Advanced Signal Processing

What is signal processing?

Gustavo Andrade Miranda PhD

gxavier.andradem@ug.edu.ec

Facultad de Ingeniería en Sistemas, Electrónica e Industrial
Universidad Técnica de Ambato

PROGRAMA DE MAESTRÍA EN TELECOMUNICACIONES

Outline

Signal Processing I

Description of the evolution of a physical phenomenon

- weather → temperature
- sound → pressure
- light intensity → photo (gray level on paper)
- electromagnetic signal → radar

Signal Processing I

Description of the evolution of a physical phenomenon

- weather $\rightarrow$ temperature
- sound $\rightarrow$ pressure
- light intensity $\rightarrow$ photo (gray level on paper)
- electromagnetic signal $\rightarrow$ radar

Signal Processing I

Description of the evolution of a physical phenomenon

- weather $\rightarrow$ temperature
- sound $\rightarrow$ pressure
- light intensity $\rightarrow$ photo (gray level on paper)
- electromagnetic signal $\rightarrow$ radar

Signal Processing I

Description of the evolution of a physical phenomenon

- weather $\rightarrow$ temperature
- sound $\rightarrow$ pressure
- light intensity $\rightarrow$ photo (gray level on paper)
- electromagnetic signal $\rightarrow$ radar

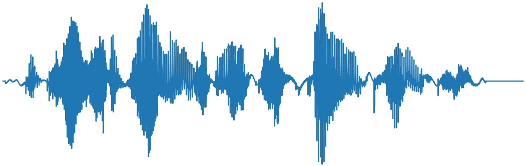
Signal Processing I

Description of the evolution of a physical phenomenon

- weather $\rightarrow$ temperature
- sound $\rightarrow$ pressure
- light intensity $\rightarrow$ photo (gray level on paper)
- electromagnetic signal $\rightarrow$ radar

Signal Processing II

Speech



$f(t) = ??$

Signal Processing III

Analysis

Understanding the information carried by the signal

Synthesis

Creating a signal to contain the given information

Outline

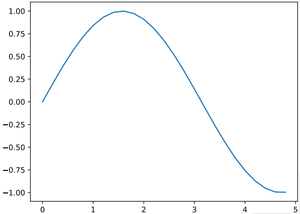
Digital Paradigm

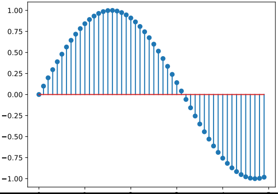
Key Ingredients

- discrete time
- discrete amplitude

Discrete Time I

$$\bar{x} = \frac{1}{b-a}\int_b^a f(t)dt$$





$$\bar{x}[n] = \frac{1}{N}\sum_{n=0}^{N-1} x[n]$$

- Extremely easy to compute

Discrete Time II

Can we describe the physical reality using a discrete time sequence??

- sampling theorem (1920)
- under appropiate "slowness" conditions in $x(t)$ we have:

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] sinc\left(\frac{t - nT_s}{T_s}\right)$$

- $x(t)$ can be written as linear combination of *sinc* blocks

Discrete Time II

Can we describe the physical reality using a discrete time sequence??

- sampling theorem (1920)
- under appropiate "slowness" conditions in $x(t)$ we have:

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] sinc\left(\frac{t - nT_s}{T_s}\right)$$

- $x(t)$ can be written as linear combination of *sinc* blocks

Discrete Time II

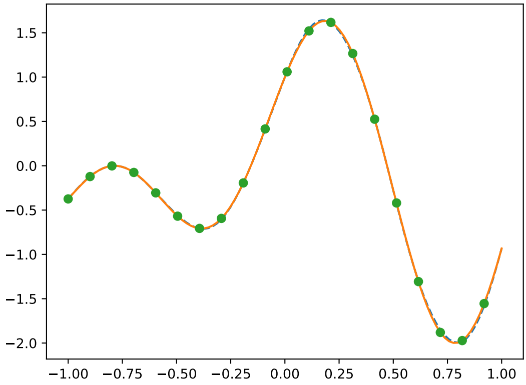
Can we describe the physical reality using a discrete time sequence??

