1. Discrete Time Signal Representation

EE 274/COE 197E

Rhandley Cajote, Ph. D.
Crisron Lucas, MSc.

Today's Lesson:

1. What are discrete time signals?
2. Examples of discrete time signals
3. Properties of discrete time signals
4. DTS representation
5. DTS classification

Discrete Time Signals

- Signals that are **sampled** in time
- Signals occurring at "points in time" → **discontinuous**
- Can be described as functions *x[n], y[n]*

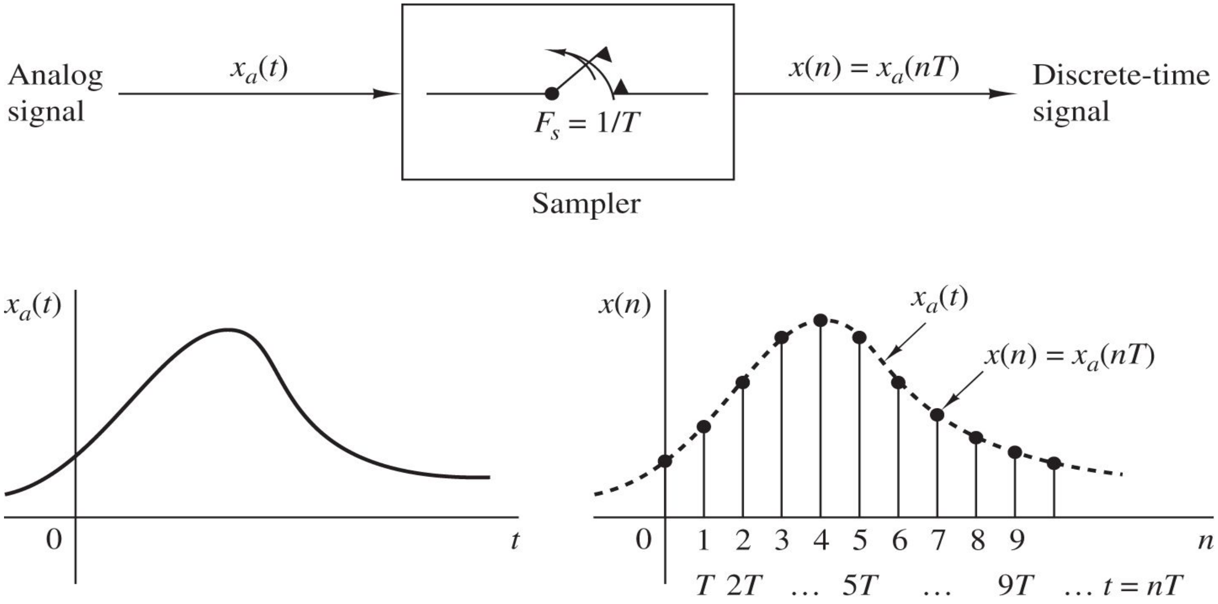


Figure 1.4.3 Periodic sampling of an analog signal.

Discrete Time Signals

- Common discrete time signals are **periodically** sampled (denoted as $x_a(nT)$)

- T is called the **sampling period**

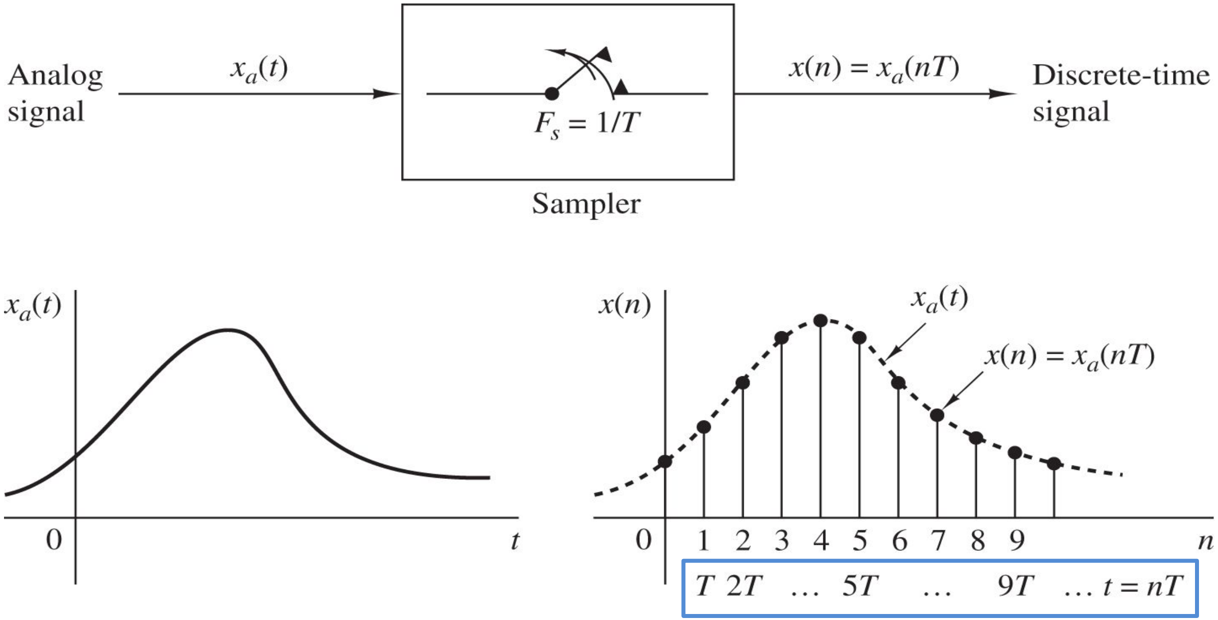
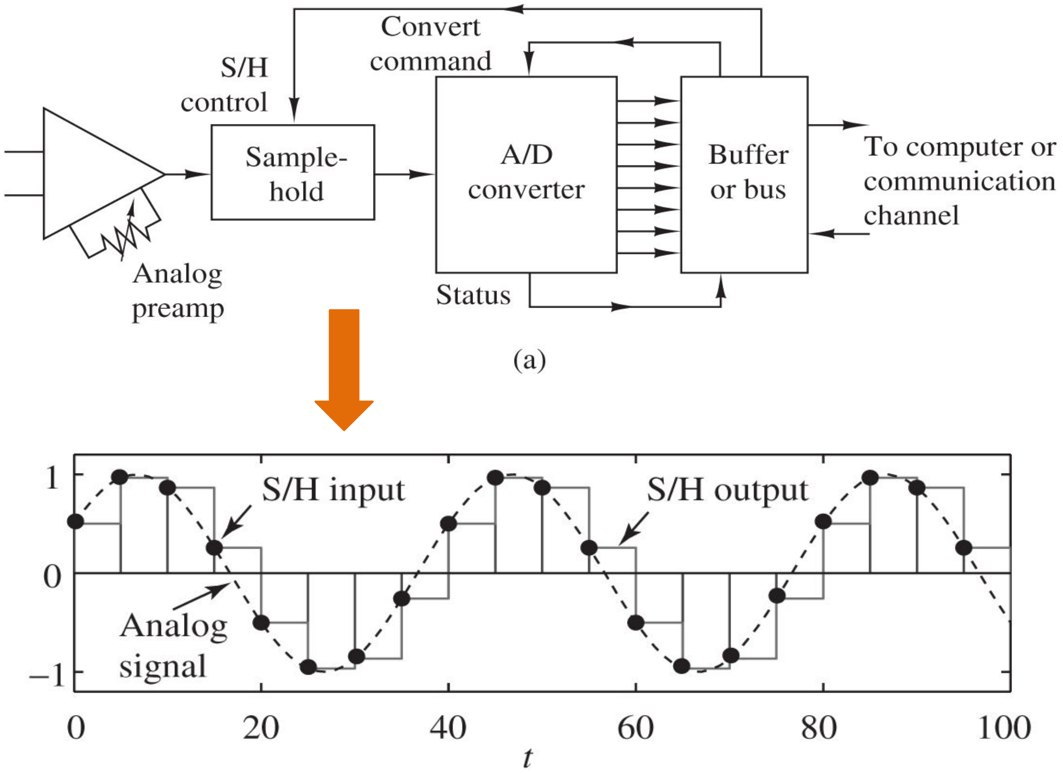


Figure 1.4.3 Periodic sampling of an analog signal.

