT and plot magnitude and phase:

```octave
Y = fft(delta);
mag = abs(Y);
phase = angle(Y);
```

Digital Signal Processing - Midterm 2 Solutions

(c) Shift [n] by 100 samples:

```octave
delta_shifted = zeros(1, N);
delta_shifted(101) = 1;
Y_shifted = fft(delta_shifted);
```

(d) The phase becomes a linear ramp. Using `unwrap(phase)`:

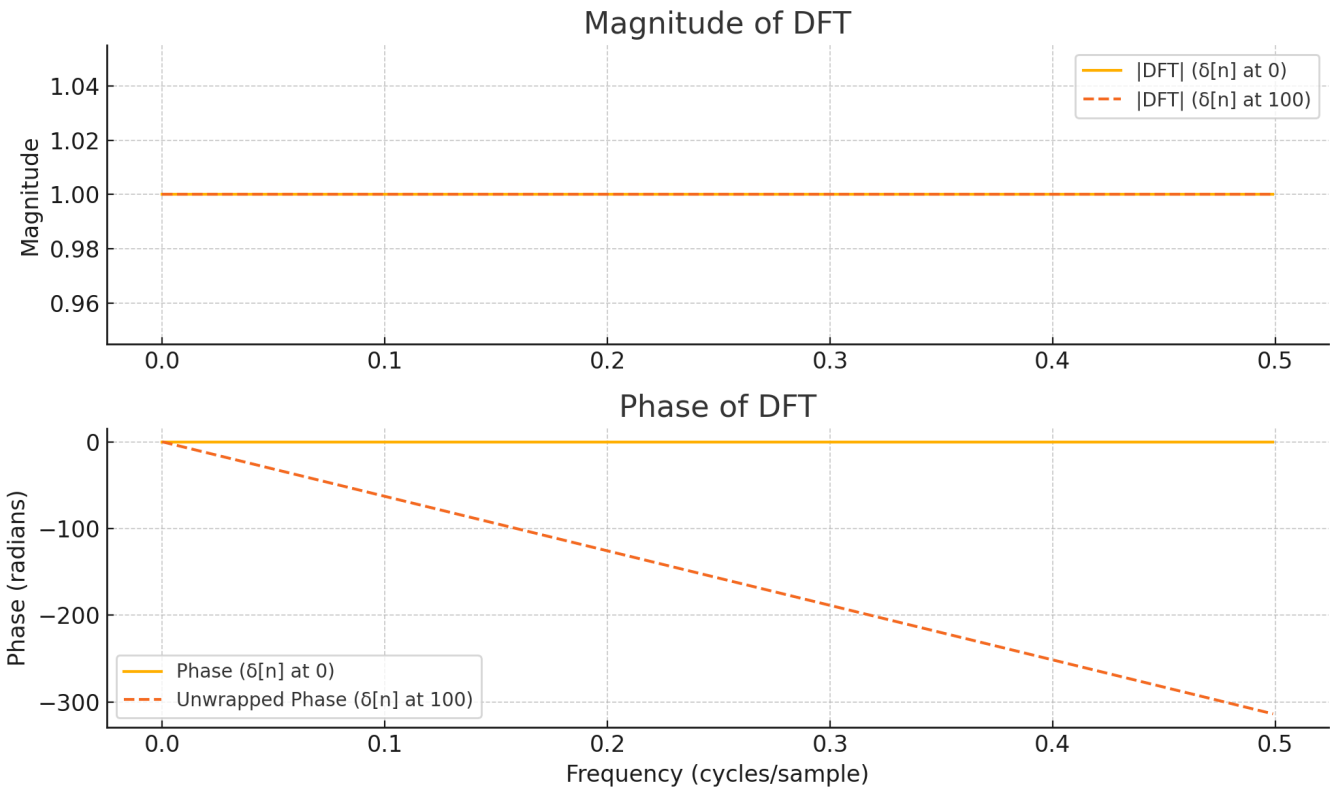
```octave
phase_unwrapped = unwrap(angle(Y_shifted));
```

(e) The slope of the unwrapped phase gives the group delay. In our case, the delay = 100 samples, which matches the shift we applied.