- sampling theorem (1920)
- under appropiate "slowness" conditions in $x(t)$ we have:

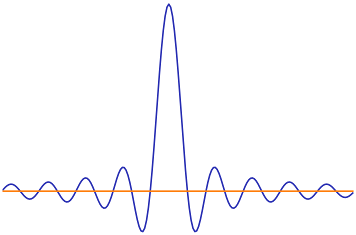
$$x(t) = \sum_{n=-\infty}^{\infty} x[n] sinc\left(\frac{t - nT_s}{T_s}\right)$$

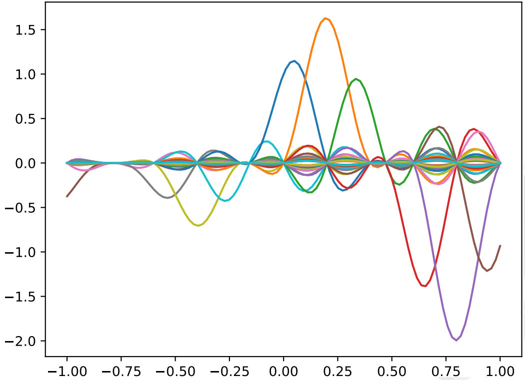
- $x(t)$ can be written as linear combination of *sinc* blocks

Discrete Time III

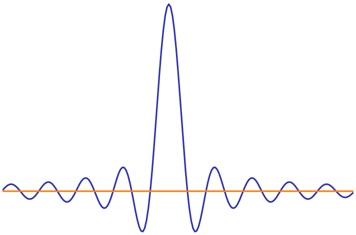


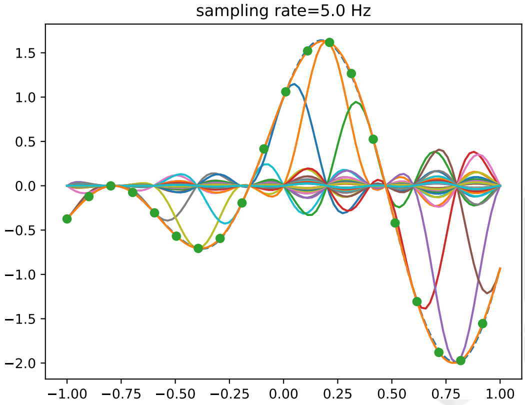
Discrete Time III





Discrete Time III





Discrete Time IV

- sufficiently “slow” with respect to how fast we sample
- sampling theorem links the speed of sampling with the maximum frequency
- spectra is computed with the Fourier transform

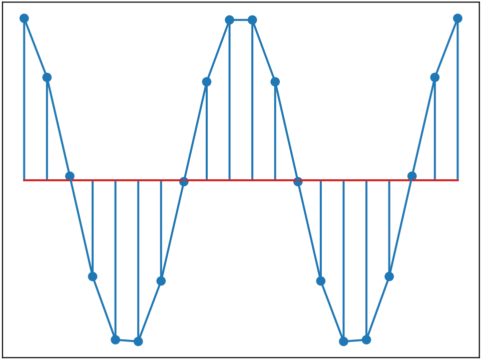
Discrete Time IV

- sufficiently "slow" with respect to how fast we sample
- sampling theorem links the speed of sampling with the maximum frequency
- spectra is computed with the Fourier transform

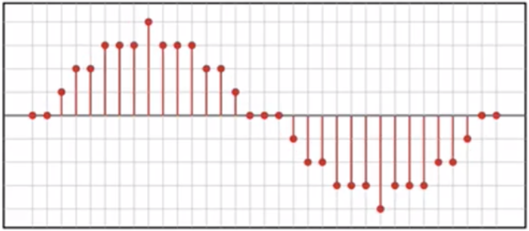
Discrete Time IV

- sufficiently “slow” with respect to how fast we sample
- sampling theorem links the speed of sampling with the maximum frequency
- spectra is computed with the Fourier transform

Discrete Amplitude I



Discrete Amplitude II



Discrete Amplitude III

Quantization

Deals with the problem of the continuum in a much “rougher” way that in the case of time: we simply accept a loss of precision with respect to the ideal model.

Outline

Why digital signals??

