

Digital Signal Processing

I. Analog and Digital Signals

Signals

- ▶ Signal = a measurable quantity which varies in time, space or some other variable
- ▶ Examples:
 - ▶ a voltage which varies in time (1D voltage signal)
 - ▶ sound pressure which varies in time (sound signal)
 - ▶ intensity of light which varies across a photo (2D image)
- ▶ Represented as a mathematical function, e.g. $v(t)$.

- ▶ Glossary:

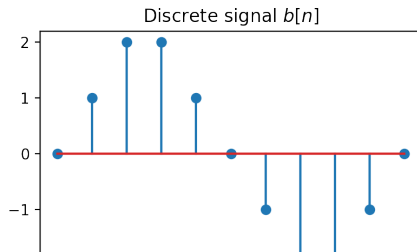
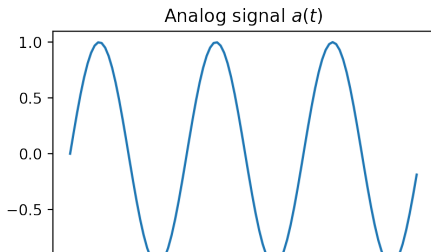
- ▶ “e.g.” = “*exempli gratia*” (lat.) = “for example” (eng.) = “de exemplu” (rom.)
- ▶ “i.e.” = “*id est*” (lat) = “that is” (eng.) = “adică” (rom.)

- ▶ **Unidimensional** (1D) signal = a function of a single variable
 - ▶ Example: a voltage signal $v(t)$ only varies in time.
- ▶ **Multidimensional** (2D, 3D ... M-D) signal = a function of a multiple variables
 - ▶ Example: intensity of a grayscale image $I(x, y)$ across the surface of a photo
- ▶ In these lectures we consider only 1D signals, but the theory is similar

Continuous and discrete signals

- ▶ Continuous (analog) signal = function of a continuous variable
 - ▶ Signal has a value for possible value of the variable in the defined range
 - ▶ The variable may be defined only in a certain range (e.g. $t \in [0, 100]$), but it is a compact range
- ▶ Discrete signal = function of a discrete variable
 - ▶ Signal has values only at certain discrete values (*samples*)
 - ▶ Indexed with natural numbers: $x[-1]$, $x[0]$, $x[1]$ etc.
 - ▶ Outside the samples, the signal is **not defined**

<matplotlib.text.Text at 0x7f770d66da90>



Notation

- ▶ We use the following notation throughout these lectures
- ▶ Continuous signal
 - ▶ Has **round parentheses**, e.g. $x_a(t)$
 - ▶ Sometimes has the a subscript
 - ▶ The variable is usually t (time)
 - ▶ $x(2.3)$ = the value of the signal $a(t)$ at $t = 2.3$
- ▶ Discrete signal
 - ▶ Has **square brackets**, e.g. $x[n]$
 - ▶ The variables are denoted as n or k (suggest natural numbers)
 - ▶ $x[3]$ = the value of the signal $x[n]$ for $n = 3$
 - ▶ $x[1.5]$ = does not exist

Signals with continuous and discrete values

- ▶ The signal values can be continuous or discrete
 - ▶ Example: signal values stored as 8-bit or 16-bit values
- ▶ On digital systems, signals always have discrete values due to finite number precision

Discrete frequency

- ▶ A signal is **periodic** if the values repeat themselves after a certain time (**period**)
- ▶ Frequency = inverse of period
- ▶ Pulsation $\omega = 2 * \pi * \text{frequency}$
- ▶ Continuous signals:
 - ▶ Periodic: $x_a(t) = x(t + T)$
 - ▶ T is usually measured in seconds (or some other unit)
 - ▶ $F = \frac{1}{T}$ is measured in Hz = $\frac{1}{s}$ (Hertz)
- ▶ Discrete signals:
 - ▶ Periodic: $x[n] = x[n + N]$
 - ▶ N **has no unit**, because it is just a number
 - ▶ $f = \frac{1}{N}$ **has no unit** also