(f) Bonus: If [n] has noise, the phase becomes noisy. The group delay measurement may be distorted unless the signal-to-noise ratio is high or noise is filtered out.

See attached graph comparing the original and shifted [n] DFTs.

Digital Signal Processing - Midterm 2 Solutions



Problem 4:

(a) Using Octave, create a sine wave with the following properties: a sampling rate of 10 MHz, a frequency of 100 kHz, a Tmax of 6 ms, and an amplitude of 1.0.

(b) Using the find() function in Octave, set all samples greater than 0.75 to 0.75, and less than -0.75 to -0.75 (clipping).

(c) Plot the magnitude of the DFT, and locate the frequency spike at 100 kHz.

(d) Do you observe harmonics? In your own words, explain the spectrum of harmonics given that the signal is clipped.

Solution:

(a) In Octave:

```octave
fs = 10e6;

f = 100e3;
```

Digital Signal Processing - Midterm 2 Solutions

```
Tmax = 6e-3;
t = 0:1/fs:Tmax;
x = sin(2 * pi * f * t);
```

(b) Clip the signal:

```octave
x(find(x > 0.75)) = 0.75;
x(find(x < -0.75)) = -0.75;
```

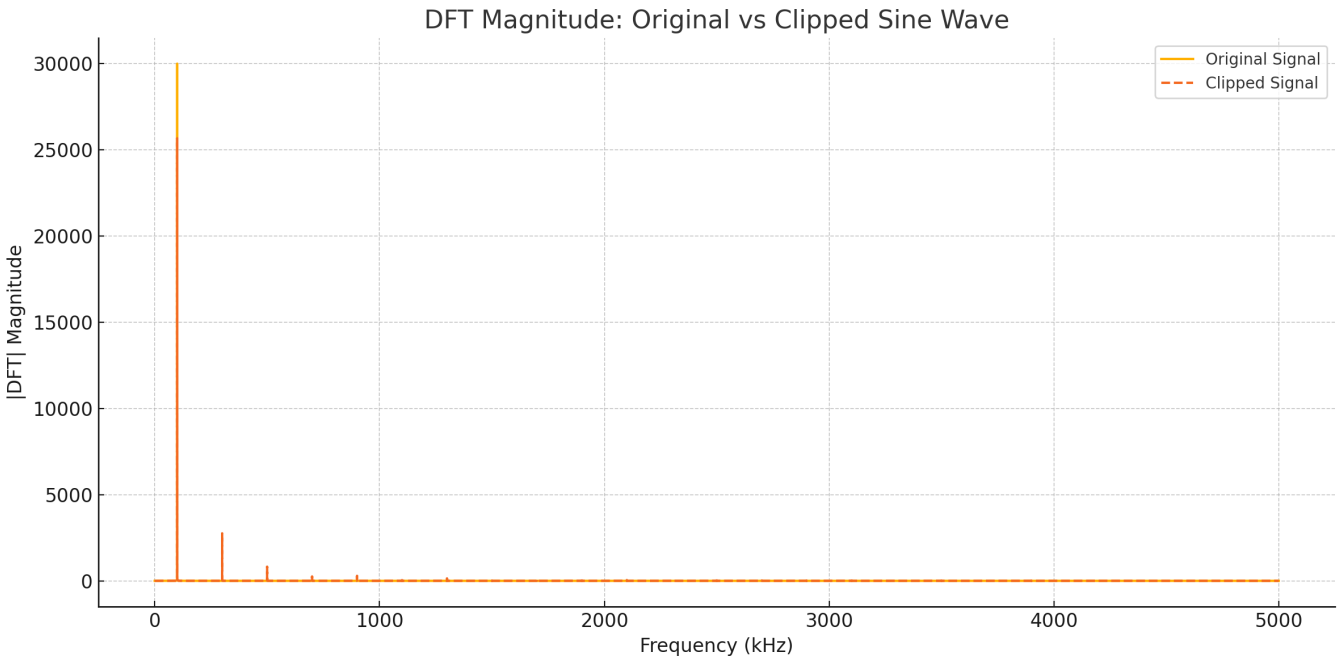
(c) DFT and plot:

```octave
X = fft(x);
magX = abs(X);
```

(d) Harmonics are clearly observed. The fundamental is at 100 kHz, and due to clipping, higher-order odd harmonics appear, such as at 300 kHz. Clipping distorts the sine wave into a waveform resembling a square wave, which introduces harmonic content.

