Task_sampling

July 1, 2025

1 Task_sampling

1.1 Sampling and reconstruction task description

Variant 6. Investigate aliasing for a sine wave with $f = 30$ Hz, sampled at $f_s = 40$ Hz.

1.2 Python code (Sampling and reconstruction)

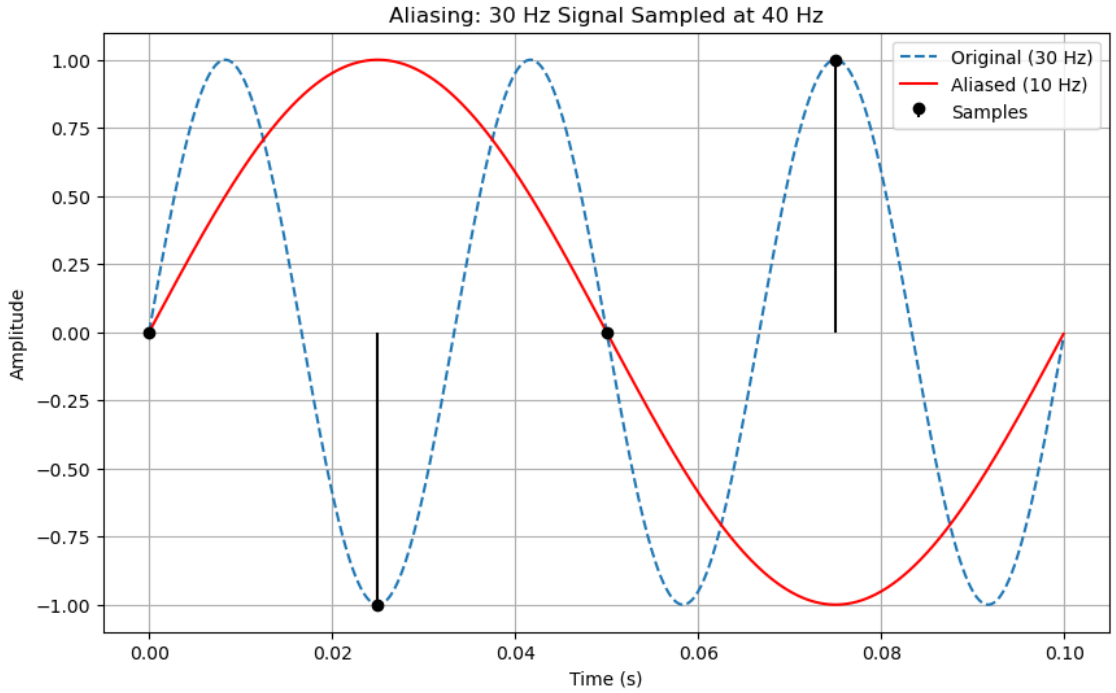
[14]:
```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
f_signal = 30  # Original frequency (Hz)
f_sample = 40  # Sampling frequency (Hz)
t = np.linspace(0, 0.1, 1000, endpoint=False)  # High-resolution time vector

# Signals
original_signal = np.sin(2 * np.pi * f_signal * t)
aliased_signal = np.sin(2 * np.pi * 10 * t)  # Expected aliased signal

# Sampling points
t_sampled = np.arange(0, 0.1, 1 / f_sample)
samples = np.sin(2 * np.pi * f_signal * t_sampled)

# Plot
plt.figure(figsize=(10, 6))
plt.plot(t, original_signal, label='Original (30 Hz)', linestyle='--')
plt.plot(t, aliased_signal, label='Aliased (10 Hz)', color='red')
plt.stem(t_sampled, samples, linefmt='k-', markerfmt='ko', basefmt=" ",␣
 ↪label='Samples')
plt.title('Aliasing: 30 Hz Signal Sampled at 40 Hz')
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.legend()
plt.grid()
plt.show()
```



1.3 Coding/Decoding task description

Variant 6. Solve Problem 2: Quantize the signal [0.3, 0.8, 1.2, 1.9, 2.7] to 5 levels and calculate the quantization error.

1.4 Python code (Coding/Decoding)

[15]:
```python
import numpy as np

def quantize(signal, levels):
    min_val, max_val = min(signal), max(signal)
    step = (max_val - min_val) / (levels - 1)
    quantized = np.round((np.array(signal) - min_val) / step) * step + min_val
    mse = np.mean((np.array(signal) - quantized) ** 2)
    return quantized, mse

signal = [0.3, 0.8, 1.2, 1.9, 2.7]
quantized, mse = quantize(signal, 5)

print("Quantized Signal:", quantized)
print("MSE:", mse)
```

```
Quantized Signal: [0.3 0.9 0.9 2.1 2.7]
MSE: 0.027999999999999997
```