Discrete Time Signals

- **Sample and Hold** (S/H) is the most common sampler implementation for A2D conversion

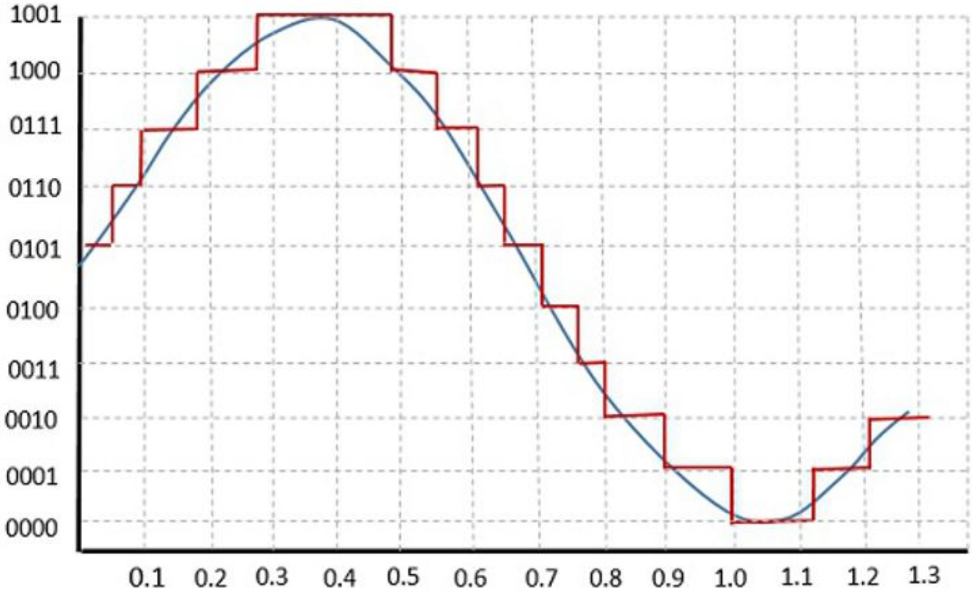
- Sampler holds the value until the next sampling period



Discrete Time Signals

- Discrete time and discrete valued signals (digital signals) are described by binary units or **bits**

- The discrete levels and bit allocation are determined by **quantization**



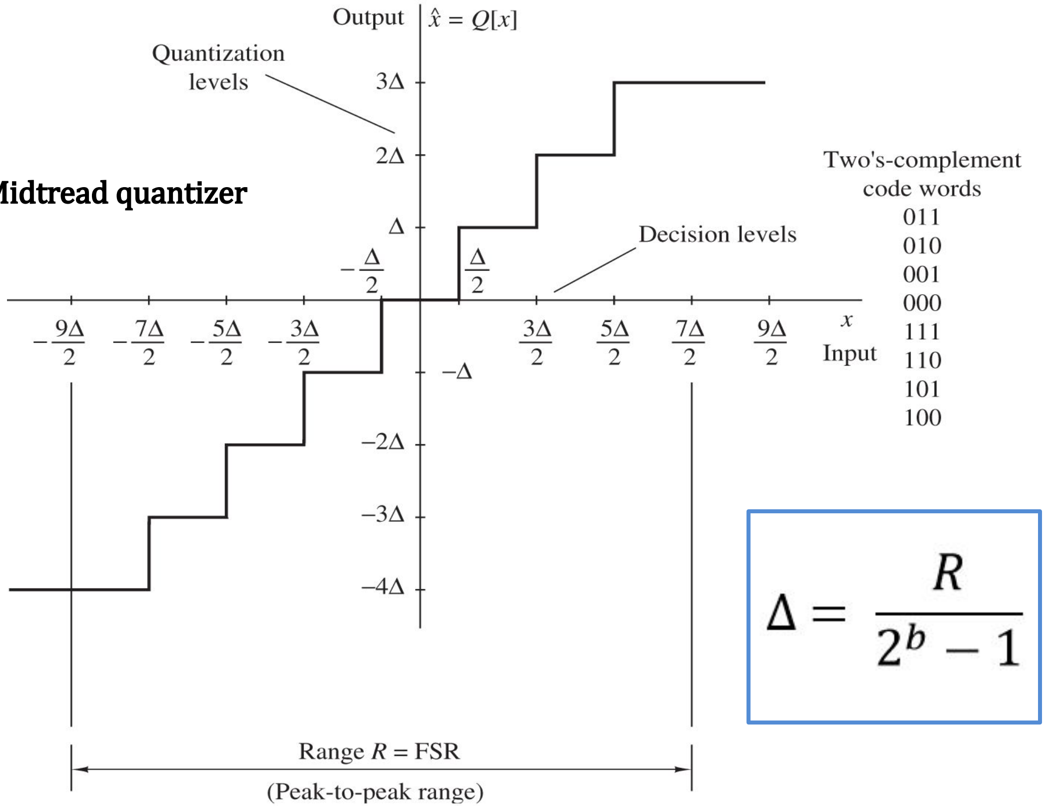
4 bit representation → 16 levels of discrete values

Discrete Time Signals

- **Quantization** is the process of mapping continuous infinite values to a smaller set of discrete finite values

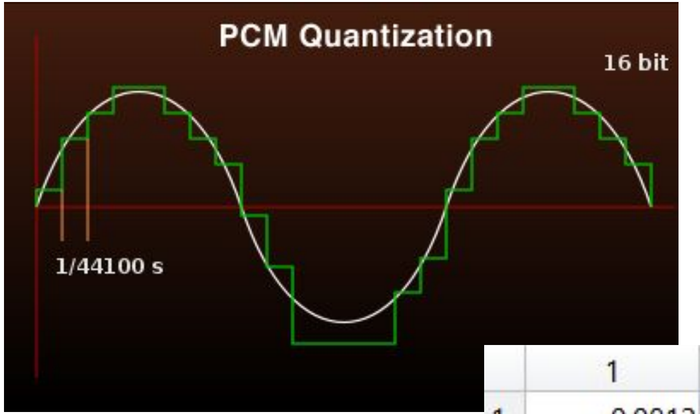
- R is the **Full-Scale Range** and b is the number of bits used **(midtread quantizer)**

Midtread quantizer

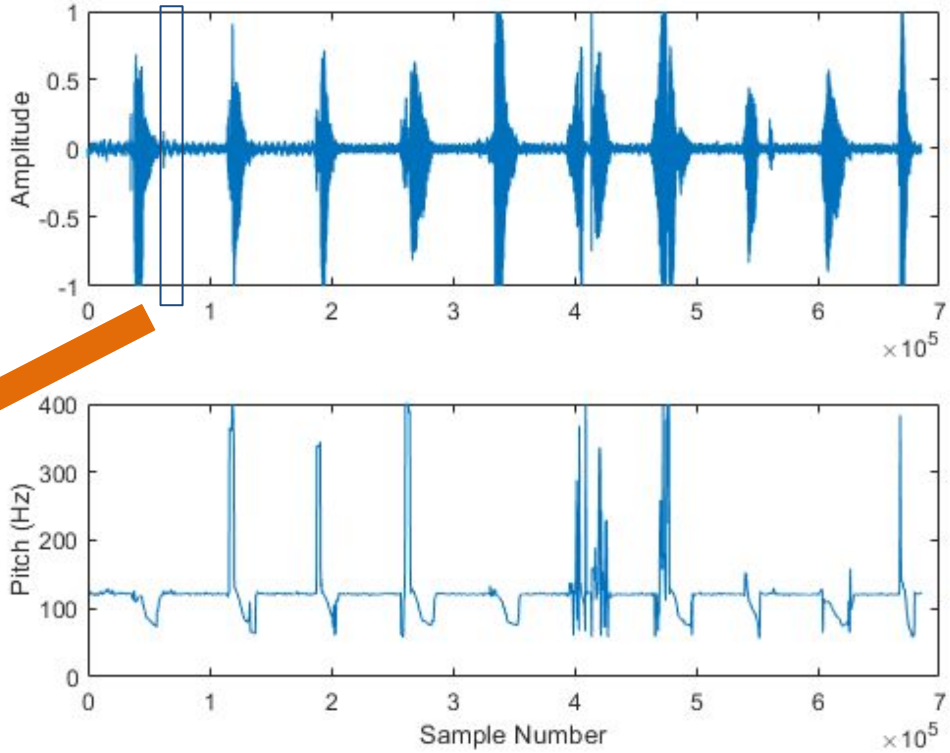


$$\boxed{\Delta = \frac{R}{2^b - 1}}$$

Audio Signal (digital)