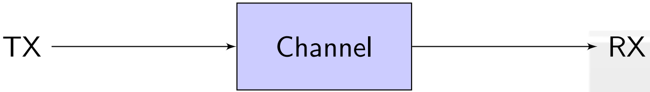
- storage
- processing
- transmission

Storage

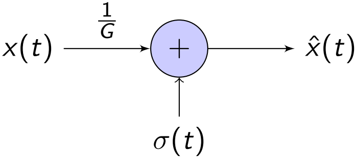
Processing - Analog vs Digital

- In Analogs signals, you had to build specific devices
- Mechanical systems require the design of very complexes gears
- Sound equalization system required discrete electronics
- With digital signals, all you need is a piece of code that will run on a generall purpose architecture
- We do not need to specialize in particular devices.

Data Transmission

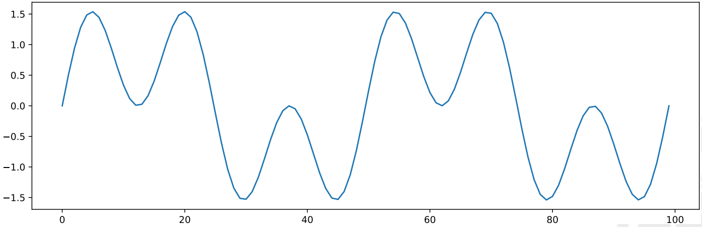


Data Transmission - analog signals



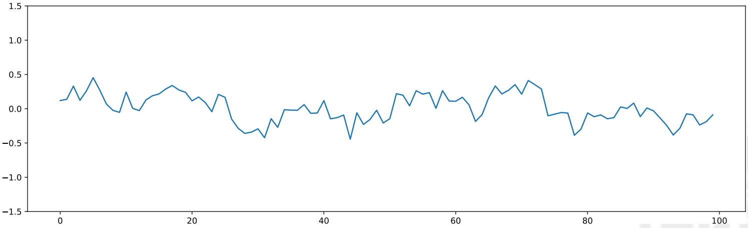
$$\hat{x}(t) = \frac{x(t)}{G} + \sigma(t)$$