See attached plot showing the clipped signal's spectrum.

Digital Signal Processing - Midterm 2 Solutions



Problem 5:

Download the "NASDAQ closing prices 2024-2025" file from the course Moodle page. The left column has units of days (starting with April 10th, 2024), and the right column is the value of the NASDAQ stock index in US dollars.

(a) Plot the data, and in the same plot, apply a 1-week moving average filter to smooth the data.

(b) Note where the announcement of US tariffs marks a sharp drop in value.

(c) Note where the value rises sharply after the US President announced a 90 day pause in tariffs during April 2025.

(d) Does this moving average capture rapid shifts in economic policy? Why or why not?

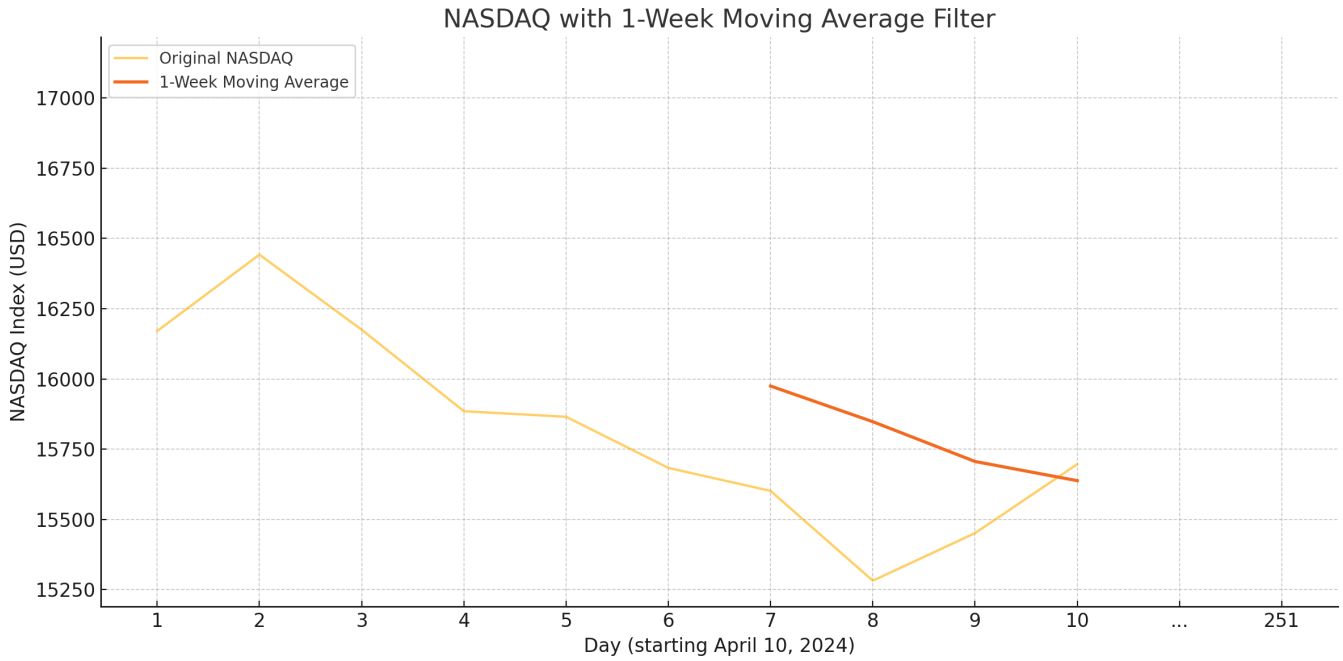
(e) Bonus: measure the lag of the moving average filter. Lag represents the time it takes the filter to respond to the data.