Domain of existence of frequency

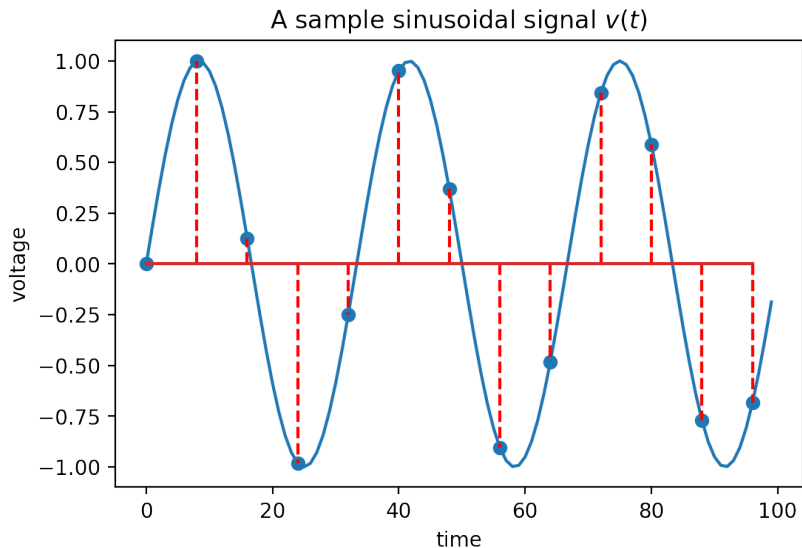
- ▶ Continuous signals
 - ▶ Period T can be as small as possible $T \rightarrow 0$
 - ▶ Therefore F could go up to ∞
- ▶ Analog signals
 - ▶ Smallest period is $N = 2$ (excluding $N = 1$, constant signals)
 - ▶ Largest possible frequency is $f_{max} = \frac{1}{2}$
 - ▶ Consequence of using natural numbers to index the samples ($x[0]$, $x[1]$, $x[2] \dots$), without any physical unit attached
- ▶ For mathematical reasons: we will consider negative frequencies as well (remember SCS)
 - ▶ they mirror the positive frequencies.

Sampling

- ▶ Taking the values from an analog signal at certain discrete moments of time, usually periodic
- ▶ Distance between two samples = **sampling period** T_s
- ▶ **Sampling frequency** $F_s = \frac{1}{T_s}$
- ▶ Why sampling?
 - ▶ Converts continuous signals to discrete
 - ▶ Processing of continuous signals is expensive
 - ▶ Processing of discrete signals is cheap (digital devices)
 - ▶ Sometimes nothing is lost due to sampling

Graphical example

<matplotlib.text.Text at 0x7f770c7b9400>



Sampling equation

- ▶ Sampling of the continuous signal x_a :

$$x[n] = x_a(n \cdot T_s)$$

- ▶ The n -th value of the discrete signal $x[n]$ is the value of the analog signal $x_a(t)$ taken after n sampling periods, at $t = n \cdot T_s$

Sampling of harmonic signals

- ▶ Let's sample a cosine: $x_a(t) = \cos(2\pi Ft)$

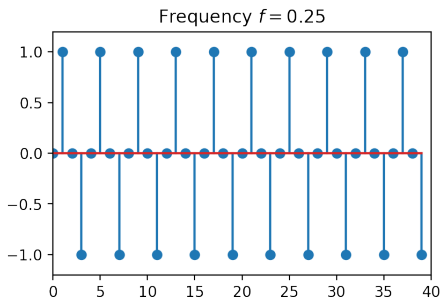
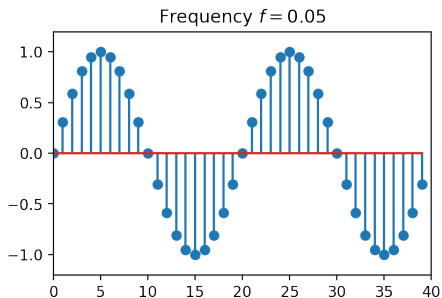
$$\begin{aligned}x[n] &= x_a(nT_s) \\&= \cos(2\pi FnT_s) \\&= \cos(2\pi Fn \frac{1}{F_s}) \\&= \cos(2\pi \underbrace{\frac{F}{F_s}}_f n)\end{aligned}$$

