QUIZ #2 - DIGITAL SIGNAL PROCESSING

NAILYNLOPEZ

1) a

$$f(t) = \begin{cases} A, & 0 < t < T/2 \\ -A, & -T/2 < t < 0 \end{cases}$$

$$f(t) = \sum_{n=1,2,3\ldots}^{\infty} \frac{4A}{n\pi} \sin\left(\frac{2\pi n t}{T}\right)$$

b octave

c octave

d octave

2) Yes, I observe Gibbs effect because the wave is not continuous

3) a

↓ octave

f

4) a

↓ octave

c

d Yes, there are harmonics, 100kHz, 300 kHz, 500kHz, etc (& all the odd ones).

5)

a octave (I did 1-10)

b) 7th day to the 8th
(15601.5) (15282.07)

c) 8th day to the 9th
(15282.07) (15451.31)

9th day to the 10th
(15451.31) (15696.64)

d) Somewhat, the moving average captures rapid shifts.

e octave

6 octave

7 octave

8 a octave

b

c The phase response is non-linear

9 a $s(t) = A\cos(2\pi f t)$

$A\cos(2\pi(f_0 - Bt)t)$

$s(t - t_d) = A\cos(2\pi(f_0 - B(t - t_d))(t - t_d))$

$s(t) \cdot s(t - t_d) = \frac{A^2}{2}\left[\cos(\phi + (t)) + \cos(\phi - (t))\right]$

$f = Bt_d$

lower

1

b)

```
f0 = 1.4e6;
T = 1 / f0;
Tmax = 10 * T;
A = 1;
N = 11;
Fs = 50e6;
dt = 1 / Fs;
t = 0:dt:Tmax;
signal = zeros(size(t));
for k = 1:2:N
 signal += (4*A)/(k*pi) * sin(2*pi*k*f0*t);
end
```

c)

```
Y = fft(signal);
n = length(Y);
f = (0:n-1)*(Fs/n);
```

d)

```
figure;
subplot(2,1,1);
plot(t*1e6, signal);
xlabel('Time (\mus)');
ylabel('Amplitude');
title('Square Wave Fourier Approximation');
subplot(2,1,2);
plot(f/1e6, abs(Y));
xlabel('Frequency (MHz)');
ylabel('|DFT|');
title('Magnitude of the DFT');
xlim([0 10*f0/1e6]);
```

3.

a)

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

b)

```
Y = fft(delta);
magY = abs(Y);
```

```
phaseY = angle(Y);

f = (0:N-1) / N;
figure;
subplot(2,1,1);
plot(f, magY);
xlabel('Normalized Frequency');
ylabel('Magnitude');
title('DFT Magnitude of δ[n]');

subplot(2,1,2);
plot(f, phaseY);
xlabel('Normalized Frequency');
ylabel('Phase (radians)');
title('DFT Phase of δ[n]');
```

c)

```
delta_shifted = zeros(1, N);
delta_shifted(101) = 1;
Y_shifted = fft(delta_shifted);
magY_shifted = abs(Y_shifted);
phaseY_shifted = angle(Y_shifted);
figure;
subplot(2,1,1);
plot(f, magY_shifted);
xlabel('Normalized Frequency');
ylabel('Magnitude');
title('Magnitude of Shifted δ[n]');
subplot(2,1,2);
plot(f, phaseY_shifted);
xlabel('Normalized Frequency');
ylabel('Phase (radians)');
title('Phase of Shifted δ[n]');
```

d)

```
phase_unwrapped = unwrap(phaseY_shifted);
figure;
plot(f, phase_unwrapped);
xlabel('Normalized Frequency');
ylabel('Unwrapped Phase (radians)');
title('Unwrapped Phase vs Frequency');
```

e)

```
df = 1/N;
```

```
dphi = diff(phase_unwrapped);
group_delay = -dphi / df;
avg_delay = mean(group_delay);
fprintf('Estimated group delay:
```

f)

```
delta_noisy = delta_shifted + 0.1 * randn(1, N);
Y_noisy = fft(delta_noisy);
phase_noisy = unwrap(angle(Y_noisy));
dphi_noisy = diff(phase_noisy);
group_delay_noisy = -dphi_noisy / df;
avg_delay_noisy = mean(group_delay_noisy);
fprintf('Estimated group delay with noise:
```

4.

a)

```
Fs = 10e6;
f = 100e3;
Tmax = 6e-3;
A = 1.0;
t = 0 : 1/Fs : Tmax;
x = A * sin(2*pi*f*t);
fprintf('Total samples: length(x));
```

b)

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

c)

```
N = length(x_clipped);
X = fft(x_clipped);
f_axis = (0:N-1) * Fs / N;
magX = abs(X);

figure;
plot(f_axis / 1e6, magX);
xlabel('Frequency (MHz)');
ylabel('|DFT|');
title('Magnitude Spectrum of Clipped Sine Wave');
xlim([0 2]);
```

5.