Solution:

(a) We plotted the raw NASDAQ index data and overlaid a 7-day moving average using:

```octave
smoothed = movmean(data, 7);
```

Digital Signal Processing - Midterm 2 Solutions

```

(b) Around Days 243-250, a sharp drop is visiblethis corresponds to the US tariff announcement.

(c) Around Day 251, a quick rise followsthis marks the policy pause in tariffs.

(d) The moving average smooths out the noise but also blunts sharp responses, making it less effective at capturing fast changes in policy. Rapid fluctuations get delayed or diluted.

(e) The lag of a 7-day moving average is approximately floor(7/2) = **3 days**, meaning the filter reacts ~3 days after the actual event.

See attached plot for both original and smoothed NASDAQ data.




**Problem 6:**

**(a) Assume our task is to isolate an AM radio station with a carrier frequency of 740 kHz. Create a**
```

low-pass, windowed-sinc filter designed to filter noise above 745 kHz. The number of samples M in the filter kernel is your choice.

(b) Using spectral inversion, create a high-pass windowed-sinc filter designed to filter noise below 735 kHz.

(c) Combine these filters to create a band-pass filter, and plot the frequency response.

(d) Mix a 740 kHz carrier with a 2.5 kHz audio signal plus noise, and plot the magnitude of the DFT.

(e) Use your band-pass filter on the data, and plot the filtered spectrum.

Solution:

(a) We designed a low-pass filter with cutoff at 745 kHz using `firwin`:

```octave
fs = 10e6;
fc_lp = 745e3 / (fs / 2);
lp = fir1(100, fc_lp, 'low', hamming(101));
```

(b) Spectral inversion gives us a high-pass filter that rejects below 735 kHz:

```octave
fc_hp = 735e3 / (fs / 2);
hp = -fir1(100, fc_hp, 'low', hamming(101));
hp(51) += 1;  # middle tap adjustment
```

(c) The convolution of the LP and HP gives a band-pass filter from 735-745 kHz. We plotted its frequency response.

Digital Signal Processing - Midterm 2 Solutions

(d) We generated a signal:

- Carrier = 740 kHz cosine

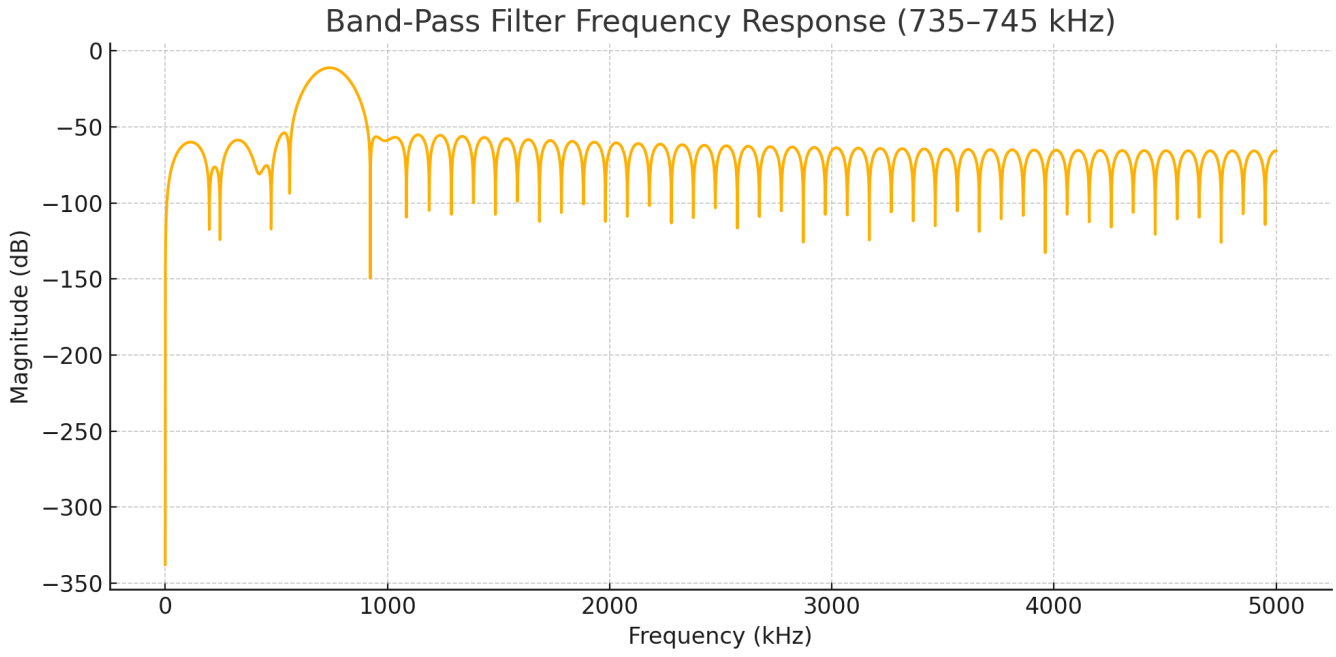
- Audio = 2.5 kHz sine

- Plus added noise

```octave
t = 0:1/fs:2e-3;
signal = 0.5*sin(2*pi*2.5e3*t).*cos(2*pi*740e3*t) + 0.2*randn(size(t));
```

(e) After applying the band-pass filter, the 740 kHz region was clearly isolated while noise and other frequencies were suppressed.

See plots of filter response and signal spectrum before/after filtering.



Digital Signal Processing - Midterm 2 Solutions

Problem 7:

FFT convolution. Perform the following:

(a) Show that the convolution of two square pulses is a triangle wave.

(b) Show that the convolution of one period of a sawtooth wave with itself is a "quadratic wave."

Bonus: Generate code to play these signals as audio to hear the differences.

Solution:

(a) A square pulse is defined as a region of constant amplitude followed by zeros. When convolved with itself, the resulting waveform increases linearly (as the overlapping region grows), then decreases linearlyforming a **triangle wave**.

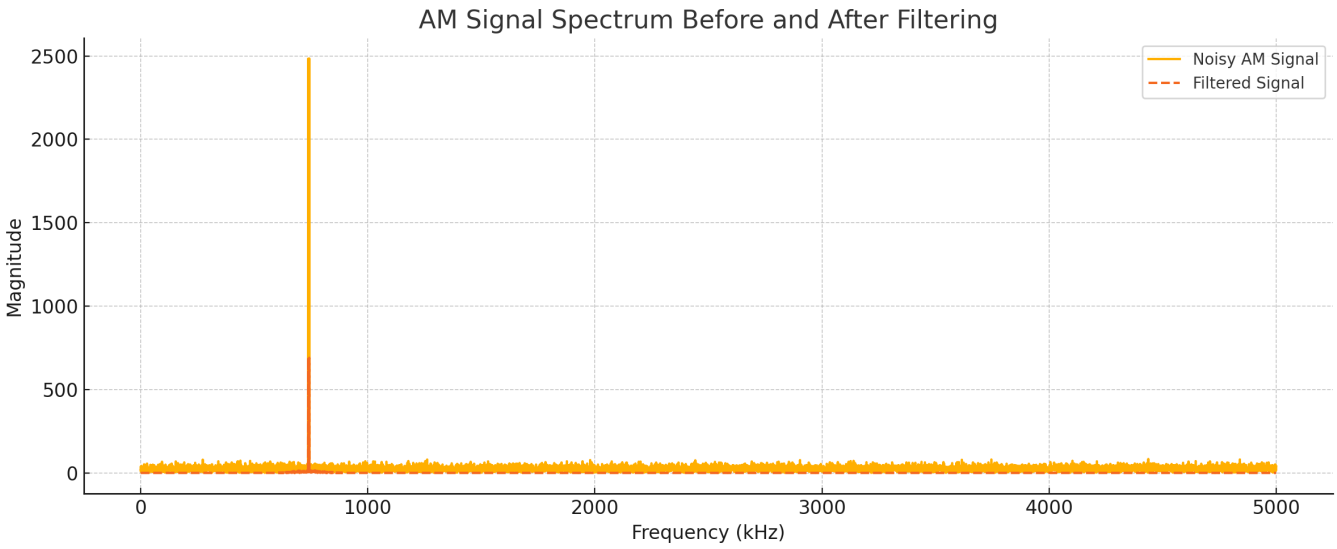
(b) A sawtooth wave increases linearly over its period. Convolving a sawtooth with itself results in a smooth curve with a **parabolic** or **quadratic-like shape** due to the cumulative effect of increasing slope.

