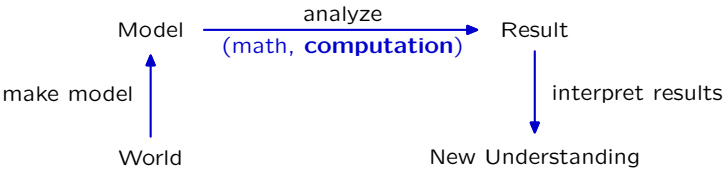
6.003: Signal Processing

Sampling and Aliasing

February 13, 2020

Importance of Discrete Representations

Our goal is to develop **signal processing** tools that help us understand and manipulate the world.



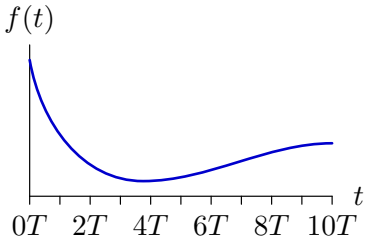
The **increasing power** and **decreasing cost** of computation makes the use of computation increasingly attractive.

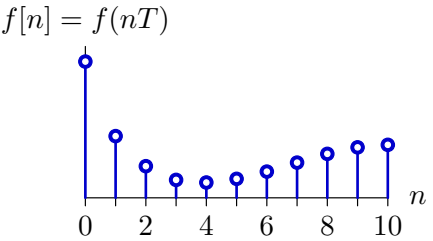
However, many important signals are naturally described with continuous functions, that must be **sampled** in order to be analyzed computationally.

Today: understand relations between **continuous** and **sampled** signals.

Sampling

How does sampling affect the information contained in a signal?





$T =$ sampling interval

Notation:

We will use parentheses to denote functions of continuous domain ($f(t)$) and square brackets to denote functions of discrete domain ($f[n]$).

Effects of Sampling Are Easily Heard

Sampling Music

$$f_s = \frac{1}{T}$$

- $f_s = 44.1$ kHz
- $f_s = 22$ kHz
- $f_s = 11$ kHz
- $f_s = 5.5$ kHz
- $f_s = 2.8$ kHz

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Effects of Sampling are Easily Seen

