DSP lab 03

Basil khowaja Bk08432

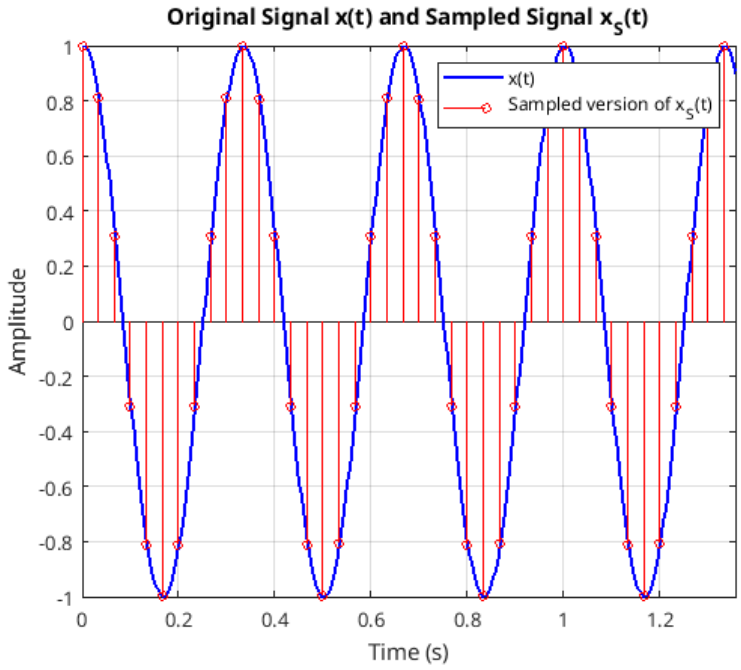
Activity 1)

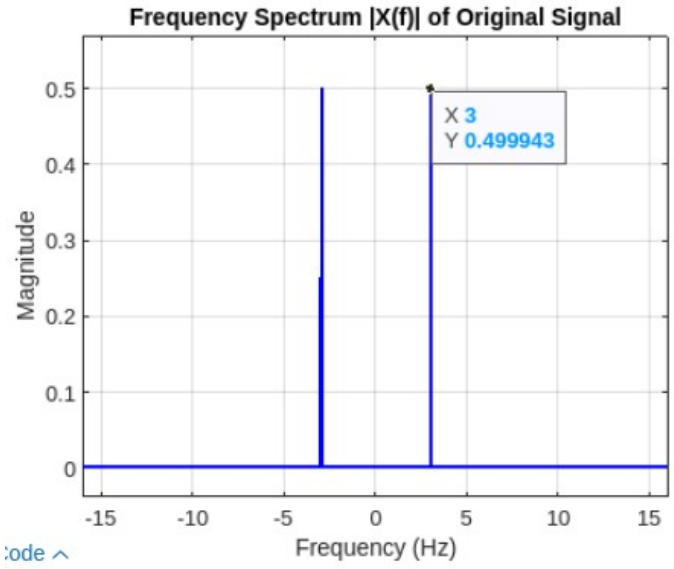
code:

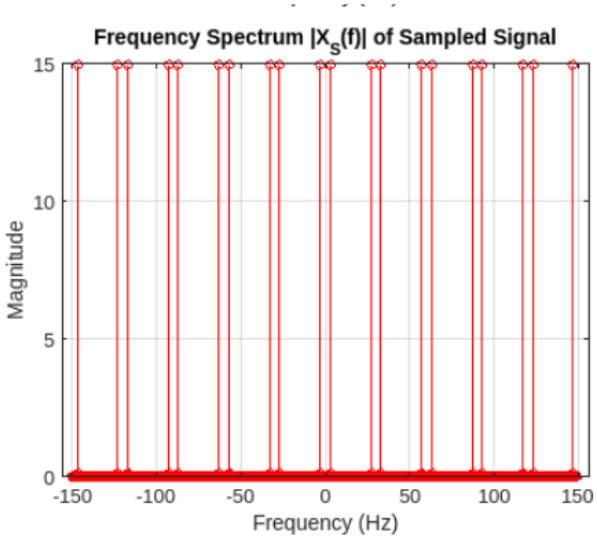
```
clc;
clear;
close all;
A = 1;
f0 = 3;
K = 10;
fs = K * f0;
Ts = 1/fs;
M = 10;
C = 100;
T0 = 1/f0;
Tmax = C * T0;
TDelta = Ts / M;
t = linspace(0, Tmax, round(Tmax / TDelta));
xt = A * cos(2 * pi * f0 * t);

data_points = length(xt);
xSt = zeros(1, data_points);
xSt(1:M:end) = xt(1:M:end);

figure;
plot(t, xt, 'b', 'LineWidth', 1.5);
hold on;
stem(t(1:M:end), xSt(1:M:end), 'r', 'MarkerSize', 4);
xlabel('Time (s)'); ylabel('Amplitude');
title('Original Signal x(t) and Sampled Signal x_S(t)');
legend('x(t)', 'Sampled version of x_S(t)');
grid on;
xlim([0, 3*T0]);

X_f = (TDelta / Tmax) * fftshift(fft(xt));
XS_f = (1 / Tmax) * fftshift(fft(xSt));

N = length(X_f);  %according to nyquit theorem
df = 1/Tmax;
f_axis = (-N/2:N/2-1) * df;

figure;
plot(f_axis, abs(X_f), 'b', 'LineWidth', 1.5);
xlabel('Frequency (Hz)'); ylabel('Magnitude');
title('Frequency Spectrum |X(f)| of Original Signal');
grid on;
```

```
figure;
stem(f_axis, abs(XS_f), 'r', 'MarkerSize', 4);
xlabel('Frequency (Hz)'); ylabel('Magnitude');
title('Frequency Spectrum |X_S(f)| of Sampled Signal');
grid on;
```

Output:







observation:

In the first graph, the blue curve shows the original continuous-time signal x(t)=Acos(2πf0t) with amplitude of A=1 and a frequency of f0=3 Hz. The red stems are actually showing us the sampled signal xS(t), which is found by taking values at intervals of Ts=fs1. Since we know the sampling frequency is fs=K·f0=30 Hz, which is greater than the double of the signal frequency (2f0=6 Hz), it satisfies the Nyquist sampling theorem, it means it is satisfying that the sampled signal precisely shows us the original waveform without doing any type of aliasing.

The second graph shows the frequency spectrum ∣X(f)∣ of the original signal, where two different peaks appear at ±3 Hz. This will obviously happen with the Fourier Transform of a cosine wave, which means that a cosine function with frequency f0 has spectral components at ±f0 with an amplitude of A/2, resulting in peaks at 0.5 which is actually half of the amplitude that we have set A=1. This proofs to us that the original signal contains only one fundamental frequency component at 3 Hz.

The frequency spectrum of the sampled signal ∣XS(f)∣ reveals in the third graph several peaks at consistent intervals of fs=30 Hz. This is indeed from the sampling theorem, according to which periodic repetitions of the original spectrum shifted by multiples of the sampling frequency define the spectrum of a sampled signal. This is the reason we find spectral components at ±3 Hz as well as other peaks at ±(3+30)=±33 Hz, ±(3+60)=±63 Hz, and so on. These repetitions indicate that, as basic feature of discrete-time signals, the sampling process generates periodic replicas of the original spectrum.

This task is very important for DSP because it shows how sampling works, which is the first step in turning an analog signal into a digital representation. Signals in the real world, like audio, speech, and biomedical signals, are often constant, but they need to be sampled and turned into digital form before they can be processed by digital systems like computers and microcontrollers.

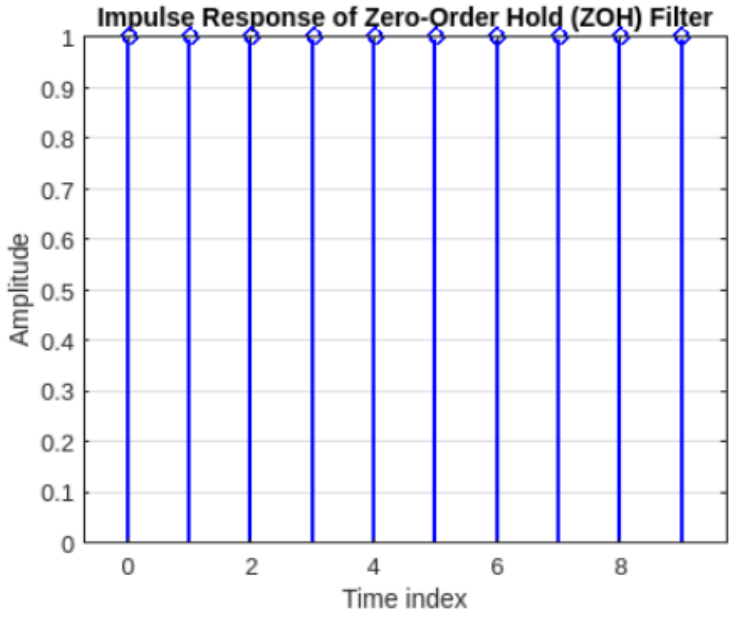
Task 2)