Data Transmission - analog signals



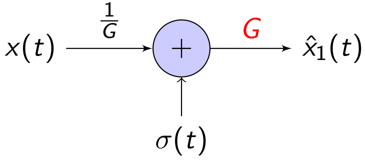
$$x(t)$$

Data Transmission - analog signals



$$\hat{x}(t) = x(t)\frac{1}{G} + \sigma(t)$$

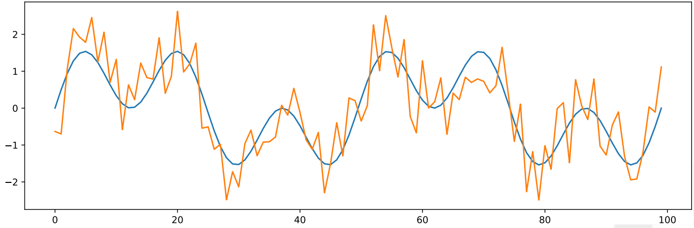
Data Transmission - analog signals



$$\hat{x}_1(t) = \left(\frac{x(t)}{G} + \sigma(t)\right) G$$

$$\hat{x}_1(t) = x(t) + G\sigma(t)$$

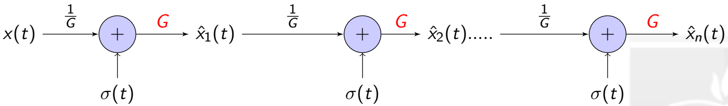
Data Transmission - analog signals



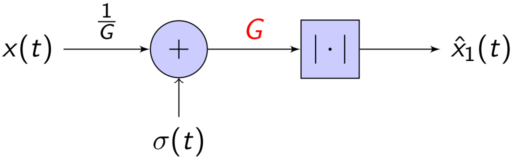
$$x(t)$$

$$\hat{x}_1(t) = x(t) + G\sigma(t)$$

Analog Transmission - long distances

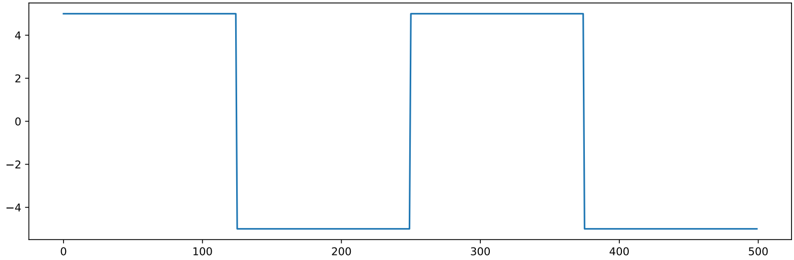


Digital Transmission



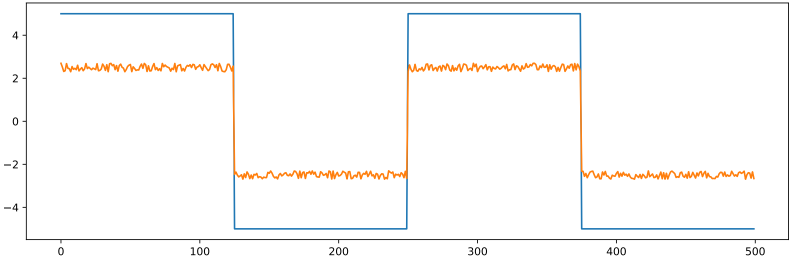
$$\hat{x}_1(t) = sgn[x(t) + G\sigma(t)]$$

Digital Transmission:Original Signal



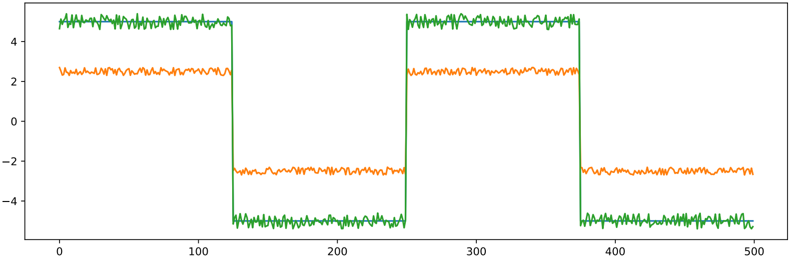
$x(t)$

Digital Transmission: Noise + Attenuation



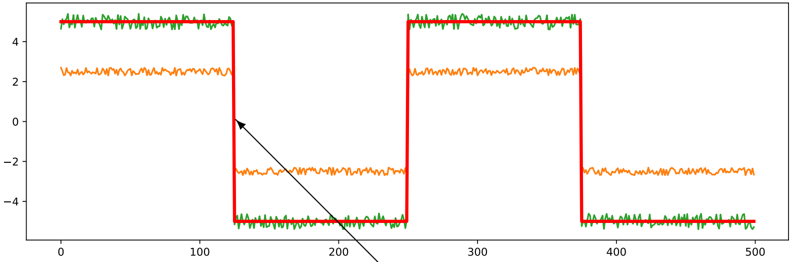
$$\hat{x}(t) = x(t)\frac{1}{G} + \sigma(t)$$

Digital Transmission: Attenuation correction



$$\hat{x}_1(t) = x(t) + G\sigma(t)$$

Digital Transmission: Attenuation correction



$$\hat{x}_1(t) = sgn[x(t) + G\sigma(t)]$$

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

Key ideas

- Discretization of time:
  - samples replaces idealized models
  - simple math replaces calculus
- Discretization of values:
  - general-purpose storage
  - general-purpose processing (CPU)
  - noise can be controlled

PYTHON - PRACTICE

- Introduction to Python
- Data Transmission

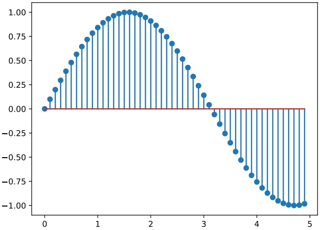
Outline

UNIVERSIDAD
TÉCNICA DE AMBATO

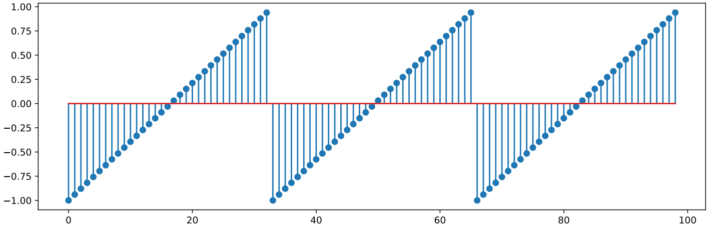
Basic Definition

Definition

A discrete-time signal is a complex-value function of an integer index n, with $n \in \mathbb{Z}$