Sampling Images

original: 2048×1536

Effects of Sampling are Easily Seen

Sampling Images

downsampled: 1024×768

Effects of Sampling are Easily Seen

Sampling Images

downsampled: $512x384$

Effects of Sampling are Easily Seen

Sampling Images

downsampled: 256×192

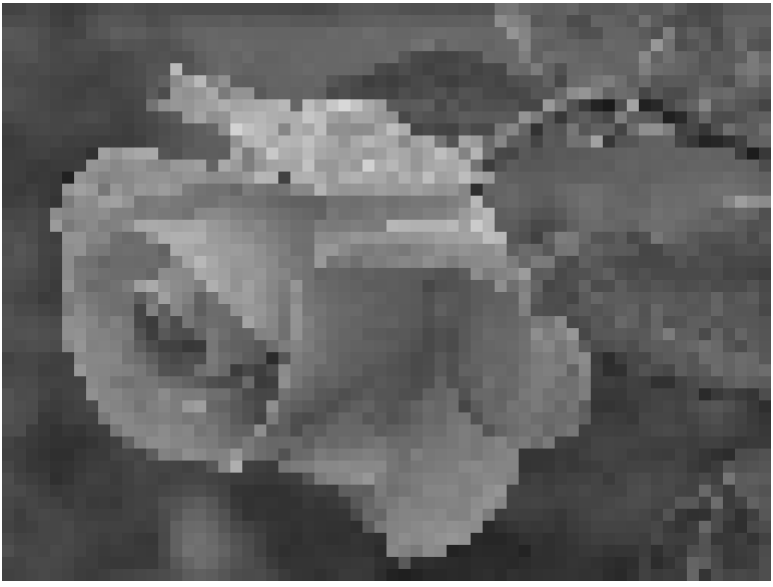
Effects of Sampling are Easily Seen

Sampling Images

downsampled: 128×96

Effects of Sampling are Easily Seen

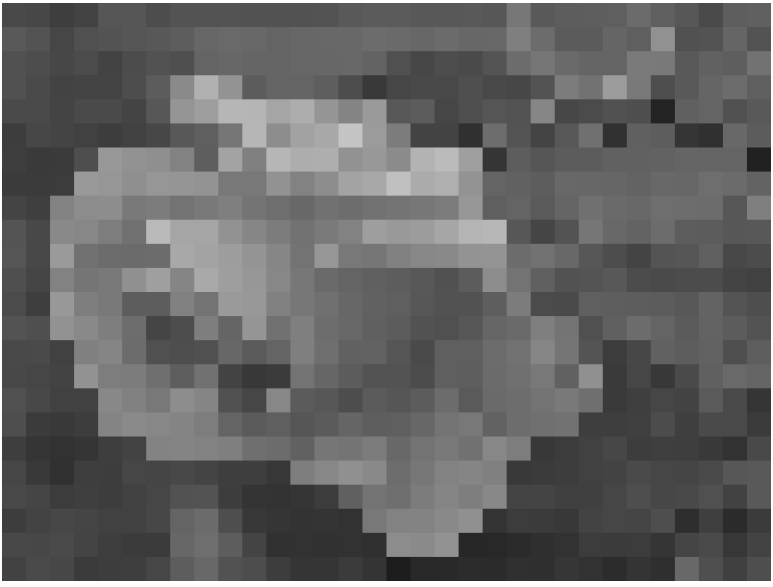
Sampling Images



downsampled: 64×48

Effects of Sampling are Easily Seen

Sampling Images



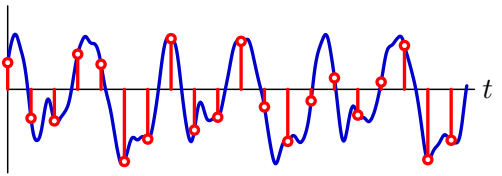
downsampled: 32×24

Characterizing Sampling

We would like to sample in a way that preserves **information**.

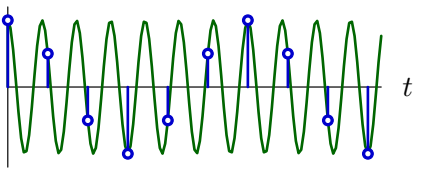
However, information is generally **lost** in the sampling process.

Example: samples provide no information about the intervening values.



Furthermore, information that is retained by sampling can be misleading.

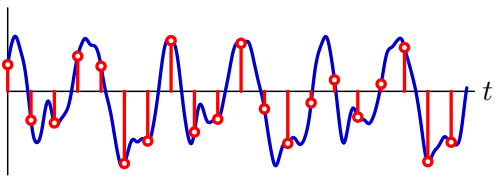
Example: samples can suggest patterns not contained in the original.



Characterizing Sampling

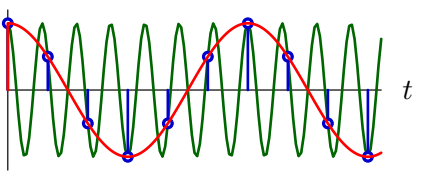
We would like to sample in a way that preserves **information**.
However, information is generally **lost** in the sampling process.

Example: samples provide no information about the intervening values.



Furthermore, information that is retained by sampling can be misleading.

Example: samples can suggest patterns not contained in the original.



Samples (blue) of the original high-frequency signal (green) could just as easily have come from a much lower frequency signal (red).

Characterizing Sampling

Our goal is to understand sampling so that we can mitigate its effects on the information contained in the signals we process.

Check Yourself

Consider 3 CT signals:

$$f_1(t) = \cos(4000t) \quad ; \quad f_2(t) = \cos(5000t) \quad ; \quad f_3(t) = \cos(6000t)$$

Each of these is sampled so that

$$f_1[n] = f_1(nT) \quad ; \quad f_2[n] = f_2(nT) \quad ; \quad f_3[n] = f_3(nT)$$

where $T = 0.001$.

Which list goes from lowest to highest DT frequency?