	1
1	-0.0812
2	-0.0752
3	-0.1326
4	-0.0181
5	0.0131



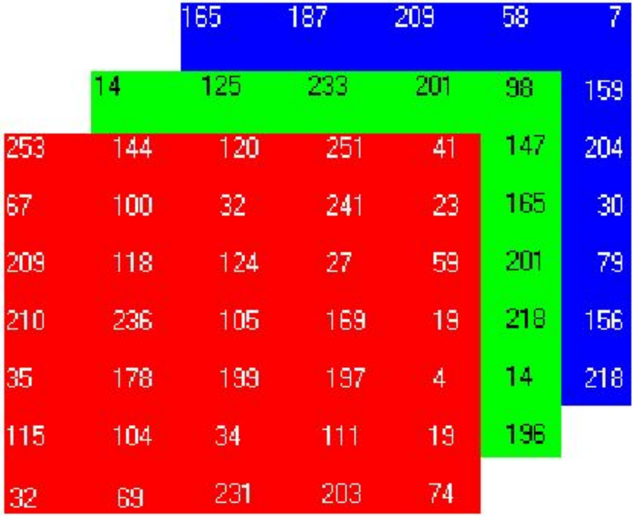
Audio Signal (digital)

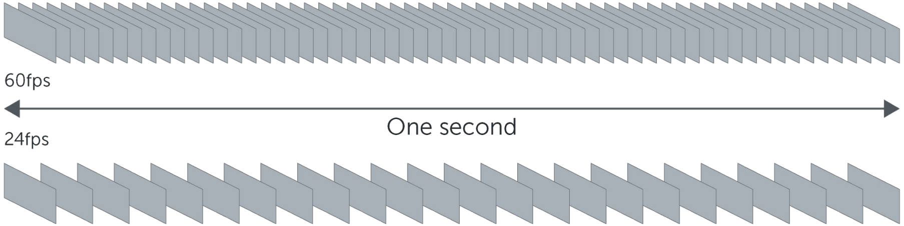
File Sizes For Stereo Digital Audio

Bit Depth	Sample Rate	Bit Rate	1 Stereo Minute	3 Minute Song
16	44,100	1.35 Mbit/sec	10.1 MB	30.3 MB
16	48,000	1.46 Mbit/sec	11.0 MB	33 MB
24	96,000	4.39 Mbit/sec	33.0 MB	99 MB
mp3	128 k/bit rate	0.13 Mbit/sec	0.94 MB	2.82 MB

Discrete Time Signals

Video Signal (digital)





Video Signal (digital)

Type	Video Bitrate, Standard Frame Rate (24, 25, 30)	Video Bitrate, High Frame Rate (48, 50, 60)
2160p (4k)	35-45 Mbps	53-68 Mbps
1440p (2k)	16 Mbps	24 Mbps
1080p	8 Mbps	12 Mbps
720p	5 Mbps	7.5 Mbps

Notes on Quantization

- Quantization introduces errors in the discrete time signal:

$$e_q(n) = x_q(n) - x(n), \quad -\frac{\Delta}{2} < e_q(n) \le \frac{\Delta}{2}$$

- This error (signal) is affected by truncation or rounding and can be treated as noise.
- Quantization errors due to decoding resolution is called **granular noise** while errors due to slope resolution is called **overload noise**
- Most of the time, we observe the quantization error as zero-mean, additive, uncorrelated, stationary, white, and uniformly distributed

Measuring Quantization error

Let $x_q(n)$ be the signal obtained by quantizing $x(n) = \sin 2\pi f_o n$

The quantization error power, P_q is defined by

$$P_q = \frac{1}{N}\sum_{n=0}^{N-1} e^2(n) = \frac{1}{N}\sum_{n=0}^{N-1}\left[x_q(n) - x(n)\right]^2(n)$$

The quality of the quantized signal can be measured by the signal-to-quantization noise ratio (SQNR)

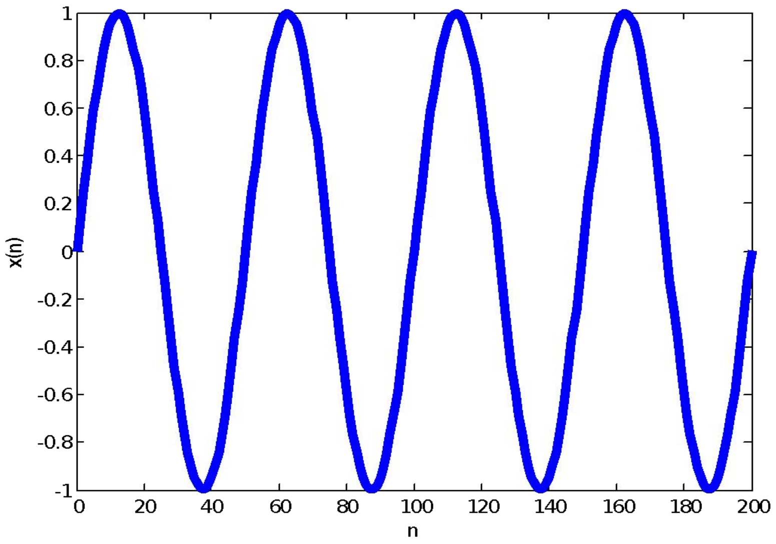
$$SQNR = 10\log_{10}\frac{P_x}{P_q}$$

where P_x is the power of the unquantized signal $x(n)$

Example

For $f_o=1/50$ and $N=200$, write a program to quantize the signal $x(n)$, using **truncation**, *to 64, 128 and 256* quantization levels. Plot $x(n)$, $x_q(n)$ and $e(n)$ and compute *SQNR*

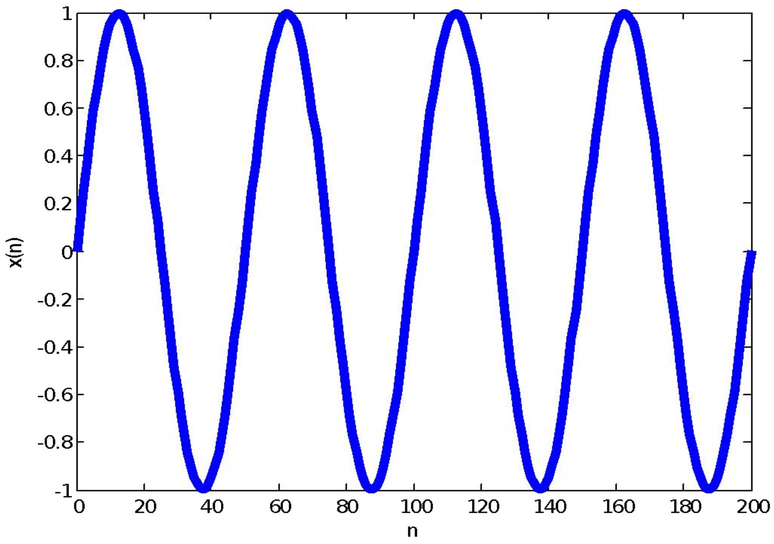
```
>> N=200;
>> fo=1/50;
>> n=[0:N]';
>> x=sin(2*pi*fo*n);
>> plot(n,x)
```



Example

For $f_o=1/50$ and $N=200$, write a program to quantize the signal $x(n)$, using **truncation**, *to 64, 128 and 256* quantization levels. Plot $x(n)$, $x_q(n)$ and $e(n)$ and compute *SQNR*

```
>> Q=256;
>> q=2/(Q+1);
>> xq=q*(floor(x/q));
>> plot(xq)
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
```