Triangular Signal



Formal definition

- One or more dimension
- notation: $x[n]$
- two-sided sequences x: $\mathbb{Z} \to \mathbb{C}$
- n is a-dimensional "time"
- analysis: periodic measurement
- synthesis: stream of generated samples

Formal definition

- One or more dimension
- notation: $x[n]$
- two-sided sequences x: $\mathbb{Z} \rightarrow \mathbb{C}$
- n is a-dimensional "time"
- analysis: periodic measurement
- synthesis: stream of generated samples

Formal definition

- One or more dimension
- notation: $x[n]$
- two-sided sequences x: $\mathbb{Z} \rightarrow \mathbb{C}$
- n is a-dimensional "time"
- analysis: periodic measurement
- synthesis: stream of generated samples

Formal definition

- One or more dimension
- notation: $x[n]$
- two-sided sequences x: $\mathbb{Z} \rightarrow \mathbb{C}$
- n is a-dimensional "time"
- analysis: periodic measurement
- synthesis: stream of generated samples

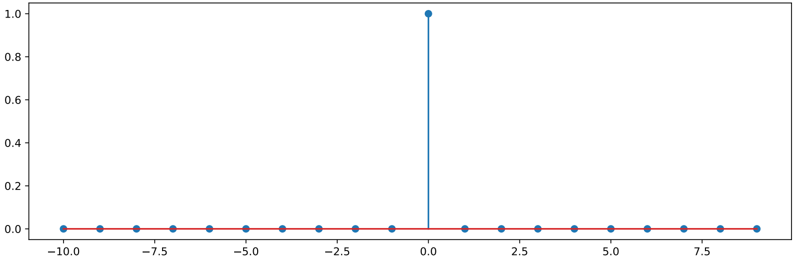
Formal definition

- One or more dimension
- notation: $x[n]$
- two-sided sequences x: $\mathbb{Z} \to \mathbb{C}$
- n is a-dimensional “time”
- analysis: periodic measurement
- synthesis: stream of generated samples

Formal definition

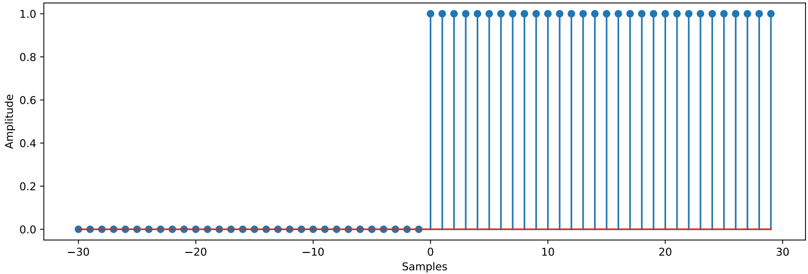
- One or more dimension
- notation: $x[n]$
- two-sided sequences x: $\mathbb{Z} \to \mathbb{C}$
- n is a-dimensional "time"
- analysis: periodic measurement
- synthesis: stream of generated samples

Delta Signal



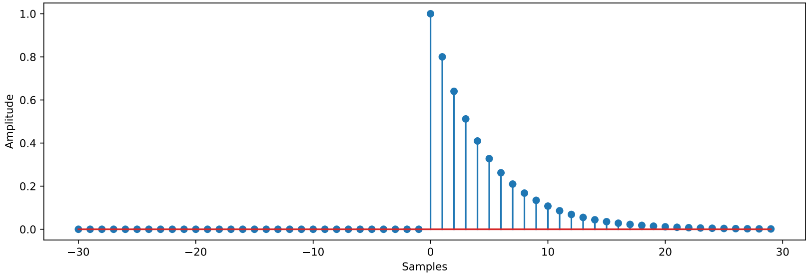
$$x[n] = \delta[n]$$

Unit Step

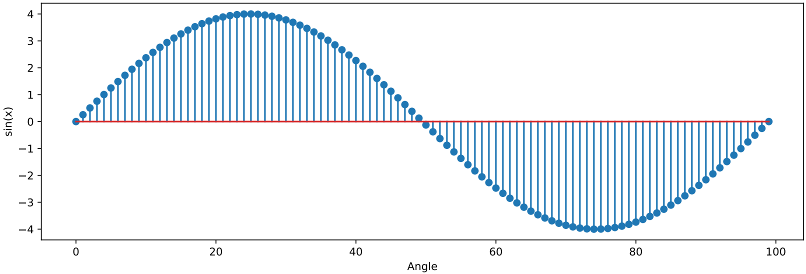


$$x[n] = u[n]$$

Exponential Decay



$$x[n] = |a|^n u[n], \ \ |a| < 1$$



$$x[n] = sin(w_o n + \theta) \;\; w_o \rightarrow [rad] \;\; \theta \rightarrow [rad]$$