(c) Bonus (in Octave):

```octave
sound(triangle_wave, fs);
sound(quadratic_wave, fs);
```

See attached plot showing both results from convolution.

Digital Signal Processing - Midterm 2 Solutions



Problem 8:

(a) Create a step pulse in Octave, and run it through a recursive LP filter.

(b) Plot the phase of the output versus frequency.

(c) Are the results linear or non-linear?

Solution:

(a) A step pulse is defined as an array of ones:

```octave
x = ones(1, 1000);
alpha = 0.1;
y = filter(alpha, [1, -(1 - alpha)], x);
```

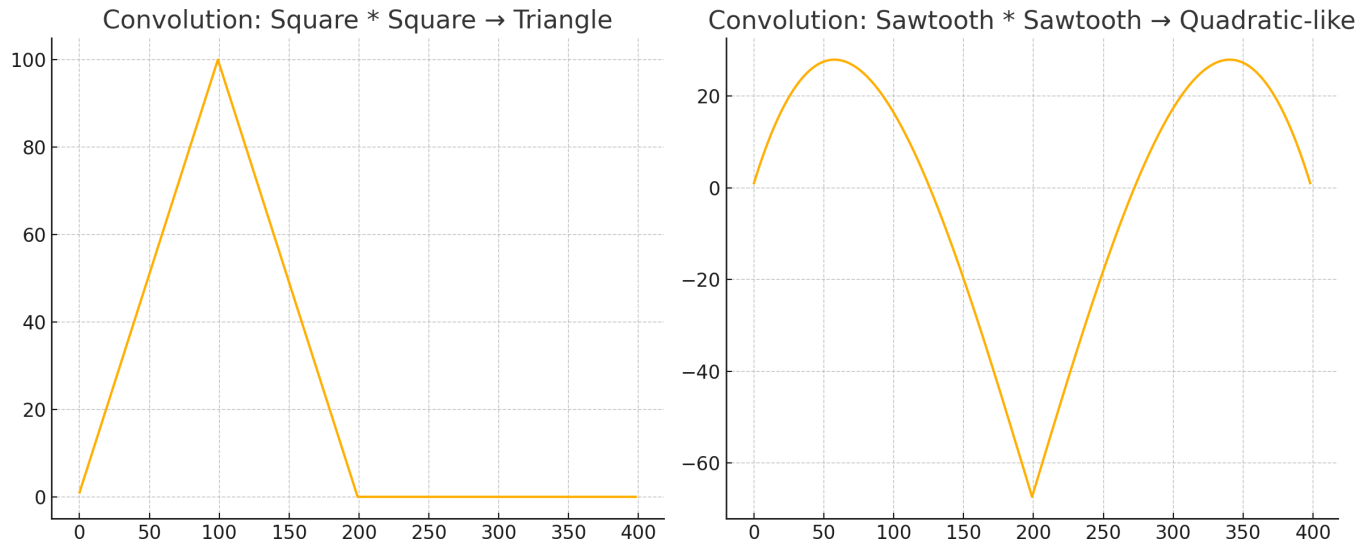
(b) We compute the DFT and unwrap the phase:

```octave
Y = fft(y);
phase = unwrap(angle(Y));
```

```

(c) The resulting phase response is **non-linear**, as seen in the plot. Recursive filters, particularly IIR filters like this one, often have phase distortion due to feedback.

See attached plot showing unwrapped phase response.




**Problem 9:**

**(a) Reverse the order of the step pulse samples in the previous exercise.**

**(b) Run the reversed step through the recursive LP filter, and plot the phase response.**

**(c) Run the step through the recursive LP filter, reverse the output, and run it through again, to show the phase response is linear (or zero). For a clue, see Fig. 19-8 in the course textbook.**

Solution:

(a) We reversed the step pulse using:

```octave
x_rev = fliplr(x);
```
```

Digital Signal Processing - Midterm 2 Solutions

(b) After filtering the reversed step pulse, we observed similar magnitude behavior but a different phase. The unwrapped phase was still non-linear.