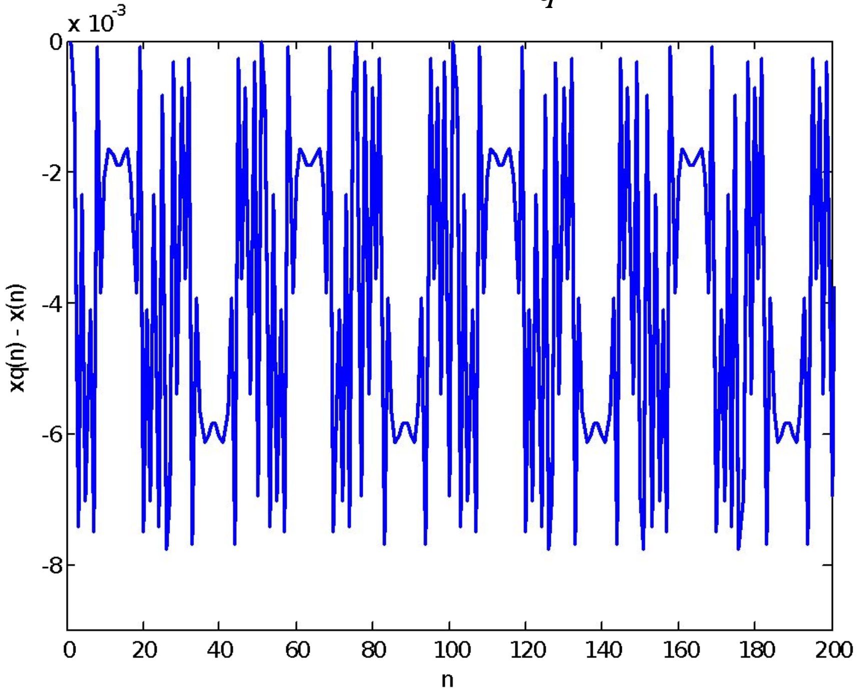


Example

For f_o=1/50 and N=200, write a program to quantize the signal $x(n)$, using **truncation**, *to 64, 128 and 256* quantization levels. Plot $x(n)$, $x_q(n)$ and $e(n)$ and compute *SQNR*

```
>> plot(xq-x)
>> axis([0 200 -9e-3 0])
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
>> Px=sum((x).^2)/N;
>> Pq=sum((xq-x).^2)/N;
>> SQNR=10*log10(Px/Pq)
```

SQNR= 43.56dB

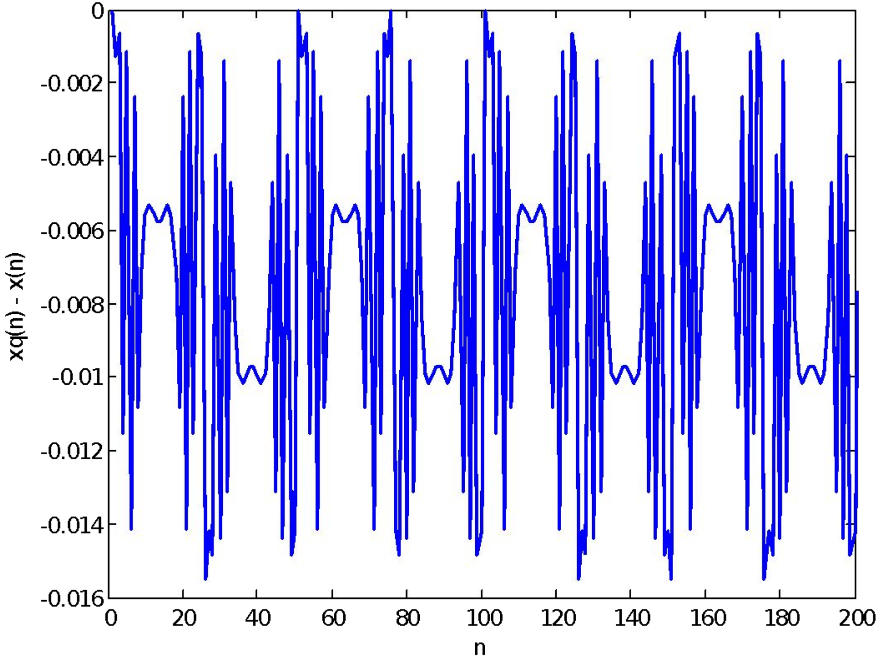


Example

For $f_o=1/50$ and $N=200$, write a program to quantize the signal $x(n)$, using **truncation**, *to 64, 128 and 256* quantization levels. Plot $x(n)$, $x_q(n)$ and $e(n)$ and compute *SQNR*

```
>> Q=128
>> plot(xq-x)
>> axis([0 200 -0.016 0])
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
>> Px=sum((x).^2)/N;
>> Pq=sum((xq-x).^2)/N;
>> SQNR=10*log10(Px/Pq)
```

SQNR= 37.83dB

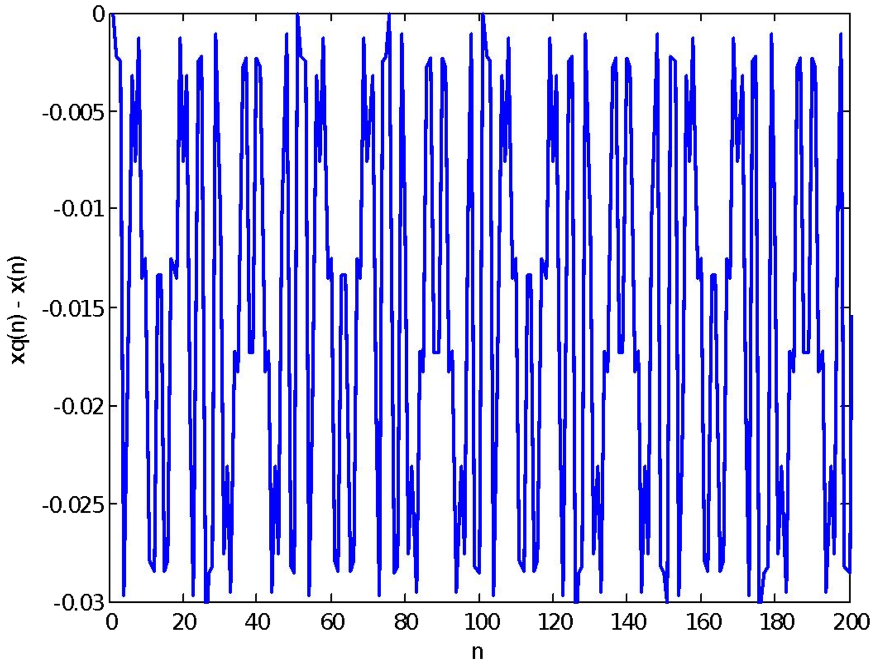


Example

For f_o=*1/50* and *N=200*, write a program to quantize the signal *x(n)*, using **truncation**, *to 64, 128 and 256* quantization levels. Plot *x(n)*, $x_q(n)$ and *e(n)* and compute *SQNR*

```
>> Q=64
>> plot(xq-x)
>> axis([0 200 -0.03 0])
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
>> Px=sum((x).^2)/N;
>> Pq=sum((xq-x).^2)/N;
>> SQNR=10*log10(Px/Pq)
```

