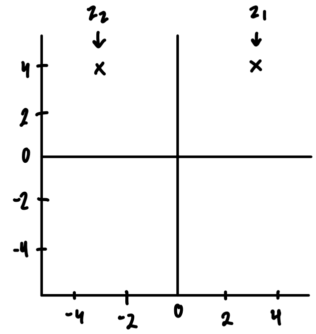
Complex Numbers and Signals

Aman Kumpavat

HW#1

1a)



1b) $z_1 + z_2 = (3+4j) + (-3+4j) = 8j$

1c) $z_1 - z_2 = (3+4j) - (-3+4j) = 6$

1d) $z_1 \cdot z_2 = (3+4j) \cdot (-3+4j) = -25$

1e) $z_1 / z_2 = \dfrac{(3+4j)(-3-4j)}{(-3+4j)(-3-4j)} = \dfrac{-25-24j}{25} = 0.28 - 0.96j$

1f) $|z_1| = \sqrt{9+16} = 5$

1g) $|z_2| = \sqrt{9+16} = 5$

1h) $\phi_1 = \arg(z_1) = \tan^{-1}\left(\frac{4}{3}\right) = \approx 0.92 \text{radians}$

1i) $\phi_2 = \arg(z_2) = \tan^{-1}\left(\frac{4}{-3}\right) = \approx 2.214 \text{radians}$

1j) $z_1 = 5\angle 0.927$ radians

$z_2 = 5\angle 2.214$ radians

2) $\cos(2\pi f t)$:

$$\cos(2\pi f t) = \frac{e^{2\pi i f t} + e^{-2\pi i f t}}{2}$$

$$e^{2\pi i f t} = \cos(2\pi f t) + j\sin(2\pi f t)$$

$$e^{-2\pi i f t} = \cos(2\pi f t) - j\sin(2\pi f t)$$

$$e^{2\pi i f t} + e^{-2\pi i f t} = 2\cos(2\pi f t)$$

so, $\cos(2\pi f t) = \dfrac{e^{2\pi i f t} + e^{-2\pi i f t}}{2}$

$\sin(2\pi f t)$:

$$\sin(2\pi f t) = \frac{e^{2\pi i f t} - e^{-2\pi i f t}}{2j}$$

$$e^{2\pi i f t} = \cos(2\pi f t) + j\sin(2\pi f t)$$

$$e^{-2\pi i f t} = \cos(2\pi f t) - j\sin(2\pi f t)$$

$$e^{2\pi i f t} - e^{-2\pi i f t} = 2j\sin(2\pi f t)$$

$$\sin(2\pi f t) = \frac{e^{2\pi i f t} - e^{-2\pi i f t}}{2j}$$

3a) $\cos(A)\cos(B) = \frac{1}{2}[\cos(A-B) + \cos(A+B)]$

$A = 2\pi f_1 t \quad , \quad B = 2\pi f_2 t - \phi$

$P = v_1(t)v_2(t) = 4\cos(2\pi f_1 t)\cdot 4\cos(2\pi f_2 t - \phi)$

$P = 16\cdot\frac{1}{2}[\cos((2\pi f_1 t) - (2\pi f_2 t - \phi)) + \cos((2\pi f_1 t) + (2\pi f_2 t - \phi))]$

$(2\pi f_1 t) - (2\pi f_2 t - \phi) = 2\pi(f_1 - f_2)t + \phi$

$(2\pi f_1 t) + (2\pi f_2 t - \phi) = 2\pi(f_1 + f_2) - \phi$

Thus,

$$P = 8[\cos(2\pi(f_1 - f_2)t + \phi) + \cos(2\pi(f_1 + f_2)t - \phi)]$$

P is a sum of two cosines:

- $f_+ = f_1 + f_2$ and $f_- = f_1 - f_2$
- phase shifts give a offset of 2ϕ

3b) If $\phi = 0$ and $f_1 = f_2$, P becomes:

$$P = 8[\cos(2\pi(f_1 - f_2)t) + \cos(2\pi(f_1 + f_2)t)]$$

Since $f_1 = f_2$:

$f_1 - f_2 = 0 \quad , \quad f_1 + f_2 = 2f_1$

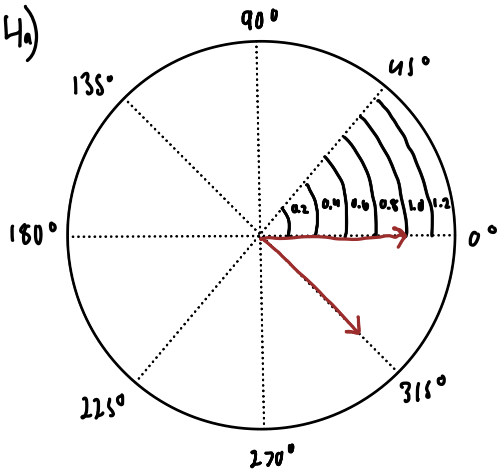
So:

$$P = 8[\cos(0) + \cos(2\pi(2f_1)t)]$$

$\cos(0) = 1 \quad , \quad P = 8[1 + \cos(4\pi f_1 t)]$

$P_{max} = 8 \cdot (1+1) = 16$

4a)



$v2(t) = 1$

$v1(t) = exp(-j\psi)$

4b) $z = e^{j0} + e^{-j\phi} = 1 + e^{-j\phi}$

$e^{-j\phi} = \cos(\phi) - j\sin(\phi)$

$z = 1 + \cos(\phi) - j\sin(\phi)$

magnitude $= |z| = \sqrt{(1+\cos(\phi))^2 + (-\sin(\phi))^2}$

$\angle z = \tan^{-1}\left(\frac{-\sin(\phi)}{1+\cos(\phi)}\right)$

For 45°:

- magnitude of the resultant signal is approximately 1.85

- the phase angle is -22.5°

4c) when $\phi = 0°$:

$e^{-j\phi} = 1$, so $z = 1+1 = 2$

magnitude $= 2$

phase $= 0$

when $\phi = 180°$:

$e^{-j\phi} = 1$, so $z = 1-1 = 0$

magnitude $= 2$

phase = undefined (zero amplitude)

3 Probability and Statistics, Noise

1(a)

This distribution is a bell-shaped curve, symmetric around x=0. The Octave code generates 10,000 random samples using the randn function, which also follows a standard normal distribution. The histogram closely matches the Gaussian curve, confirming that the samples generated by randn follow the expected probability density function. The histogram bars approximate the values of p(x) for different bins, while the Gaussian curve provides the theoretical continuous distribution.

Octave code:

```
% gaussian curve (Eq. 6)
x = linspace(-4, 4, 1000);  % range of x values
sigma = 1;  % stdev
mu = 0;  % mean
p_x = (1 / sqrt(2 * pi * sigma^2)) * exp(-((x - mu).^2) / (2 * sigma^2));

% generate random samples
samples = randn(10000, 1);  % random samples from standard normal distribution

% plot gaussian curve
figure(1);
plot(x, p_x, 'LineWidth', 2);
hold on;

% plot histogram of samples
histogram(samples, 30, 'Normalization', 'pdf', 'FaceAlpha', 0.6);
title('Gaussian Distribution vs Histogram of Samples');
xlabel('x');
ylabel('Probability Density');
legend('Gaussian Distribution (Eq. 6)', 'Histogram of Samples');
grid on;
```

1(c)

Octave code:

```
% parameters for sine wave
```

```
fs = 1000;  % sampling frequency
t = linspace(0, 1, fs);  % time vector (1 second duration)
A = 1;  % amplitude of sine wave
f = 5;  % frequency of sine wave in Hz

% generate sine wave
sine_wave = A * sin(2 * pi * f * t);

% add gaussian noise
sigma = 0.5;  % standard deviation of noise
noise = normrnd(0, sigma, size(t));  % gaussian noise with mean 0 and std 0.5
noisy_signal = sine_wave + noise;

% plot sine wave and noisy signal
figure(3);
plot(t, sine_wave, 'LineWidth', 2);
hold on;
plot(t, noisy_signal, 'LineWidth', 1);
title('Sine Wave with Added Gaussian Noise');
xlabel('Time (s)');
ylabel('Amplitude');
legend('Original Sine Wave', 'Sine Wave with Gaussian Noise');
grid on;
```

(2)

Octave Code:

```
% parameters
num_samples = 10000;  % number of random numbers
num_sums = 12;  % number of uniform random numbers to sum

% generate random numbers and compute sums
uniform_random_numbers = rand(num_samples, num_sums);  % uniform random numbers
sums = sum(uniform_random_numbers, 2);  % the sum across rows

% normalize the sums to create a standard gaussian
sums = (sums - mean(sums)) / std(sums);

% plot histogram of normalized sums
```

```
figure(1);
histogram(sums, 30, 'Normalization', 'pdf', 'FaceAlpha', 0.6);
hold on;

% plot theoretical gaussian curve for comparison
x = linspace(-4, 4, 1000);
p_x = (1 / sqrt(2 * pi)) * exp(-x.^2 / 2);
plot(x, p_x, 'LineWidth', 2);

% labels and title
title('Gaussian Distribution from Summed Uniform Random Numbers');
xlabel('Value');
ylabel('Probability Density');
legend('Summed Uniform Numbers', 'Gaussian Curve');
grid on;
```

4 ADC and DAC

```
% parameters
f = 5;  % signal frequency in hz
T = 1;  % duration of the signal in seconds
fs1 = 50;  % high sampling frequency (10 times signal frequency)
fs2 = 10;  % sampling frequency equal to 2 times signal frequency
fs3 = 7;   % sampling frequency less than 2 times signal frequency

% generate time vectors for different sampling frequencies
t1 = 0:1/fs1:T;  % high sampling rate
t2 = 0:1/fs2:T;  % nyquist rate
t3 = 0:1/fs3:T;  % below nyquist rate

% generate sine waves
sine_wave1 = sin(2 * pi * f * t1);
sine_wave2 = sin(2 * pi * f * t2);
sine_wave3 = sin(2 * pi * f * t3);

% plot sine waves
figure(1);
subplot(3, 1, 1);
stem(t1, sine_wave1, 'filled');
```

```
title('high sampling rate (fs >> 2f)');
xlabel('time (s)');
ylabel('amplitude');
grid on;

subplot(3, 1, 2);
stem(t2, sine_wave2, 'filled');
title('nyquist sampling rate (fs = 2f)');
xlabel('time (s)');
ylabel('amplitude');
grid on;

subplot(3, 1, 3);
stem(t3, sine_wave3, 'filled');
title('below nyquist sampling rate (fs < 2f)');
xlabel('time (s)');
ylabel('amplitude');
grid on;
```

When the sampling frequency is much higher than twice the signal frequency (fs >> 2f), the signal graph accurately represents the sine wave with smooth transitions between the sampled points, allowing for precise reconstruction. At the Nyquist rate (fs = 2f), the signal is still well sampled, but the number of points is minimal, which leads to a less smooth but still accurate representation. However, when the sampling frequency is below the Nyquist rate (fs < 2f), aliasing occurs, causing the signal graph to appear distorted. This distortion results in a misrepresentation of the original sine wave, making accurate reconstruction impossible.