0. $f_1[n]$ $f_2[n]$ $f_3[n]$
1. $f_1[n]$ $f_3[n]$ $f_2[n]$
2. $f_2[n]$ $f_1[n]$ $f_3[n]$
3. $f_2[n]$ $f_3[n]$ $f_1[n]$
4. $f_3[n]$ $f_1[n]$ $f_2[n]$
5. $f_3[n]$ $f_2[n]$ $f_1[n]$

Check Yourself

The CT signals are simple sinusoids:

$$f_1(t) = \cos(4000t) \quad ; \quad f_2(t) = \cos(5000t) \quad ; \quad f_3(t) = \cos(6000t)$$

The DT signals are sampled versions ($T = 0.001$):

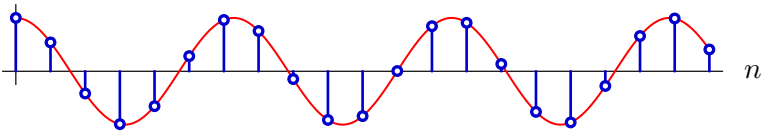
$$f_1[n] = \cos(4n) \quad ; \quad f_2[n] = \cos(5n) \quad ; \quad f_3[n] = \cos(6n)$$

How does the shape of $\cos(\Omega n)$ depend on Ω?

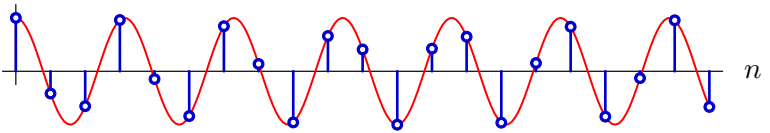
Check Yourself

As the frequency Ω increases, the shapes of the sampled signals deviate from those of the underlying CT signals.

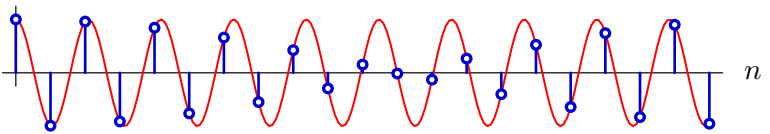
$\Omega = 1 : x[n] = \cos(n)$



$\Omega = 2 : x[n] = \cos(2n)$



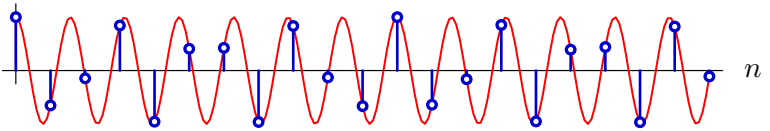
$\Omega = 3 : x[n] = \cos(3n)$



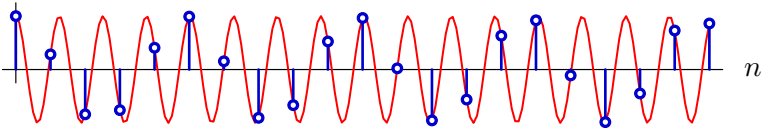
Check Yourself

Worse and worse representation.

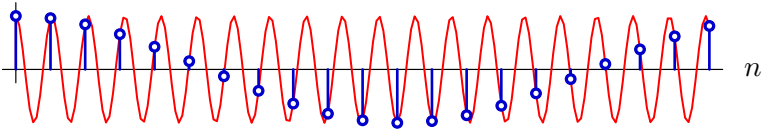
$$\Omega = 4 : x[n] = \cos(4n) = \cos\Big((2\pi - 4)n\Big) \approx \cos(2.283n)$$



$$\Omega = 5 : x[n] = \cos(5n) = \cos\Big((2\pi - 5)n\Big) \approx \cos(1.283n)$$



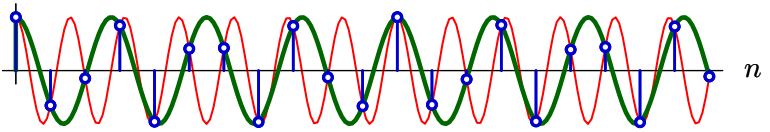
$$\Omega = 6 : x[n] = \cos(6n) = \cos\Big((2\pi - 6)n\Big) \approx \cos(0.283n)$$



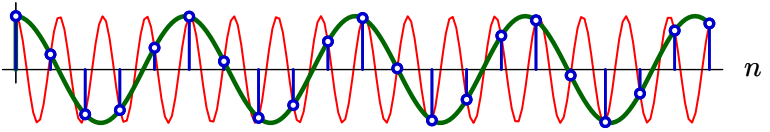
Check Yourself

For $\Omega > \pi$, a lower frequency Ω_L has the same sample values as Ω.