- ▶ Sampling a continuous cosine (or sine) produces a discrete cosine (or sine)
- ▶ The discrete frequency is $f = \frac{F}{F_s}$

False friends

- **Note:** A discrete sinusoidal signal might not *look* sinusoidal, when its frequency is high (close to $\frac{1}{2}$).

<matplotlib.text.Text at 0x7f770bedf9e8>



Sampling theorem (Nyquist-Shannon)

- ▶ If a signal that has maximum frequency F_{max} is sampled with a sampling frequency

$$F_s \geq 2F_{max},$$

- ▶ then it can be perfectly reconstructed from its samples using the formula:

$$x_a(t) = \sum_{n=-\infty}^{+\infty} x[n] \cdot \frac{\sin(\pi(F_s t - n))}{\pi(F_s t - n)}.$$

Comments on the sampling theorem

- ▶ All the information in the original signal is contained in the samples, provided the sampling frequency is high enough
- ▶ We can process discrete samples instead of the original analog signals
- ▶ Sampling with $F_s \geq 2F_{max}$ makes the discrete frequency smaller than $1/2$

$$f = \frac{F}{F_s} \leq \frac{F_{max}}{F_s} \leq \frac{1}{2}$$

Aliasing

- ▶ <http://www.dictionary.com/browse/alias>:
 - ▶ “alias”: a false name used to conceal one’s identity; an assumed name
- ▶ What happens when the sampling frequency is not high enough?
- ▶ Every discrete frequency that exceeds $f_{max} = \frac{1}{2}$ is **identical** (an alias) to a frequency that is lower than $f_{max} = \frac{1}{2}$
- ▶ Proof:
 - ▶ Consider $x[n] = \cos(2\pi fn)$, $f > \frac{1}{2}$
 - ▶ We can always subtract $2\pi n$ since $\cos()$ is periodical
 - ▶ This means reducing f with 1
 - ▶ Thus we can always end up a frequency $f' \in [-1/2, 1/2]$ (up to a sign change)

Aliasing (continued)

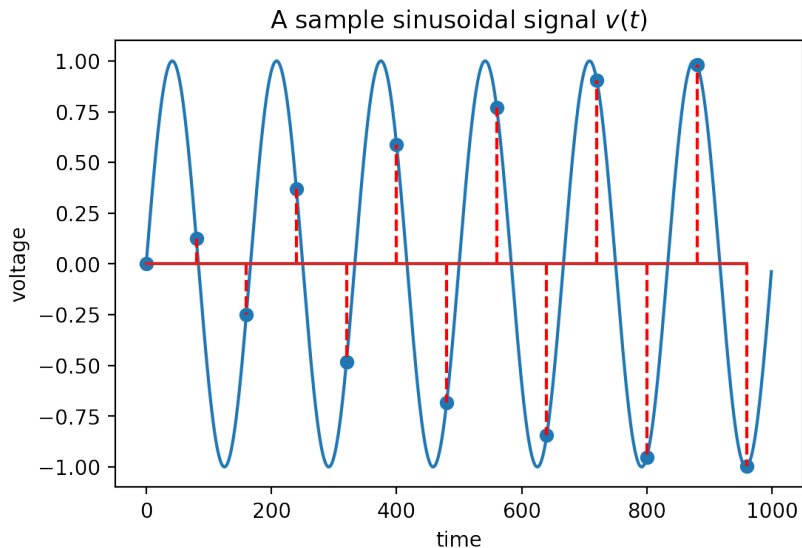
$$\cos(2\pi(\frac{1}{2} + \epsilon)n) = \cos(2\pi(\frac{1}{2} - \epsilon)n)$$

$$\sin(2\pi(\frac{1}{2} + \epsilon)n) = -\sin(2\pi(\frac{1}{2} - \epsilon)n)$$

- ▶ Aliasing only affects digital signals
- ▶ Sampling with $F_s \geq 2F_{max}$ ensures $f \leq \frac{1}{2}$, so no aliasing