SQNR= 31.53dB

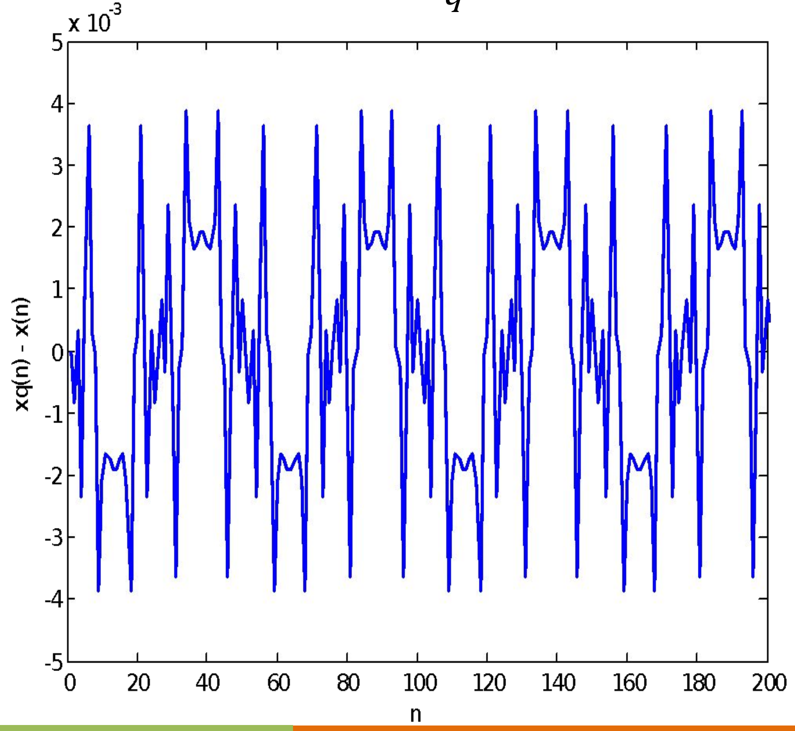


Example

For $f_o=1/50$ and $N=200$, write a program to quantize the signal $x(n)$, using **rounding**, *to 64, 128 and 256* quantization levels. Plot $x(n)$, $x_q(n)$ and $e(n)$ and compute *SQNR*

```
>> Q=256
>> xq=q*(round(x/q));
>> plot(xq-x)
>> axis([0 200 -0.005 0.005])
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
>> Px=sum((x).^2)/N;
>> Pq=sum((xq-x).^2)/N;
>> SQNR=10*log10(Px/Pq)
```

SQNR= 51.05dB

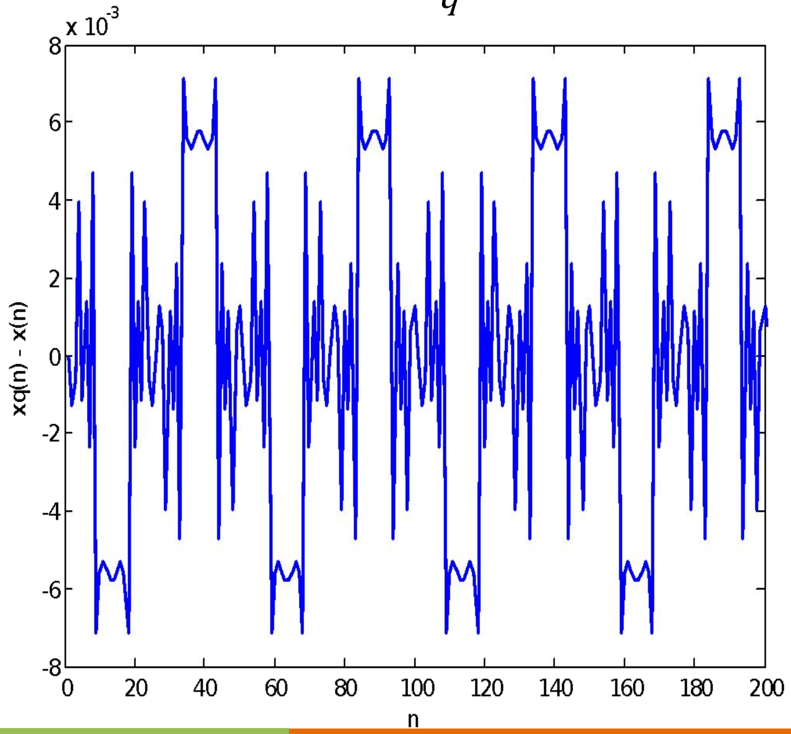


Example

For f_o*=1/50* and *N=200*, write a program to quantize the signal *x(n)*, using **rounding**, *to 64, 128 and 256* quantization levels. Plot *x(n)*, $x_q(n)$ and *e(n)* and compute *SQNR*

```
>> Q=128
>> xq=q*(round(x/q));
>> plot(xq-x)
>> axis([0 200 -0.008 0.008])
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
>> Px=sum((x).^2)/N;
>> Pq=sum((xq-x).^2)/N;
>> SQNR=10*log10(Px/Pq)
```

SQNR= 44.48dB

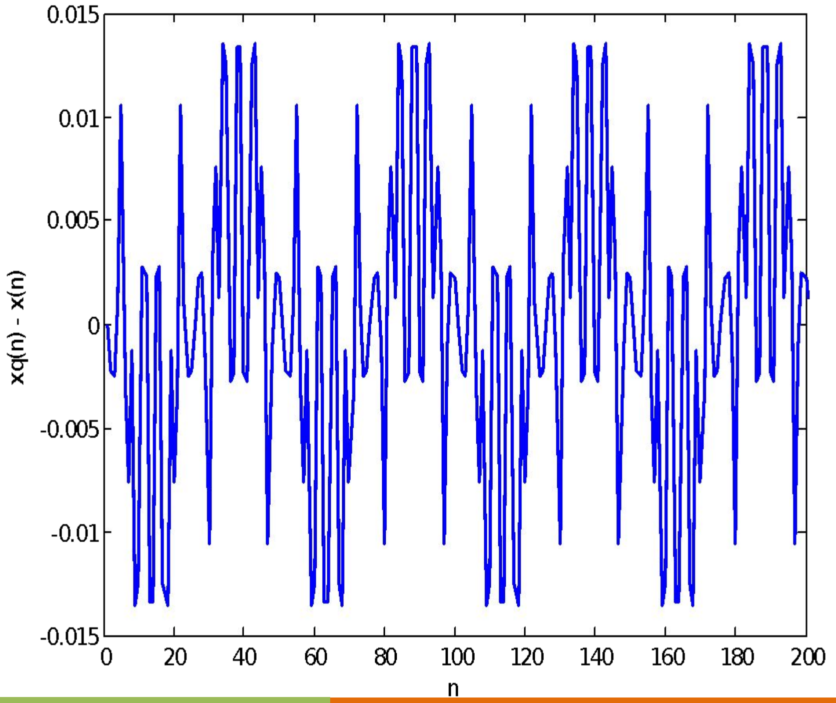


Example

For f_o=1/50 and N=200, write a program to quantize the signal $x(n)$, using **rounding**, *to 64, 128 and 256* quantization levels. Plot $x(n)$, $x_q(n)$ and $e(n)$ and compute *SQNR*

```
>> Q=64
>> xq=q*(round(x/q));
>> plot(xq-x)
>> axis([0 200 -0.015 0.015])
>> xlabel('n');
>> ylabel('xq(n) - x(n)')
>> Px=sum((x).^2)/N;
>> Pq=sum((xq-x).^2)/N;
>> SQNR=10*log10(Px/Pq)
```

SQNR= 39.34dB



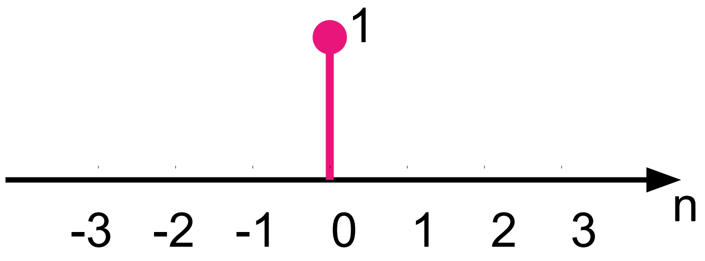
Discrete Time Signal Representation

- Most of the discrete time signals we encounter can be described by their wave characteristics (e.g. amplitude, frequency, phase, decay time)

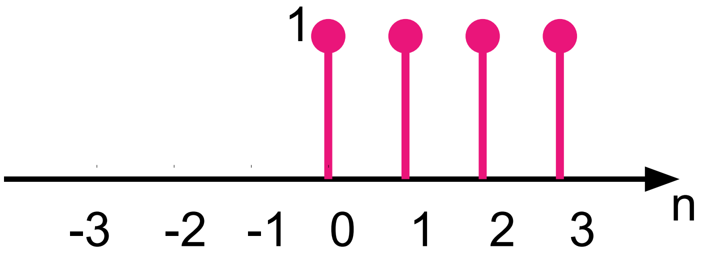
- In general, they can be mathematically represented using **elementary functions, exponential functions,** and **sinusoidal functions**