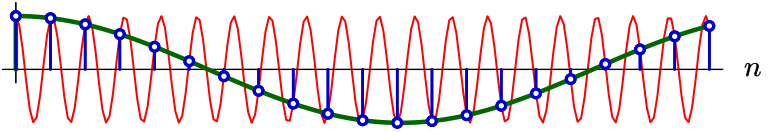
$$\Omega = 4 : x[n] = \cos(4n) = \cos\Big((2\pi - 4)n\Big) \approx \cos(2.283n)$$



$$\Omega = 5 : x[n] = \cos(5n) = \cos\Big((2\pi - 5)n\Big) \approx \cos(1.283n)$$



$$\Omega = 6 : x[n] = \cos(6n) = \cos\Big((2\pi - 6)n\Big) \approx \cos(0.283n)$$



The same DT sequence represents multiple different values of Ω.

Check Yourself

Consider 3 CT signals:

$$f_1(t) = \cos(4000t) \quad ; \quad f_2(t) = \cos(5000t) \quad ; \quad f_3(t) = \cos(6000t)$$

Each of these is sampled so that

$$f_1[n] = f_1(nT) \quad ; \quad f_2[n] = f_2(nT) \quad ; \quad f_3[n] = f_3(nT)$$

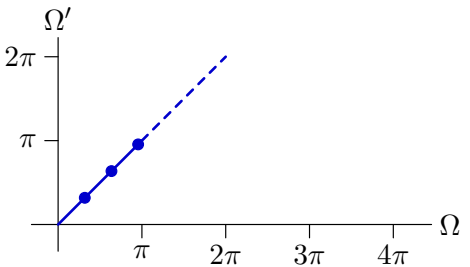
where $T = 0.001$.

Which list goes from lowest to highest DT frequency? 5

0. $f_1[n]$ $f_2[n]$ $f_3[n]$
1. $f_1[n]$ $f_3[n]$ $f_2[n]$
2. $f_2[n]$ $f_1[n]$ $f_3[n]$
3. $f_2[n]$ $f_3[n]$ $f_1[n]$
4. $f_3[n]$ $f_1[n]$ $f_2[n]$
5. $f_3[n]$ $f_2[n]$ $f_1[n]$

Aliasing

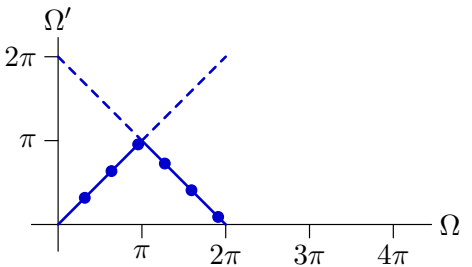
Plot the relation between Ω (the original discrete frequency) and Ω' which is the apparent discrete frequency of the sampled signal.



The first three points of the previous example fall on the line $\Omega' = \Omega$.

Aliasing

Plot the relation between Ω (the original discrete frequency) and Ω' which is the apparent discrete frequency of the sampled signal.



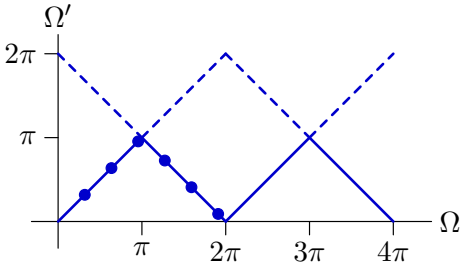
The first three points fall on the line $\Omega' = \Omega$.