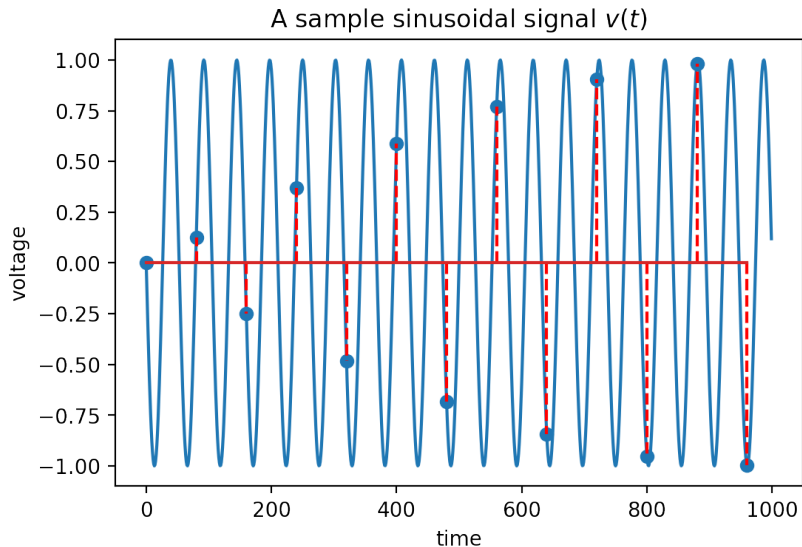
Aliasing example - low frequency signal

<matplotlib.text.Text at 0x7f770bd859b0>



Aliasing example - high frequency signal, same samples

<matplotlib.text.Text at 0x7f770d21d390>



The problem of aliasing

- ▶ Sampling different signals leads to exactly same samples
- ▶ How to know from what signal did the samples come from?
Impossible.
- ▶ Better remove from the signal the frequencies larger than $\frac{F_s}{2}$, otherwise they will create a false frequency and bring confusion
- ▶ Anti-alias filter: a low-pass filter situated before a sampling circuit, rejecting all frequencies $F > \frac{F_s}{2}$ from the signal before sampling
 - ▶ Standard practice in the design of processing systems

Signal reconstruction from samples

- ▶ A discrete frequency $f \in [-\frac{1}{2}, \frac{1}{2}]$ will be reconstructed as follows:

$$x_r(t) = x\left[\frac{t}{T_s}\right] = x[t \cdot F_s]$$

- ▶ For a discrete frequency outside the $[-\frac{1}{2}, \frac{1}{2}]$ interval
 - ▶ Reconstruction of the original frequency is impossible
 - ▶ The frequency is replaced with the aliased frequency f' from the interval $[-\frac{1}{2}, \frac{1}{2}]$
- ▶ Reconstruction always produces signals with frequencies in $[-\frac{F_s}{2}, \frac{F_s}{2}]$
- ▶ Only signals sampled according to the sampling theorem will be reconstructed identically

Signal quantization and coding

- ▶ In practice, the values of the samples are rounded to fixed levels, e.g. 8-bit, 16-bit values.
- ▶ This “rounding” is known as **quantization**
- ▶ The “rounding error” is known as **quantization error**
- ▶ Converting the value in binary form is known as **coding**

A/D and D/A conversion

- ▶ Sampling + quantization + coding is usually done by an **Analog to Digital Converter (ADC)**
 - ▶ It takes an analog signal and produces a sequence of binary-coded values
- ▶ Reconstructing an analog signal from numeric samples is done by a **Digital to Analog Converter (DAC)**
 - ▶ Usually reconstruction is not based on sampling theorem equation, which is too complex, but with simpler empiric approaches.