Elementary Functions

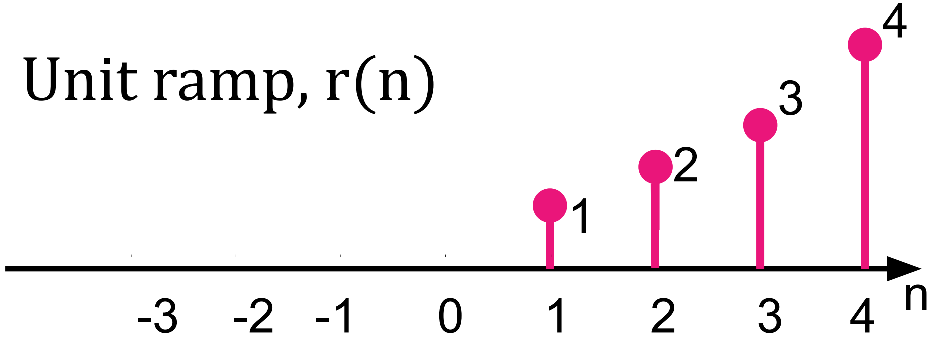
Unit impulse, δ(n)



Unit step, u(n)



Unit ramp, r(n)



Exponential Functions

Exponential, $a^n u(n)$

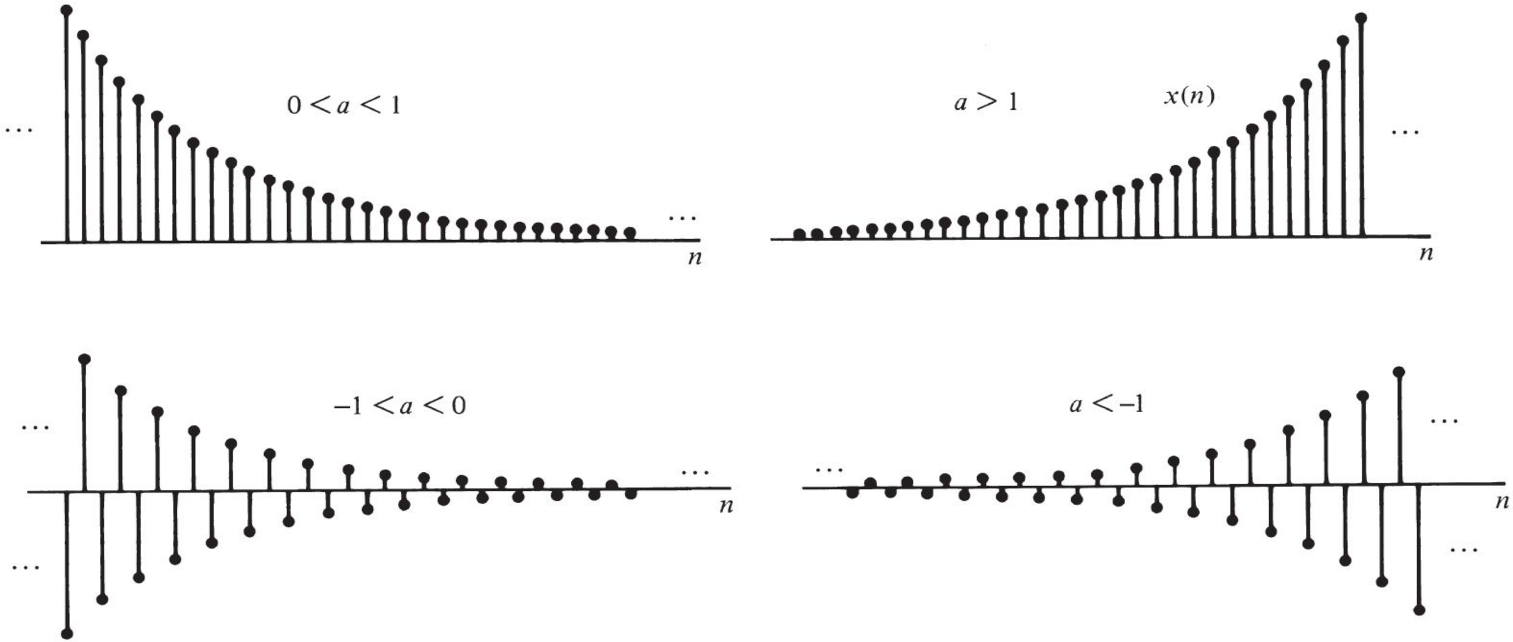


Figure 2.1.5 Graphical representation of exponential signals.

Sinusoidal Functions

$$x[n] = \text{Asin}(\omega n + \emptyset)$$

$$x[n] = \text{Asin}\left(\frac{2\pi kn}{N} + \emptyset\right)$$

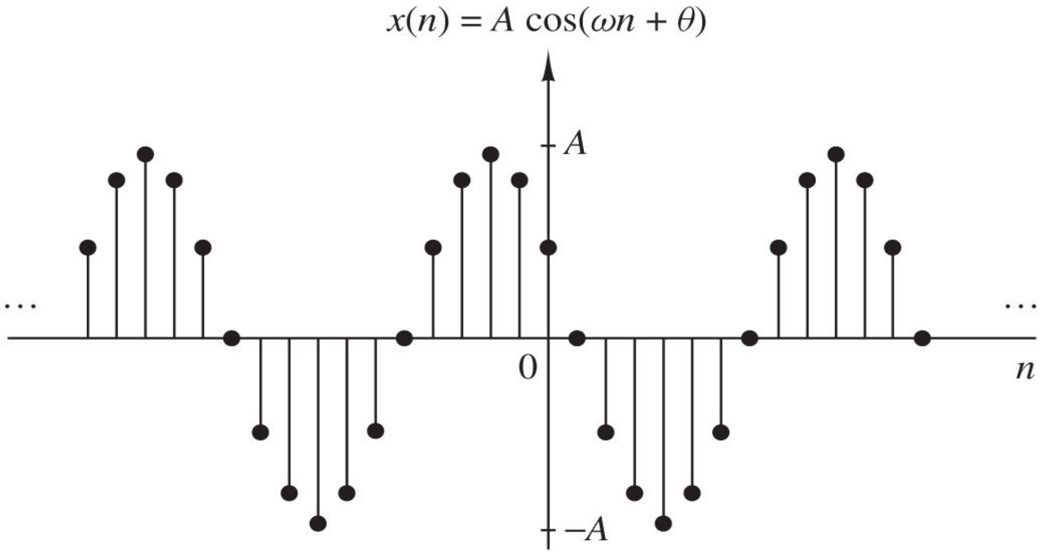
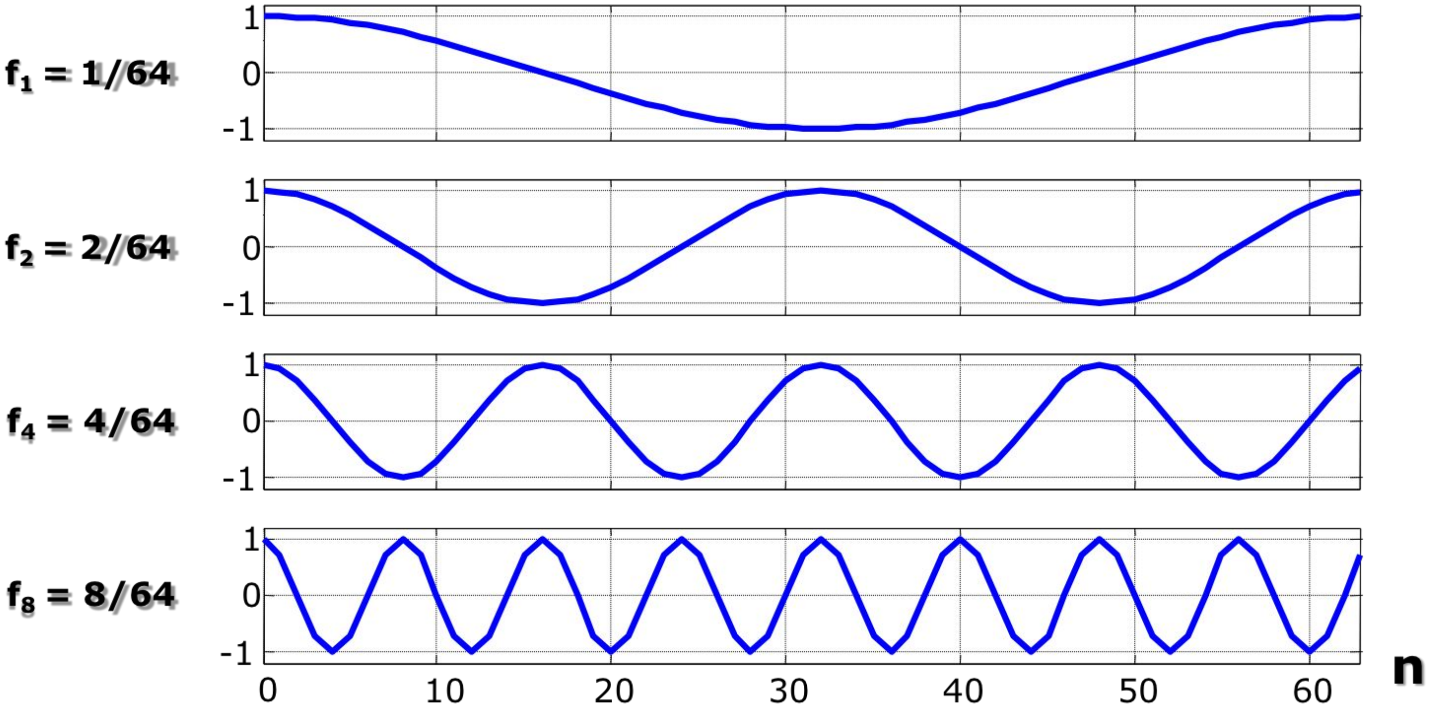


