EE22BTECH11049 - Shivansh Kirar

Question EC 31 2023
The signal-to-noise ratio (SNR) of an ADC with a full-scale sinusoidal input is given to be 61.96 dB. The resolution of the ADC is GATE EC 2023
Solution:

Variable	Defination
T	Time Period
f	Frequency
A	Amplitude
Ps	Signal Power
Pe	Noise Power
q	Quantization step size
n	No of bits
e	Quantization Error

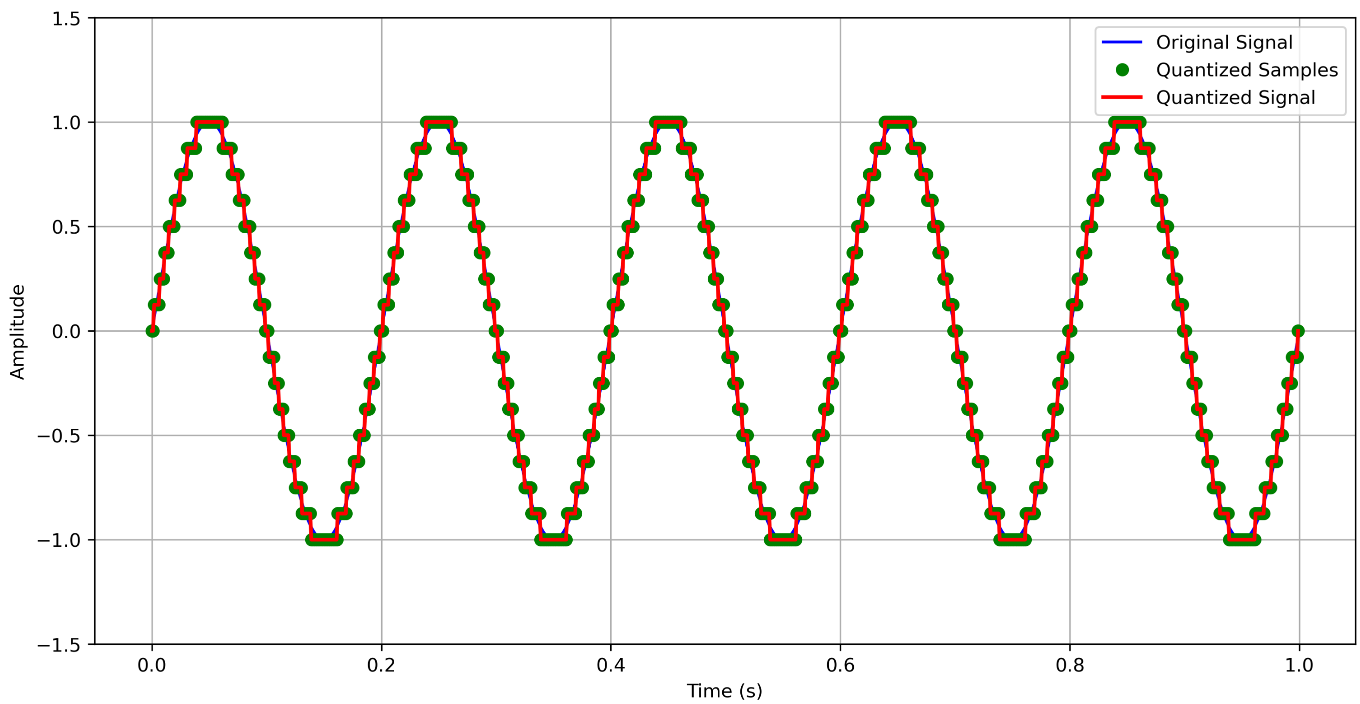

Fig. 1: Quantization of Sinusoidal Signal

1) Signal Power:
The power of a continuous-time signal is defined as the average value of the square of the signal over a certain time interval. For a sinusoidal signal $x(t) = A\sin(2\pi f t + \phi)$, the power (P_s) is calculated as:

$$P_s = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} |x(t)|^2 dt \tag{1}$$

where ϕ is the phase of the signal.

$$P_s = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} |A \sin(2\pi f t + \phi)|^2 dt \tag{2}$$

$$P_s = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} A^2 \sin^2(2\pi f t + \phi) dt \tag{3}$$

$$P_s = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} A^2 \cdot \frac{1 - \cos(4\pi f t + 2\phi)}{2} dt \tag{4}$$

$$P_s = \frac{1}{2} \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} A^2 dt - \frac{1}{2} \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} A^2 \cos(4\pi f t + 2\phi) dt \tag{5}$$

$$P_s = \frac{1}{2} \cdot A^2 - 0 \tag{6}$$

$$P_s = \frac{A^2}{2} \tag{7}$$

Here, $A = 1$, so:

$$P_s = \frac{1}{2} \tag{8}$$

2) Noise Power:
Quantized to 2^n discrete levels, equivalent to n bits, one quantization step q has a size of

$$q = \frac{2}{2^n} = 2^{-(n-1)} \tag{9}$$

We know,quantization error has a maximum value of plus or minus half the step size, so

$$|e| \le \frac{q}{2} \text{ and therefore, } |e| \le 2^{-n} \tag{10}$$

For a large enough number of quantization steps, the probability density function of the quantization error tends toward being flat 1. Pdf of error (e) of quantization is defined as

$$P_e(X) = \begin{cases} \frac{1}{q}, & \text{if } -\frac{q}{2} < X < \frac{q}{2} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