Four signal classes

- finite-length
- infinite-length
- periodic
- finite-support

Four signal classes

- finite-length
- infinite-length
- periodic
- finite-support

Four signal classes

- finite-length
- infinite-length
- periodic
- finite-support

Four signal classes

- finite-length
- infinite-length
- periodic
- finite-support

Finite-length

- sequence notations $x[n], \; n = 0, 1, ..., N-1$
- $\mathbf{x} = [\mathbf{x}_0 \; \mathbf{x}_1 \; \ldots \; \mathbf{x}_{\mathbf{n}-1}]$
- practial entities, good for numerical packages (Numpy)

Finite-length

- sequence notations $x[n],\ n = 0, 1, ..., N - 1$
- $\mathbf{x} = [\mathbf{x_0}\ \mathbf{x_1}\ \dots\ \mathbf{x_{n-1}}]$
- practial entities, good for numerical packages (Numpy)

Finite-length

- sequence notations $x[n], \; n = 0, 1, ..., N - 1$
- $\mathbf{x} = [\mathbf{x_0} \; \mathbf{x_1} \; ... \; \mathbf{x_{n-1}}]$
- practial entities, good for numerical packages (Numpy)

Infinite-length

- sequence notations x[n], $n \in \mathbb{Z}$
- abstractions, good for theorems

Infinite-length

- sequence notations x[n], $n \in \mathbb{Z}$
- abstractions, good for theorems

Periodic

- N-periodic sequence: $\tilde{x}[n] = \tilde{x}[n + kN] \;\; n, k, N \in \mathbb{Z}$
- same information as finite-length of length N
- the natural bridge between finite and infinite

Periodic

- N-periodic sequence: $\tilde{x}[n] = \tilde{x}[n + kN] \;\; n, k, N \in \mathbb{Z}$
- same information as finite-length of length N
- the natural bridge between finite and infinite

Periodic

- N-periodic sequence: $\tilde{x}[n] = \tilde{x}[n + kN] \; n, k, N \in \mathbb{Z}$
- same information as finite-length of length N
- the natural bridge between finite and infinite

Finite-support

- Finite support sequence:

$$\bar{x}[n] = \begin{cases} x[n] & \text{if } 0 \leq n < N \\ 0 & \text{otherwise} \end{cases}$$

- same information as finite-length of length N
- another bridge betweeb finite and infinite lengths

Finite-support

- Finite support sequence:

$$\bar{x}[n] = \begin{cases} x[n] & \text{if } 0 \leq n < \text{N} \\ 0 & \text{otherwise} \end{cases}$$

- same information as finite-length of length N
- another bridge betweeb finite and infinite lengths

Finite-support

- Finite support sequence:

$$\bar{x}[n] = \begin{cases} x[n] & \text{if } 0 \leq n < N \\ 0 & \text{otherwise} \end{cases}$$

- same information as finite-length of length N
- another bridge betweeb finite and infinite lengths

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
    - shift of a finite-length: fine-support
    - shift of a finite-length: periodic extension (circular shift)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
  - shift of a finite-length: fine-support
  - shift of a finite-length: periodic extension (circular shift)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
  - shift of a finite-length: fine-support
  - shift of a finite-length: periodic extension (circular shift)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
  - shift of a finite-length: fine-support
  - shift of a finite-length: periodic extension (circular shift)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
  - shift of a finite-length: fine-support
  - shift of a finite-length: periodic extension (circular)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
    - shift of a finite-length: fine-support
    - shift of a finite-length: periodic extension (circular shift)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
  - shift of a finite-length: fine-support
  - shift of a finite-length: periodic extension (circular shift)