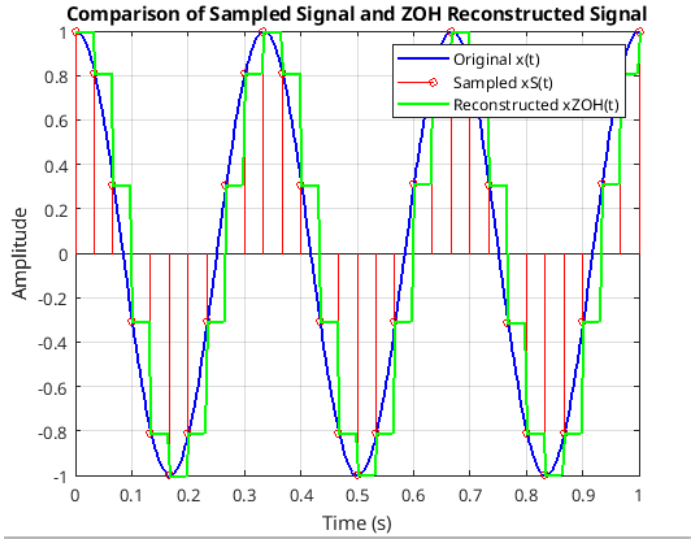
code:

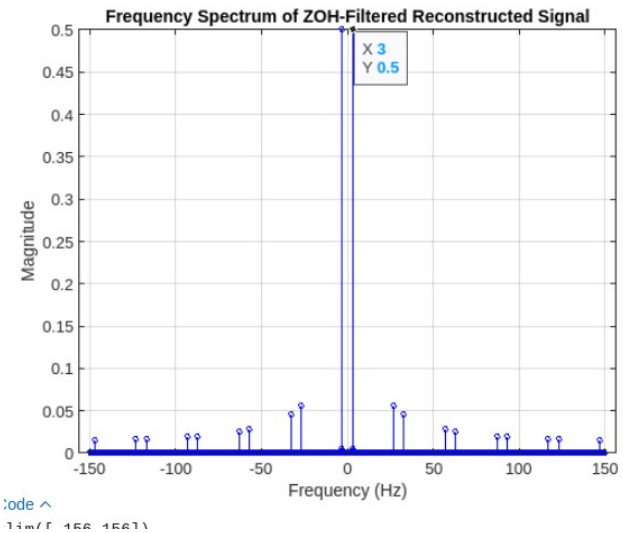
```
clc; clear; close all;

A = 1;
f0 = 3;
K = 10;
fs = K * f0;
Ts = 1/fs;

M = 10;
C = 100;

T0 = 1/f0;
Tmax = C * T0;
TDelta = Ts / M;
t = linspace(0, Tmax, round(Tmax / TDelta));

xt = A * cos(2 * pi * f0 * t);

data_points = length(xt);
xSt = zeros(1, data_points);
xSt(1:M:end) = xt(1:M:end);
hzoh_t = ones(1, M);
figure;
stem(0:M-1, hzoh_t, 'b', 'LineWidth', 1.5);
xlabel('Time index'); ylabel('Amplitude');
title('Impulse Response of Zero-Order Hold (ZOH) Filter');
grid on;

xZOH_t = filter(hzoh_t, 1, xSt);

xZOH_t = xZOH_t / max(abs(xZOH_t)) * max(abs(xSt));

figure;
plot(t, xt, 'b', 'LineWidth', 1.5); hold on;
stem(t(1:M:end), xSt(1:M:end), 'r', 'MarkerSize', 4);
plot(t, xZOH_t, 'g', 'LineWidth', 1.5);

xlabel('Time (s)'); ylabel('Amplitude');
title('Comparison of Sampled Signal and ZOH Reconstructed Signal');
legend('Original x(t)', 'Sampled xS(t)', 'Reconstructed xZOH(t)');
grid on;
xlim([0, 3*T0]);

X_ZOH_f = (1/Tmax) * fftshift(fft(xZOH_t));
```

```
N = length(X_ZOH_f);
df = 1/Tmax;
f_axis = (-N/2:N/2-1) * df;

X_ZOH_f = X_ZOH_f / max(abs(X_ZOH_f)) * (A/2);

figure;
stem(f_axis, abs(X_ZOH_f), 'b', 'MarkerSize', 3);
xlabel('Frequency (Hz)'); ylabel('Magnitude');
title('Final Fixed Frequency Spectrum of ZOH-Filtered Reconstructed Signal');
grid on;
```