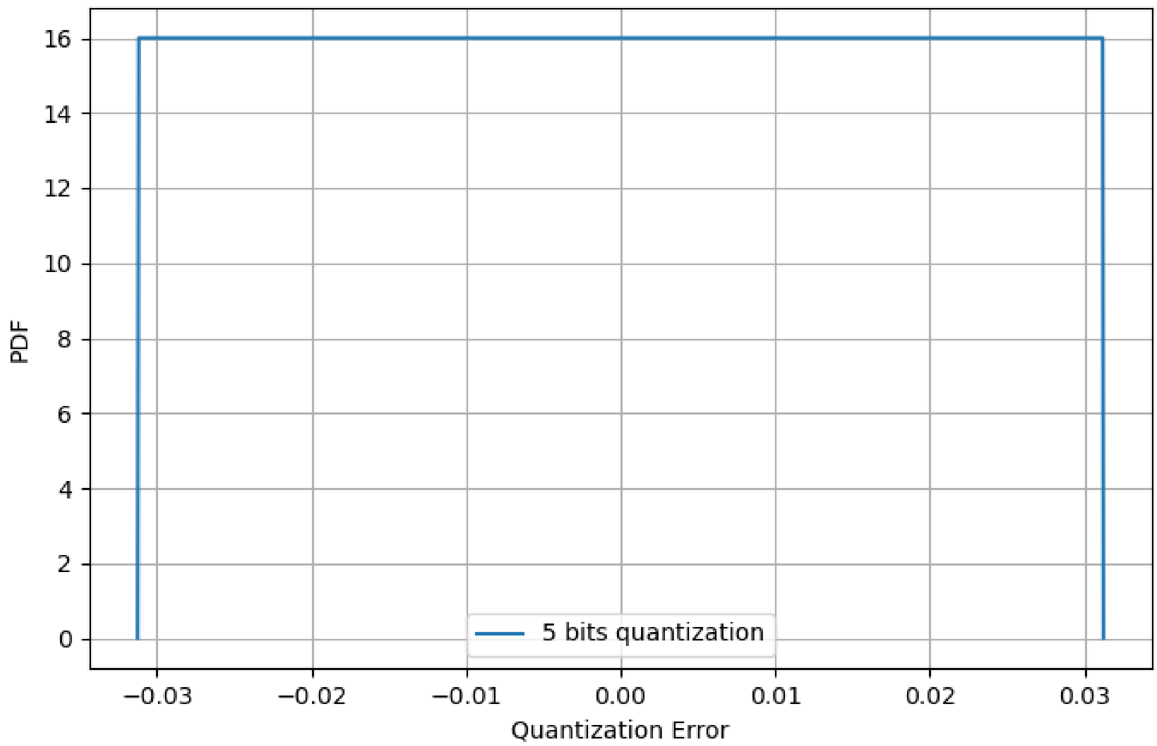

Fig. 2: plot of pdf of Quantization Error

So, we can calculate its mean power or variance as the 2nd moment of its distribution. Since, the distribution of error is uniform hence E[e]=0.

$$E[e^2] = \int_{-\frac{q}{2}}^{\frac{q}{2}} p(e)e^2\, de = \frac{1}{q} \cdot \frac{1}{3}\left(\left(\frac{q}{2}\right)^3 - \left(-\frac{q}{2}\right)^3\right) = \frac{1}{3q} \cdot \frac{q^3}{4} = \frac{q^2}{12} \tag{12}$$

On putting $q = 2^{-(n-1)}$, we have:

$$E[e^2] \approx \frac{2^{-2n}}{3} \tag{13}$$

$$Pe \approx \frac{2^{-2n}}{3} \tag{14}$$

Thus, an ideal ADC would have a signal-to-noise ratio

$$SNR = \frac{Ps}{Pe} = 1.5 \cdot 2^{2n} \tag{15}$$

or, expressed in decibels,

$$SNR = 1.76 + 6.02n \tag{16}$$

Plug in the values:

$$n = \frac{61.96 \text{ dB} - 1.76 \text{ dB}}{6.02 \text{ dB}} \approx 10 \text{ bits} \tag{17}$$

So, the resolution of the ADC is approximately 10 bits.

a) Simulation and related steps

```
import numpy as np

Given_SNR = 61.96
# Signal parameters
amplitude = 1  # Amplitude of the sinusoidal signal
frequency = 10  # Frequency of the sinusoidal signal in Hz
sampling_frequency = 1000  # Sampling frequency in Hz
duration = 1  # Duration of the signal in seconds
num_samples = int(sampling_frequency * duration)
time = np.arange(0, duration, 1/sampling_frequency)

# Generating the full-scale sinusoidal input signal
input_signal = amplitude * np.sin(2 * np.pi * frequency * time)

# Calculate Signal Power (Ps) by averaging the square of the signal
signal_power = np.mean(input_signal**2)

# Initialize the simulated resolution as None
resolution_simulated = None

# Iteratating through different quantization bit values
for n in range(1, 20):
    # Calculate the number of quantization levels
    quantization_levels = 2**n

    # Calculate the quantization step size
    quantization_step = 2 * amplitude / quantization_levels

    # Quantize the input signal
    quantized_signal = np.round(input_signal / quantization_step) * quantization_step

    # Calculate the quantization error signal
    quantization_error_signal = input_signal - quantized_signal

    # Calculate Signal Power (Pe) of quantization error signal
    signal_power_quantization_error = np.mean(quantization_error_signal**2)

    # Calculate Signal Power in decibels (dB)
    SNR_simulated = 10 * np.log10(signal_power / signal_power_quantization_error)

    # Check if the simulated SNR is close to the given SNR with a tolerance of 1 dB
    if np.isclose(Given_SNR, SNR_simulated, atol=1):
        resolution_simulated = n
        break

print(f"Simulated Resolution: {resolution_simulated} bits")
```