Elementary Operators

- scaling: $y[n] = \alpha x[n]$
- sum: $y[n] = x[n] + z[n]$
- product: $y[n] = x[n] \cdot z[n]$
- integration

$$y[n] = \sum_{k=-\infty}^{n} x[k]$$

- differentiation $y[n] = x[n] - x[n-1]$
- shift by k (delay) $y[n] = x[n-k]$
  - shift of a finite-length: fine-support
  - shift of a finite-length: periodic extension (circular shift)

Energy and Power

$$E_x = \sum_{n=-\infty}^{+\infty} |x[n]|^2$$

$$P_x = \lim_{N \to \infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x[n]|^2$$

- The power of a signal measure its energy per unit of time

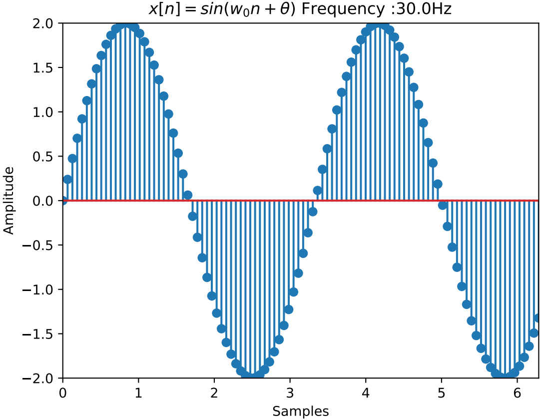
Energy and Power: Periodic Signals

$$E_{\tilde{x}} = \infty$$

$$P_x = \frac{1}{N} \sum_{n=0}^{N-1} |\tilde{x}[n]|^2$$

Outline

Discrete-Time Sinusoid



Digital vs Physical Frequency

- Discrete time:
  - n: no physical dimension (just a counter)
  - Periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

Digital vs Physical Frequency

- Discrete time:
  - n: no physical dimension (just a counter)
  - Periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

Digital vs Physical Frequency

- Discrete time:
  - n: no physical dimension (just a counter)
  - Periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

Digital vs Physical Frequency

- Discrete time:
  - n: no physical dimension (just a counter)
  - Periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

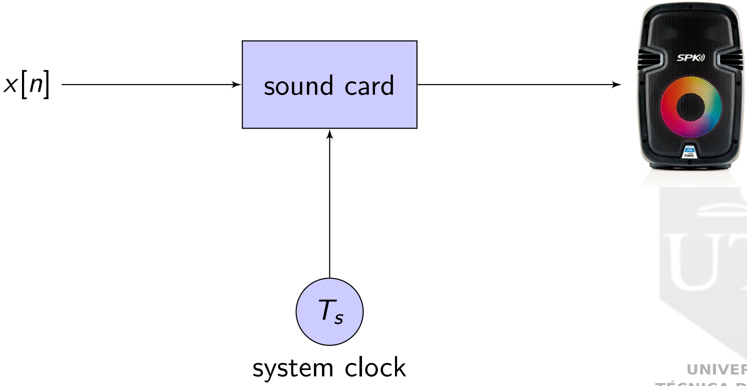
Digital vs Physical Frequency

- Discrete time:
  - n: no physical dimension (just a counter)
  - Periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

Digital vs Physical Frequency

- Discrete time:
  - n: no physical dimension (just a counter)
  - Periodicity: how many samples before pattern repeats
- Physical world:
  - periodicity: how many seconds before pattern repeats
  - frequency measured in Hz (s^{-1})

PC playing sounds



Digital vs Physical Frequency

- set T_s, time in seconds between samples
- periodicity of M samples (digital domain) $\rightarrow$ periodicity of MT_s seconds (physical domain)
- real world frequency:

$$f = \frac{1}{MT_s} Hz$$

Digital vs Physical Frequency

- Usually we use F_s the number of samples per seconds
- $T_s = 1/F_s$
- For a typical $F_s = 48000, \; T_s \approx 20.8\mu s$. If $M = 110$ $f \approx 440Hz$

Digital vs Physical Frequency

- Usually we use F_s the number of samples per seconds
- $T_s = 1/F_s$
- For a typical $F_s = 48000$, $T_s \approx 20.8\mu s$. If $M = 110$ $f \approx 440Hz$

Digital vs Physical Frequency

- Usually we use F_s the number of samples per seconds
- $T_s = 1/F_s$
- For a typical $F_s = 48000,\ \ T_s \approx 20.8\mu s$. If $M = 110$ $f \approx 440 Hz$