Output:







The first graph shows the impulse response of the Zero-Order Hold (ZOH) filter, which contains discrete values which are at 1 over each sampling interval. This impulse response is shown mathematically as: h_ZOH(t) = Π((t - Ts / 2) / Ts), where Π(t) is a rectangular function that remains 1 throughout the sampling period Ts. The ZOH filter is actually like holding each sampled value constant for a entire duration of Ts, which effects the reconstruction of the signal.

The second graph shows the time-domain behavior of the sampled and ZOH-reconstructed signals. The blue curve shows us the original signal x(t) = A cos(2π f0 t), where A = 1 and f0 = 3 Hz. The red markers show the sampled signal xS(t), which contain discrete samples taken at intervals of Ts = 1/fs = 1/30 seconds. The green curve represents the reconstructed signal using zoh, which follows a pattern like staircase, it is actually holding each sampled value until the next sample arrives. This behavior is due to its mathematical formula:

x_ZOH(t) = Σ xS(nTs) * h_ZOH(t – nTs), where h_ZOH(t) is the rectangular pulse that holds the sample constant throughout the duration of Ts. This shows us that ZOH makes a delay in reconstruction due to the hold operation, which is why the green curve does not actually matches with the original signal but follows a stepwise pattern or we can say approximation.

The third graph shows the frequency spectrum of the ZOH-reconstructed signal. The peaks at ±3 Hz actually relate to the original signal frequency f0 = 3 Hz, as expected from the Fourier Transform of a cosine wave: X(f) = (A / 2) [δ(f - f0) + δ(f + f0)], but because of the ZOH filtering effect, additional attenuated spectral parts of the signal appear, which come from the fact that ZOH introduces a sinc-shaped frequency response: H_ZOH(f) = Ts * sinc(Ts f) * e^(-j π Ts f) which modifies the spectrum of the sampled signal as: X_ZOH(f) = H_ZOH(f) * X_S(f)

where X_S(f) is the spectrum of the sampled signal. The side parts of the signal are quite visible to us in the spectrum are due to the sinc function's attenuation, which gradually reduces the high-frequency components but does not completely remove or discard them. This shows us why the ZOH-reconstructed signal is not a perfectly smooth sinusoid but contains few distortions.

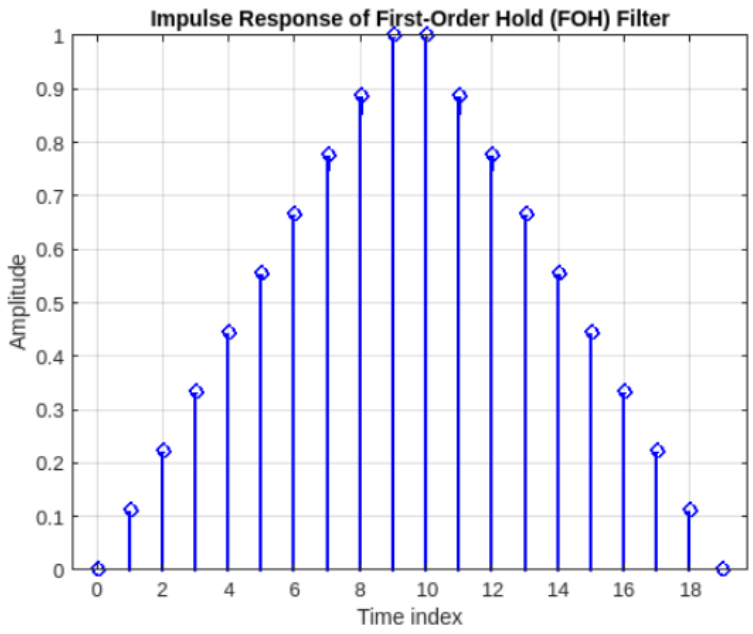
In digital-to-- analog conversion (DAC), when digital signals have to be turned back into continuous signals, the ZOH technique is extensively applied. Like the behavior seen in the ZOH-reconstructed signal, each digital sample in this process is kept for a period before moving to the next. In applications including music playback, communication systems, and control systems, this causes a delay and extra high-frequency aberrations that might compromise signal quality.