a)

```
days = [1 2 3 4 5 6 7 8 9 10];
prices = [16170.36 16442.2 16175.09 15885.02 15865.25 15683.37 15601.5 15282.01
15451.31 15696.64];
window = 7;
moving_avg = filter(ones(1, window)/window, 1, prices);
figure;
plot(days, prices, 'b-o', 'DisplayName', 'Original Data');
hold on;
plot(days, moving_avg, 'r-', 'LineWidth', 2, 'DisplayName', '7-day Moving Average');
xlabel('Day (starting April 10, 2024)');
ylabel('NASDAQ Closing Price (USD)');
title('NASDAQ Closing Prices with 7-day Moving Average');
legend;
grid on;
```

e)

```
lag = window - 1;
fprintf('Lag of moving average: days\n', lag);
```

6)

a.

```
Fs = 3e6;
fc_lp = 745e3;
M = 101;
fc_lp_norm = fc_lp / (Fs/2);
n = 0:M-1;
h_lp = sinc(fc_lp_norm * (n - (M-1)/2));
w = hamming(M)';
h_lp = h_lp .* w;
h_lp = h_lp / sum(h_lp);
```

b.

```
fc_hp = 735e3;
fc_hp_norm = fc_hp / (Fs/2);
h_lp2 = sinc(fc_hp_norm * (n - (M-1)/2)) .* w;
h_lp2 = h_lp2 / sum(h_lp2);
h_hp = -h_lp2;
h_hp((M-1)/2 + 1) += 1;
```

c.

```
h_bp = conv(h_lp, h_hp);  % Band-pass = LP * HP (convolution)
N = 1024;
H = fft(h_bp, N);
f = linspace(0, Fs/2, N/2);
```

```
figure;
plot(f/1e6, abs(H(1:N/2)));
xlabel('Frequency (MHz)');
ylabel('|H(f)|');
title('Band-pass Filter Frequency Response');
grid on;
```

d.

```
T = 0.01;
t = 0:1/Fs:T;
f_carrier = 740e3;
f_audio = 2.5e3;
audio = sin(2*pi*f_audio*t);
noise = 0.4 * randn(size(t));
x = (1 + audio) .* cos(2*pi*f_carrier*t) + noise;I
X = fft(x, N);
f_axis = (0:N-1) * Fs / N;
figure;
plot(f_axis(1:N/2)/1e6, abs(X(1:N/2)));
xlabel('Frequency (MHz)');
ylabel('|X(f)|');
title('Spectrum of AM Signal + Noise');
grid on;
```

e.

```
x_filtered = conv(x, h_bp, 'same');
Xf = fft(x_filtered, N);
figure;
plot(f_axis(1:N/2)/1e6, abs(Xf(1:N/2)));
xlabel('Frequency (MHz)');
ylabel('|Filtered X(f)|');
title('Filtered Spectrum (Band-Pass Applied)');
grid on;
```

7)

a.

```
N = 1024;
pulse = zeros(1, N);
pulse(200:300) = 1;
fft_size = 2*N;
X1 = fft(pulse, fft_size);
X2 = fft(pulse, fft_size);
Y = ifft(X1 .* X2);
```

```
figure;
plot(Y);
title('Convolution of Two Square Pulses = Triangle');
xlabel('Sample');
ylabel('Amplitude');
grid on;
```

b.

```
saw = linspace(-1, 1, N);
S1 = fft(saw, 2*N);
S2 = fft(saw, 2*N);
Q = ifft(S1 .* S2);
figure;
plot(Q);
title('Convolution of Sawtooth with Itself = Quadratic Shape');
xlabel('Sample');
ylabel('Amplitude');
grid on;
```

c.

```
Fs = 44100;
triangle = real(Y);
triangle = triangle / max(abs(triangle));
quad_wave = real(Q);
quad_wave = quad_wave / max(abs(quad_wave));
sound(triangle(1:Fs), Fs);
pause(1.5);
sound(quad_wave(1:Fs), Fs);
```

8.

a)

```
N = 1024;
step = ones(1, N);
alpha = 0.9;
y = zeros(1, N);
for n = 2:N
    y(n) = (1 - alpha) * step(n) + alpha * y(n-1);
end
figure;
plot(1:N, y);
title('Output of Recursive Low-Pass Filter to Step Input');
xlabel('Sample');
ylabel('Amplitude');
```

```
grid on;

b)
impulse = [1, zeros(1, N-1)];
h = zeros(1, N);
for n = 2:N
   h(n) = (1 - alpha) * impulse(n) + alpha * h(n-1);
end
H = fft(h);
f = linspace(0, 1, N);
phaseH = angle(H);
phase_unwrapped = unwrap(phaseH);
figure;
plot(f, phase_unwrapped);
title('Phase Response of Recursive LP Filter');
xlabel('Normalized Frequency (×π rad/sample)');
ylabel('Phase (radians)');
grid on;

9)
a.
step = ones(1, 1024);
step_rev = fliplr(step);

b)
alpha = 0.9;
N = length(step_rev);
y_rev = zeros(1, N);
for n = 2:N
   y_rev(n) = (1 - alpha) * step_rev(n) + alpha * y_rev(n-1);
end
impulse_rev = [1, zeros(1, N-1)];
h_rev = zeros(1, N);
for n = 2:N
   h_rev(n) = (1 - alpha) * impulse_rev(n) + alpha * h_rev(n-1);
end
H_rev = fft(h_rev);
f = linspace(0, 1, N);
phase_rev = unwrap(angle(H_rev));
figure;
plot(f, phase_rev);
title('Phase Response of Recursive LP Filter (Reversed Step Input)');
xlabel('Normalized Frequency');
ylabel('Phase (radians)');
```

```
grid on;
```

c)

```
step = ones(1, N);
y1 = zeros(1, N);
for n = 2:N
    y1(n) = (1 - alpha) * step(n) + alpha * y1(n-1);
end
y1_rev = fliplr(y1);
y2 = zeros(1, N);
for n = 2:N
    y2(n) = (1 - alpha) * y1_rev(n) + alpha * y2(n-1);
end
y_zero_phase = fliplr(y2);
H_zp = fft(y_zero_phase);
phase_zp = unwrap(angle(H_zp));
figure;
plot(f, phase_zp);
title('Zero-Phase Filter Response (Forward-Backward)');
xlabel('Normalized Frequency');
ylabel('Phase (radians)');
grid on;
```