Figure 1.3.3 Example of a discrete-time sinusoidal signal ($\omega = \pi/6$ and $\theta = \pi/3$).

Sinusoidal Functions



Signal Frequency vs. Sampling Frequency

Nyquist's Sampling Theorem:

> " If the highest frequency contained in an analog signal, $x_a(t)$, is F_{max} and the signal is sampled at a rate $F_s > 2F_{max}$, then $x_a(t)$ can be exactly recovered from its sample values using the interpolation function"

- Not following Nyquist criterion may lead to **aliasing**

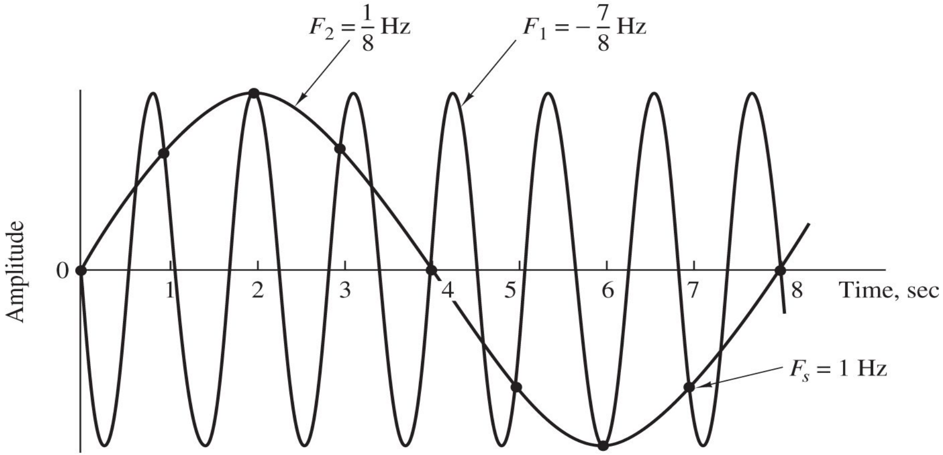
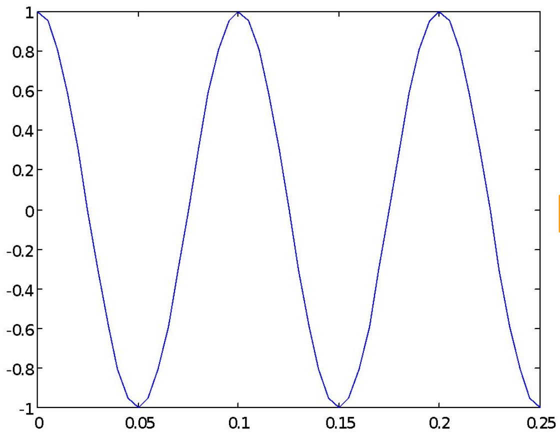


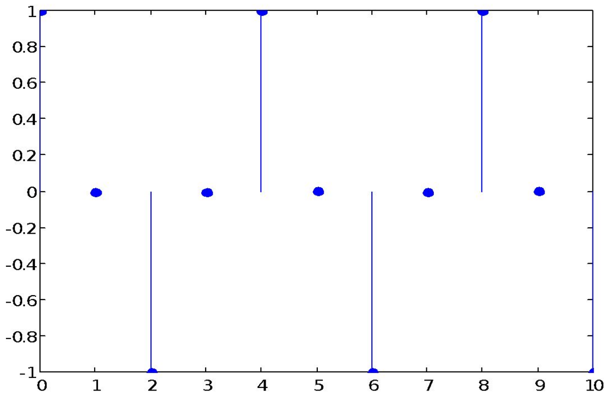
Figure 1.4.5 Illustration of aliasing.

Signal Frequency vs. Sampling Frequency

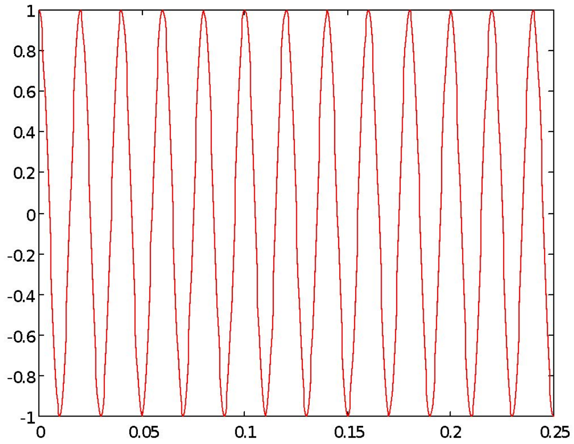
Fs = 40 Hz



$x_1(t) = cos(2\pi\ 10t)$



$x(n) = cos(0.5\pi n)$
$= cos(2.5\pi n)$



$x_2(t) = cos(2\pi\ 50t)$

Discrete Time Signal Classification

Discrete time signals can be classified according to:

1. Power or Energy
2. Periodicity
3. Symmetry

Discrete Time Signal Classification

We define the **energy** of a discrete time signal as:

$$E \equiv \sum_{n=-\infty}^{\infty} \left| x(n) \right|^2$$

- ➢ If *E* is finite, then *x(n)* is an *energy signal*

Discrete Time Signal Classification

We define the **power** of a discrete time signal as:

$$P = \lim_{N \to \infty} \frac{1}{2N+1} \sum_{n=-N}^{N} \left| x(n) \right|^2$$

- If P is finite, then $x(n)$ is a *power signal*

Discrete Time Signal Classification

- The unit step has infinite energy, it is a power signal with $P=0.5$

- Complex exponential $Ae^{j\omega n}$ is a power signal with $P=A^2$

- The unit ramp sequence is neither a power signal nor an energy signal

Discrete Time Signal Classification

A signal *x(n)* is periodic with period *N* if and only if

$$x(n+N) = x(n) \qquad \textit{for all } n$$

Ex. $x(n) = A\sin(2\pi f_0 n)$ periodic when f_o is a rational number

➢ **Periodic** signals are **power** signals.