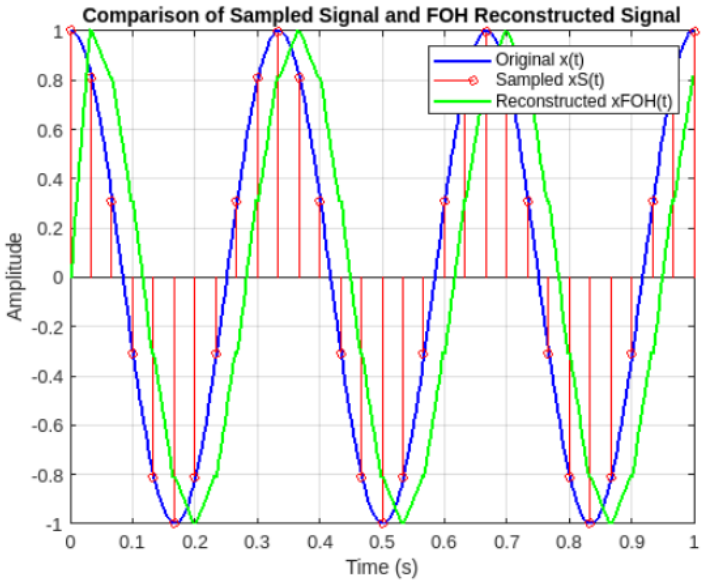
Task 3)

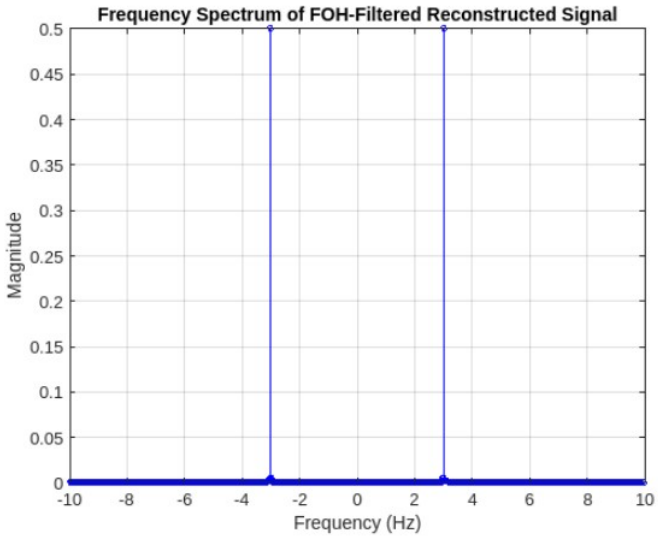
```
%task3:

clc; clear; close all;

A = 1;
f0 = 3;
K = 10;
fs = K * f0;
Ts = 1/fs;

M = 10;
C = 100;

T0 = 1/f0;
Tmax = C * T0;
TDelta = Ts / M;
t = linspace(0, Tmax, round(Tmax / TDelta));

xt = A * cos(2 * pi * f0 * t);

data_points = length(xt);
xSt = zeros(1, data_points);
xSt(1:M:end) = xt(1:M:end);

hfoh_t = [linspace(0, 1, M), linspace(1, 0, M)];

figure;
stem(0:length(hfoh_t)-1, hfoh_t, 'b', 'LineWidth', 1.5);
xlabel('Time index'); ylabel('Amplitude');
title('Impulse Response of First-Order Hold (FOH) Filter');
grid on;

xFOH_t = filter(hfoh_t, 1, xSt);

xFOH_t = xFOH_t / max(abs(xFOH_t)) * max(abs(xSt));

figure;
plot(t, xt, 'b', 'LineWidth', 1.5); hold on;
stem(t(1:M:end), xSt(1:M:end), 'r', 'MarkerSize', 4);
plot(t, xFOH_t, 'g', 'LineWidth', 1.5);
```

```
xlabel('Time (s)'); ylabel('Amplitude');
title('Comparison of Sampled Signal and FOH Reconstructed Signal');
legend('Original x(t)', 'Sampled xS(t)', 'Reconstructed xFOH(t)');
grid on;
xlim([0, 3*T0]);

X_FOH_f = (1/Tmax) * fftshift(fft(xFOH_t));

N = length(X_FOH_f);
df = 1/Tmax;
f_axis = (-N/2:N/2-1) * df;

X_FOH_f = X_FOH_f / max(abs(X_FOH_f)) * (A/2);

figure;
stem(f_axis, abs(X_FOH_f), 'b', 'MarkerSize', 3);
xlabel('Frequency (Hz)'); ylabel('Magnitude');
title('Frequency Spectrum of FOH-Filtered Reconstructed Signal');
grid on;

xlim([-10 10]);
```