The next three points fall on the line $\Omega' = 2\pi - \Omega$

$$\cos\Big((2\pi - \Omega)n\Big) = \cos(2\pi n)\cos(\Omega n) - \sin(2\pi n)\sin(\Omega n) = \cos(\Omega n)$$

Aliasing

Plot the relation between Ω (the original discrete frequency) and Ω' which is the apparent discrete frequency of the sampled signal.



The first three points fall on the line $\Omega' = \Omega$.

The next three points fall on the line $\Omega' = 2\pi - \Omega$

$$\cos\Big((2\pi - \Omega)n\Big) = \cos(2\pi n)\cos(\Omega n) - \sin(2\pi n)\sin(\Omega n) = \cos(\Omega n)$$

The cosine function is periodic with period 2π.

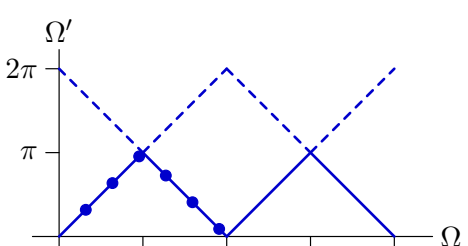
Therefore this pattern repeats:

$\Omega' = 2\pi + \Omega$

$\Omega' = 4\pi - \Omega$

Aliasing

Plot the relation between Ω (the original discrete frequency) and Ω' which is the apparent discrete frequency of the sampled signal.



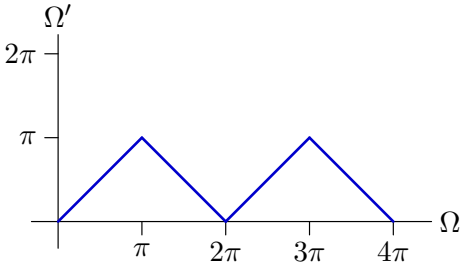
Notice that there are an infinite number of Ω' that produce the same samples as Ω.

Also, every value of apparent frequency Ω' could have come from an infinite number of input frequencies Ω.

Most importantly, sampling a CT sinusoid results in a DT sinusoid.
The frequency may not be what we expected, but the behavior is sinusoidal!

Aliasing

We can assure a unique output at every input frequency Ω by requiring that $0 \leq \Omega' \leq \pi$. We refer to this region as the **baseband**.



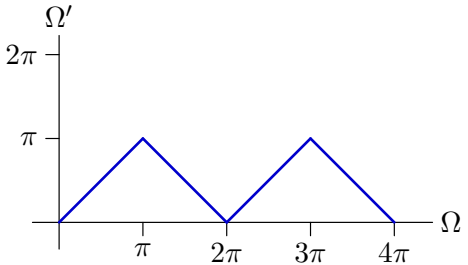
I used **baseband reconstruction** to produce the sampled versions of music in the beginning of today's lecture.

Baseband reconstruction can be implemented by removing output frequencies greater than π. The operation of removing components based on their frequency content is called **filtering**.

We can use Fourier analysis to express the output signal as a sum of sinusoids and remove sinusoids with frequencies above π.

Aliasing

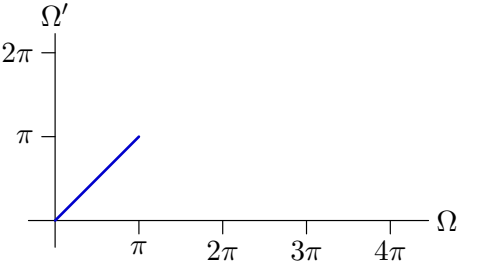
The remaining distortions are (primarily) caused by input frequencies $\Omega > \pi$. These frequencies generate output frequencies Ω' that are not desired.



Since multiple input frequencies generate the same output frequency, we refer to this process as **aliasing**.

Aliasing

We can prevent aliasing by removing **input** frequencies greater than π.
This is just another application of **filtering**.