Discrete Time Signal Classification

A DT signal *x(n)* is **even** if:

$$\boxed{x[n] = x[-n]}$$

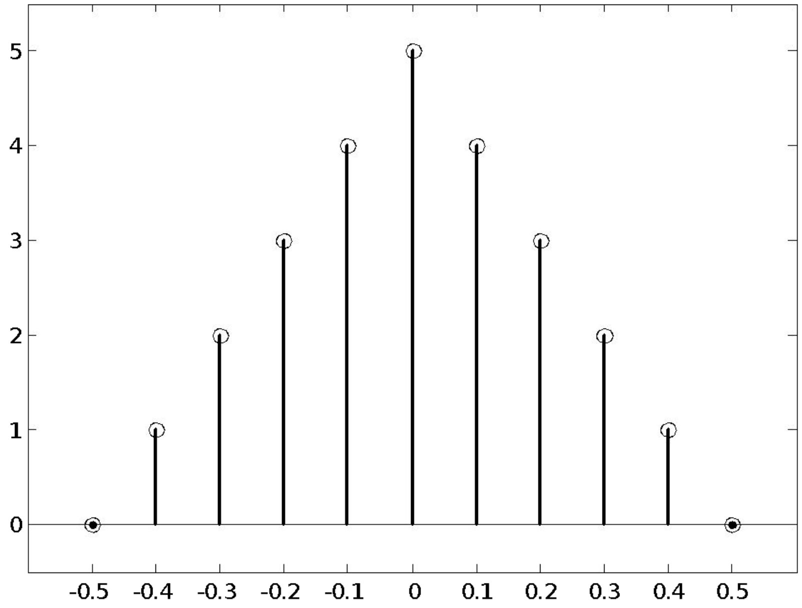
A DT signal *x(n)* is **odd** if:

$$\boxed{x[n] = -x[-n]}$$

Discrete Time Signal Classification

The following DTS signal is:

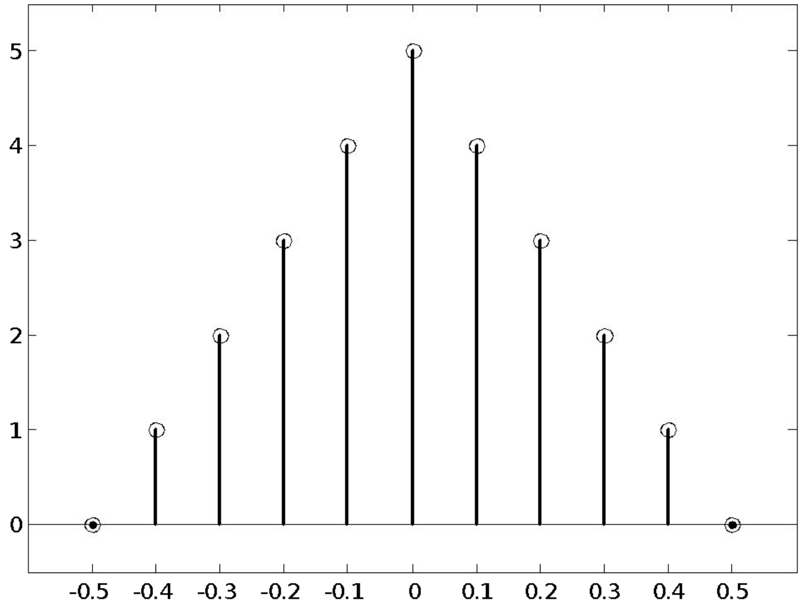
a. Power
b. Energy
c. Periodic
d. Aperiodic
e. Even
f. Odd



Discrete Time Signal Classification

The following DTS signal is:

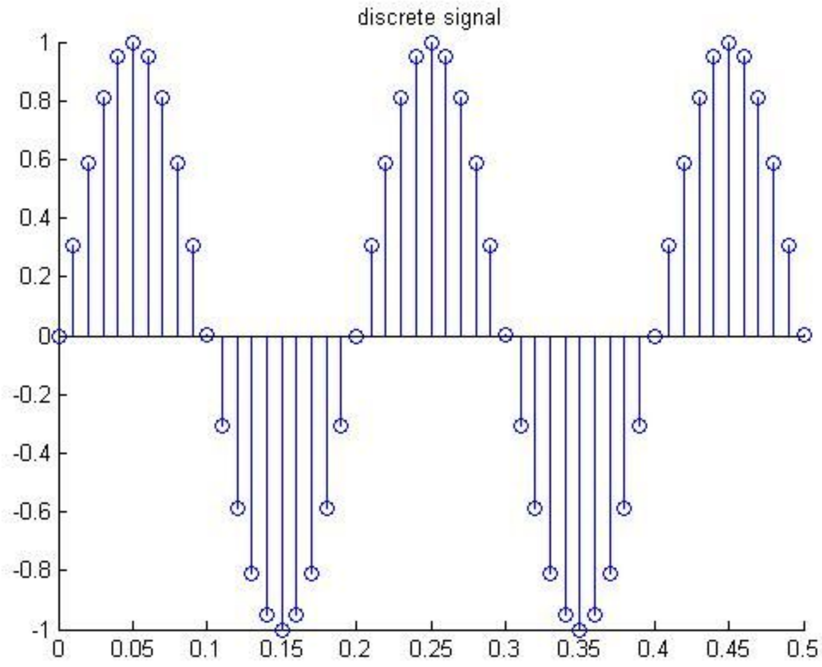
a. Power
b. Energy
c. Periodic
d. Aperiodic
e. Even
f. Odd



Discrete Time Signal Classification

The following DTS signal is:

a. Power
b. Energy
c. Periodic
d. Aperiodic
e. Even
f. Odd

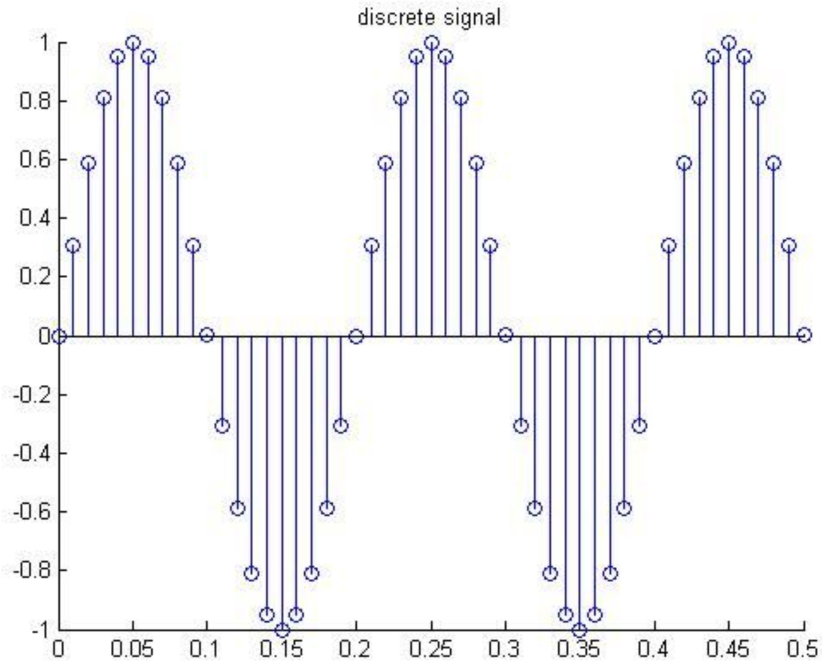


(repeating sequence)

Discrete Time Signal Classification

The following DTS signal is:

a. **Power**
b. Energy
c. **Periodic**
d. Aperiodic
e. Even
f. **Odd**



(repeating sequence)

Summary

- Discrete time signals can be obtained from sampling and quantizing continuous time signals.
- To ensure proper signal representation, Nyquist criteria and proper quantization must be followed
- Elementary functions, exponential functions, and sinusoidal functions can be used to mathematically represent any discrete time signal
- Discrete time signals can be classified as power or energy, periodic or aperiodic, and odd or even

For further reading...

- Chapters 1.4-2.1.2

  "Digital Signal Processing, 3rd ed. By Proakis, J. & Manolakis, D."

- Chapter 1-2.3

  "Digital Signal Processing: A Computer-Based Approach by Mitra, S."

1. Discrete Time Signal Representation

EE 274/COE 197E

Rhandley Cajote, Ph. D.
Crisron Lucas, MSc.