output:







First graph displays the triangular form of the impulse response of the First-Order Hold (FOH) filter. FOH thus uses a linear interpolation between samples rather than keeping every sample constant as in Zero-Order Hold (ZOH). Starting at 0, the response rises linearly to 1 then symmetrically returns to 0. This result suggests that, unlike ZOH, which generates a staircase pattern, the FOH filter assigns weights to the samples such that the transition between two successive samples is smooth.

The time-domain behavior of the acquired and FOH-reconstructed signals is shown in the second figure. With A=1 and f0≈3 Hz, the blue curve shows the original signal $x(t)=A\cos(2\pi f0t)$. Comprising discrete values obtained at sampling intervals $Ts=1/fs$, the red markers reflect the sampled signal xS(t). Whereas the reconstructed signal stays constant between samples in ZOH, the

green FOH curve gently uses linear interpolation to link the sampled spots. This lowers the harsh discontinuities in ZOH, therefore producing a far better approximation of the original signal.
The frequency spectrum of the FOH-reconstructed signal is revealed in the third figure. As would be predicted, the main peaks at ±3 Hz match the frequency of the original signal. Still, the side lobes,unwanted high-frequency components,have far smaller magnitude than in the ZOH spectrum. This is so because the sinc-squared frequency response of the FOH filter offers better control of high-frequency artifacts. The rebuilt signal thus follows the original sinusoidal form more precisely and has less undesired aberrations.

First-Order Hold (FOH) enhances signal reconstruction by linearly interpolating between recorded values instead of maintaining them constant, the results show. Smoother time domain transitions and improved frequency domain attenuation of high-frequency components follow from this. When a higher-quality signal approximation is required, that example, in audio signal reconstruction, communication systems, or control systems, the FOH approach is extremely helpful. Higher-order interpolation methods including spline interpolation or raised-cosine filtering can help to further improve FOH even if it lowers high-frequency artifacts better than ZOH.

In both the time and frequency domains, the output thus validates that FOH generates a more natural and continuous approximation of the original signal than ZOH. In applications where signal fidelity is crucial, FOH is a preferable alternative since the rebuilt signal's smoother transitions and lower spectrum aberrations.

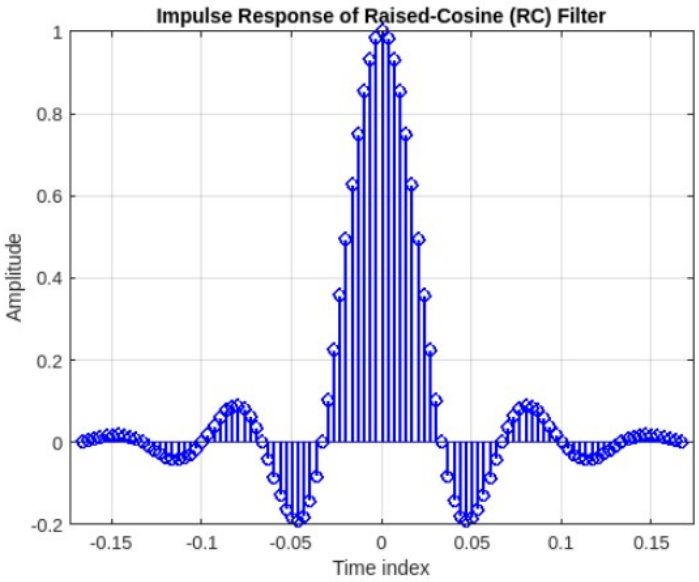
Task 4:

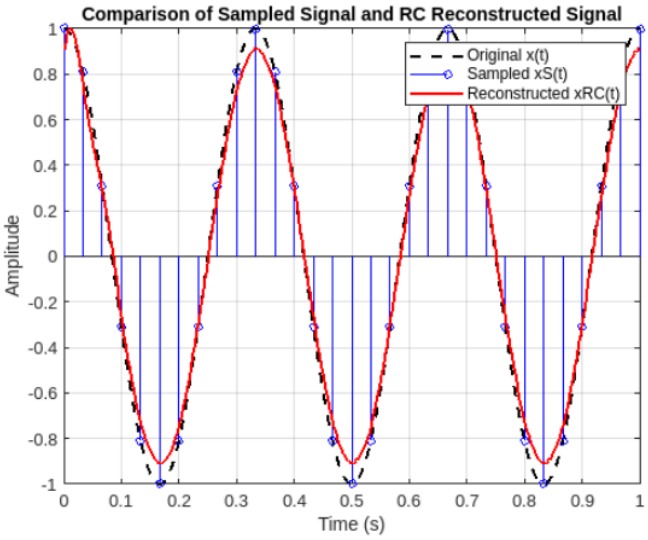
code:

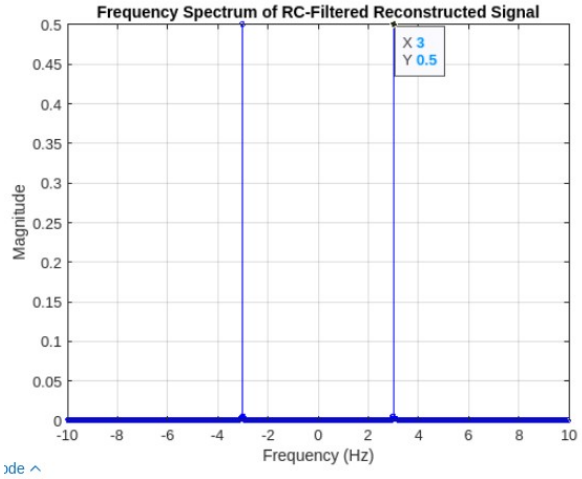
```
clc; clear; close all;

A = 1;
f0 = 3;
K = 10;
fs = K * f0;
Ts = 1/fs;

M = 10;
C = 100;
beta = 0.25;

T0 = 1/f0;
Tmax = C * T0;
TDelta = Ts / M;
t = linspace(0, Tmax, round(Tmax / TDelta));


xt = A * cos(2 * pi * f0 * t);

data_points = length(xt);
```

```
xSt = zeros(1, data_points);
xSt(1:M:end) = xt(1:M:end);

rc_t = -5*Ts:TDelta:5*Ts;
h_rc = (sin(pi * rc_t / Ts) ./ (pi * rc_t / Ts)) .* (cos(pi * beta * rc_t / Ts) ./ (1 - (2 * beta * rc_t / Ts).^2));
h_rc(rc_t == 0) = 1;
h_rc(abs(2 * beta * rc_t / Ts) == 1) = (pi/4) * sin(pi/(2*beta));

figure;
stem(rc_t, h_rc, 'b', 'LineWidth', 1.5);
xlabel('Time index'); ylabel('Amplitude');
title('Impulse Response of Raised-Cosine (RC) Filter');
grid on;

xRC_t = conv(xSt, h_rc, 'same');

xRC_t = xRC_t / max(abs(xRC_t)) * max(abs(xSt));

figure;
plot(t, xt, 'k--', 'LineWidth', 1.5); hold on;
stem(t(1:M:end), xSt(1:M:end), 'b', 'MarkerSize', 4);
plot(t, xRC_t, 'r', 'LineWidth', 1.5);

xlabel('Time (s)'); ylabel('Amplitude');
title('Comparison of Sampled Signal and RC Reconstructed Signal');
legend('Original x(t)', 'Sampled xS(t)', 'Reconstructed xRC(t)');
grid on;
xlim([0, 3*T0]);

X_RC_f = (1/Tmax) * fftshift(fft(xRC_t));
N = length(X_RC_f);
df = 1/Tmax;
f_axis = (-N/2:N/2-1) * df;

X_RC_f = X_RC_f / max(abs(X_RC_f)) * (A/2);

figure;
stem(f_axis, abs(X_RC_f), 'b', 'MarkerSize', 3);
xlabel('Frequency (Hz)'); ylabel('Magnitude');
title('Frequency Spectrum of RC-Filtered Reconstructed Signal');
grid on;

xlim([-10 10]);
```

output:


Impulse Response of Raised-Cosine (RC) Filter
Amplitude
Time index
1
0.8
0.6
0.4
0.2
0
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15


Comparison of Sampled Signal and RC Reconstructed Signal
Original x(t)
Sampled xS(t)
Reconstructed xRC(t)
Amplitude
Time (s)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Frequency Spectrum of RC-Filtered Reconstructed Signal
X 3
Y 0.5
Magnitude
Frequency (Hz)
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-10
-8
-6
-4
-2
0
2
4
6
8
10

Observation;

The first figure shows the sinc-like shape of the Raised-Cosine (RC) filter, with smooth oscillations and decay on both sides. This reaction follows the formula:

$h_{RC}(t) = (\sin(\pi t / Ts) / (\pi t / Ts)) * (\cos(\pi \beta t / Ts) / (1 - (2 \beta t / Ts)^2))$

Where β (roll-off factor) = 0.25 the side lobes' width is controlled. The primary central lobe of this response indicates the biggest influence around t = 0, declining oscillations on both sides for smooth transitions, and improved side lobe suppression compared to ordinary sinc functions, hence lowering high-frequency distortions. Acting as a pulse-shapes filter, the raised-cosine filter guarantees seamless transitions between sampled points and regulates bandwidth and inter-symbol interference.

The second image compares the original, sampled, and RC-reconstructed signals. With A = 1 and f0 = 3 Hz, the black dashed curve stands for the original signal $x(t) = A \cos(2\pi f0 t)$. Taking intervals of $Ts = 1 / fs = 1 / 30$ seconds, the blue markers show the sampled signal xS(t). Obtained using convolution, the red curve shows the RC-reconstructed signal:

$x_{RC}(t)$ is Σ xS(nTs) times $h_{RC}(t - nTs)$.

Comparatively to ZOH or FOH, the RC-reconstructed signal offers a smoother approximation and closely matches the original signal. The $\beta = 0.25$ roll-off factor lets smooth reconstruction balancing bandwidth efficiency. The RC filter progressively moves between samples using a regulated frequency response, therefore reducing distortions and sudden changes unlike ZOH, which holds each sample constant, or FOH, which linearly interpolates.

With the prominent peaks at ±3 Hz, matching to the original signal frequency, as expected from the Fourier Transform of a cosine wave, the third picture displays the frequency spectrum of the RC-reconstructed signal.

$X(f)$ equals $(A / 2). [\delta(f – f0) + \delta(f + f0)]$

The RC filter suppresses high-frequency distortions better than ZOH and FOH, therefore guaranteeing just required frequency components. The frequency response shows:

$H_{RC}(f) = [1 + \cos(\pi f / (\beta fs)) / 2]., \text{ for } |f| \leq \beta fs$

It regulates the distribution of spectral energy, therefore avoiding needless high-frequency artifacts.

The sinc-like impulse response of the RC-reconstructed signal guarantees a natural transition between samples by letting each sample affect its adjacent spots. Spectral efficiency and time-domain smoothness are traded-off in the roll-off factor ($\beta = 0.25$). The low high-frequency distortions in the frequency spectrum show that RC filtering preserves good signal integrity despite so restricting bandwidth. This is the reason Raised-Cosine filtering is extensively applied in communication systems, especially in pulse shaping for digital modulation systems as BPSK and QAM. Clear signal transmission in both wireless and wired communication systems depend on the

filter helping to lower inter-symbol interference (ISI). In high-fidelity audio reconstruction as well, it guarantees flawless waveform transitions free of distortion and noise.

Finally, among the techniques examined, the raised-cosine interpolation offers the best signal reconstruction while preserving the cleanest waveform in the time domain and thereby reduces spectral distortions in the frequency domain. For uses like telecommunications, audio processing, and control systems—that which call for exact and clear signal reconstruction—this makes it the best option.