Anti-Aliasing Demonstration

Sampling Music

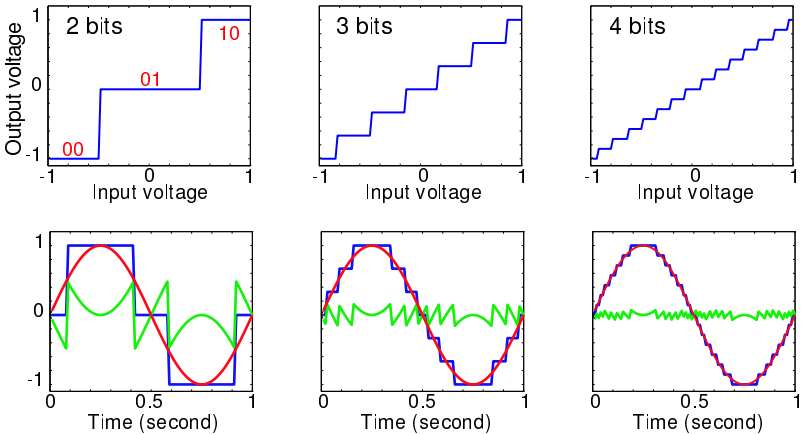
$$\omega_s = \frac{2\pi}{T} = 2\pi f_s$$

- $f_s = 11$ kHz without anti-aliasing
- $f_s = 11$ kHz with anti-aliasing
- $f_s = 5.5$ kHz without anti-aliasing
- $f_s = 5.5$ kHz with anti-aliasing
- $f_s = 2.8$ kHz without anti-aliasing
- $f_s = 2.8$ kHz with anti-aliasing

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization

The information content of a signal depends not only with sample rate but also with the number of bits used to represent each sample.



Bit rate = (# bits/sample)×(# samples/sec)

Check Yourself

We hear sounds that range in amplitude from 1,000,000 to 1.

How many bits are needed to represent this range?

1. 5 bits
2. 10 bits
3. 20 bits
4. 30 bits
5. 40 bits

Check Yourself

How many bits are needed to represent 1,000,000:1?

bits	range
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1,024
11	2,048
12	4,096
13	8,192
14	16,384
15	32,768
16	65,536
17	131,072
18	262,144
19	524,288
20	1,048,576

Check Yourself

We hear sounds that range in amplitude from 1,000,000 to 1.

How many bits are needed to represent this range? 3

1. 5 bits
2. 10 bits
3. 20 bits
4. 30 bits
5. 40 bits

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization Demonstration

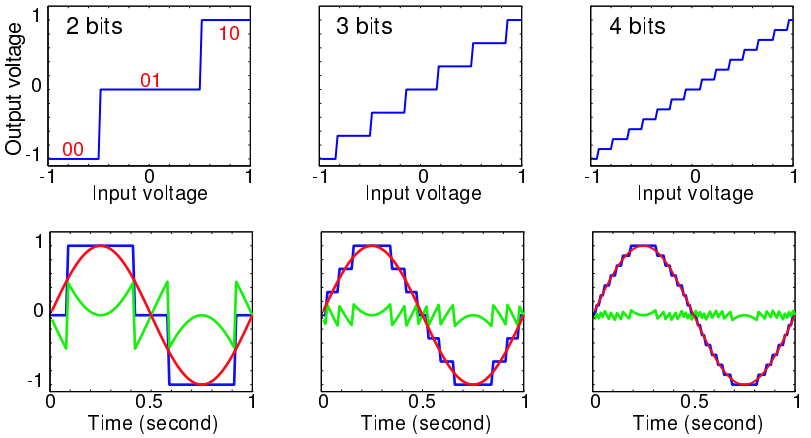
Quantizing Music

- 16 bits/sample
- 8 bits/sample
- 6 bits/sample
- 4 bits/sample
- 3 bits/sample
- 2 bit/sample

J.S. Bach, Sonata No. 1 in G minor Mvmt. IV. Presto
Nathan Milstein, violin

Quantization

We measure discrete amplitudes in bits.



Example: high-quality audio

$$2\,\text{channels} \times 16\,\frac{\text{bits}}{\text{sample}} \times 44,100\,\frac{\text{samples}}{\text{sec}} \times 60\,\frac{\text{sec}}{\text{min}} \times 74\,\text{min} \approx 6.3\,\text{G bits}$$

$$\approx 0.78\,\text{G bytes}$$

Quantizing Images

Converting an image from a continuous representation to a discrete representation involves the same sort of issues.