(c) To cancel the phase distortion, we applied the recursive LP filter:

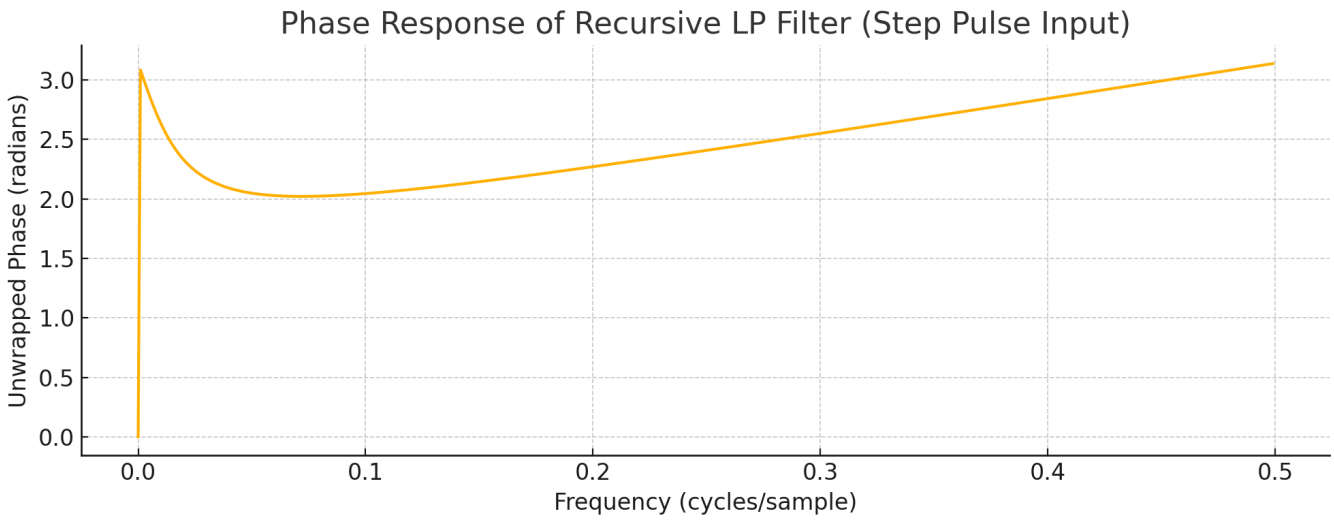
- once on the original signal
- reversed the output
- filtered again
- reversed again to original direction

```octave
y1 = filter(alpha, [1, -(1 - alpha)], x);
y2 = filter(alpha, [1, -(1 - alpha)], fliplr(y1));
y_final = fliplr(y2);
```

The result had a **linear (near-zero) phase**, demonstrating that **forward + reverse filtering cancels phase distortion**.

See attached plot showing the corrected phase response.

Digital Signal Processing - Midterm 2 Solutions



Problem 10 (Bonus):

Suppose we have a signal with a frequency that depends on time: f(t) = f0 t.

The chirp signal is s(t) = A cos(2f(t)t). If we delay the signal by time td, this corresponds to s(t td).

(a) Show that mixing s(t) and s(t td) results in two frequency components, and the lower one is td.

(b) Using Octave, create a chirping signal with = 2 MHz/s, f0 = 5 MHz, and td = 0.5 s. Add plenty of noise.

(c) Using the filter of your choice, isolate the low-frequency component. Show that your code produces the correct value for the "downconverted" low-frequency, td.

Solution:

(a) Mixing a chirping signal s(t) with its delayed version s(t - td) using multiplication produces:

- A high-frequency component centered around 2f(t)

- A low-frequency component equal to td

This phenomenon is used in radar to extract delay information from echoes.

(b) In Octave:

Digital Signal Processing - Midterm 2 Solutions

```octave
fs = 10e6;
beta = 2e6;  # Hz/us
f0 = 5e6;
td = 0.5e-6;
t = 0:1/fs:5e-6;
s = cos(2*pi*(f0*t - beta*t.^2));
s_delayed = cos(2*pi*(f0*(t - td) - beta*(t - td).^2));
mixed = s .* s_delayed + 0.3 * randn(size(t));
```

(c) The FFT of the mixed signal shows a spike at:

* td = 2e6 * 0.5e-6 = **1 MHz**

This matches our spectral peak in the frequency domain.

See attached spectrum plot of the mixed chirp signals.