This image has 280×280 pixels, with brightness quantized to 8 bits.

Quantizing Images

8 bit image

7 bit image

Quantizing Images

8 bit image

6 bit image

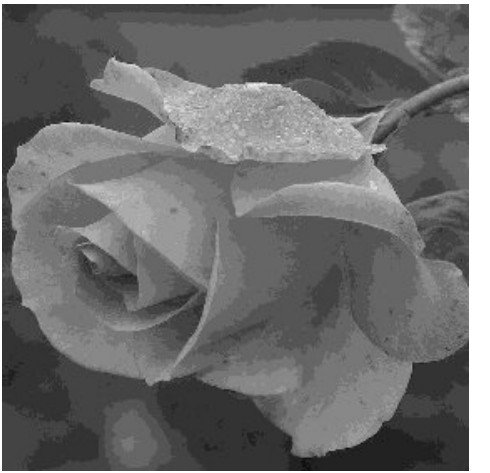
Quantizing Images

8 bit image

5 bit image

Quantizing Images

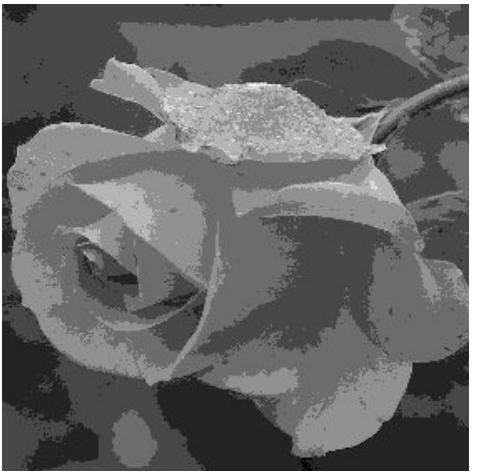
8 bit image



4 bit image

Quantizing Images

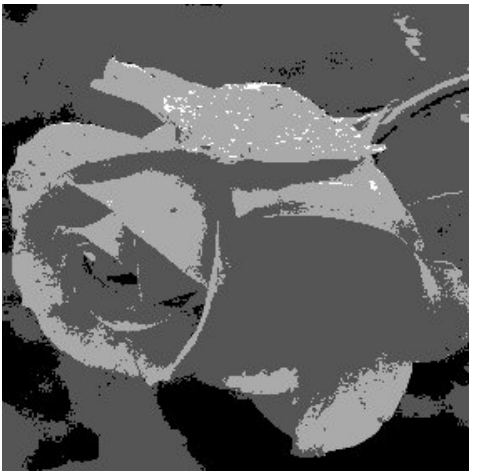
8 bit image



3 bit image

Quantizing Images

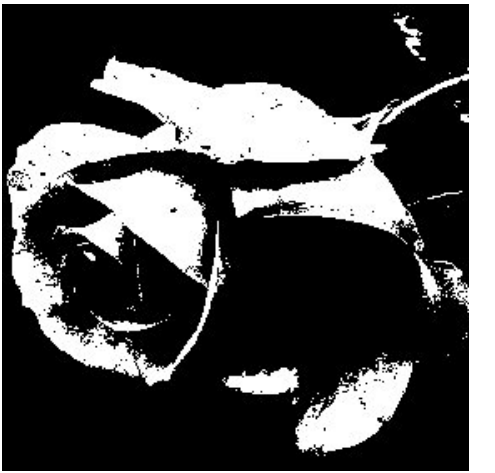
8 bit image



2 bit image

Quantizing Images

8 bit image



1 bit image

Summary

Sampling and Discrete-Time (DT) signals.

DT signals are useful for modeling discrete sequences:
e.g., Fourier series coefficients

DT signals also allow us to analyze CT behaviors using computation,
but this kind of analysis requires **sampling**.

Ill-posed sampling can result in unwanted distortions, but these distortions can be minimized.

Sinusoidal analysis provide a powerful tool for understanding and reducing sampling artifacts.

Quantization of a signal's amplitude can also introduce unwanted distortions. We will study those distortions later in the semester.

Next Week: Discrete-Time Fourier Series