Outline

Introduction

Digital signals

- Discrete Time
- Discrete Amplitude

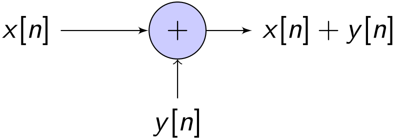
Why it is important

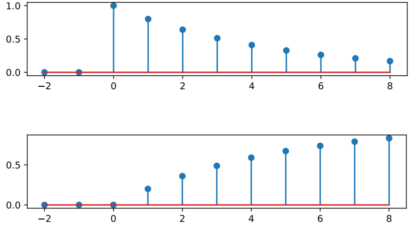
Discrete time signals

Plays Discrete-time sounds

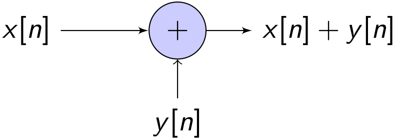
DSP as Building Blocks

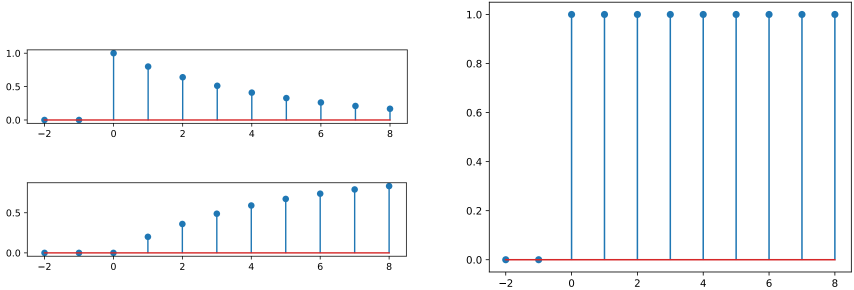
Building Blocks: adder





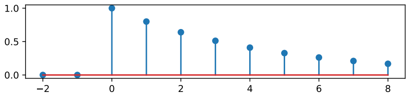
Building Blocks: adder





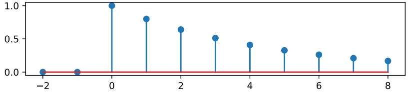
Building Blocks: multiplier

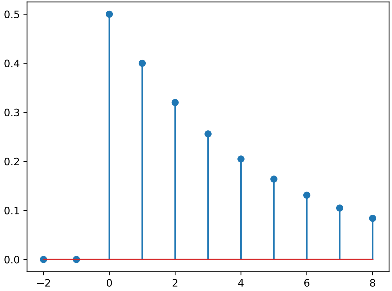
$$x[n] \xrightarrow{\quad\quad\alpha\quad\quad} \alpha x[n]$$



Building Blocks: multiplier

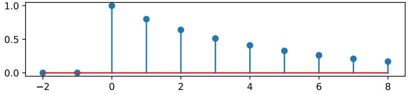
$$x[n] \xrightarrow{\quad\alpha\quad} \alpha x[n]$$





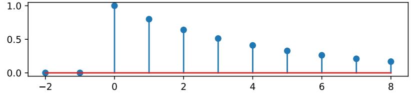
Building Blocks: Unit delay

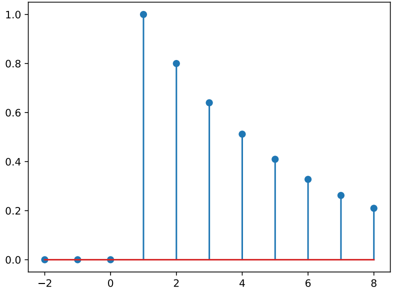
$$x[n] \longrightarrow \boxed{Z^{-1}} \longrightarrow x[n-1]$$



Building Blocks: Unit delay

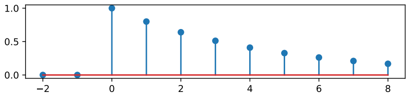
$$x[n] \longrightarrow \boxed{Z^{-1}} \longrightarrow x[n-1]$$





Building Blocks: Arbitrary delay

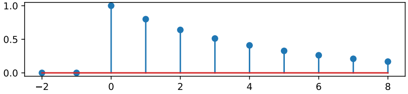
$$x[n] \longrightarrow \boxed{Z^{-N}} \longrightarrow x[n-N]$$

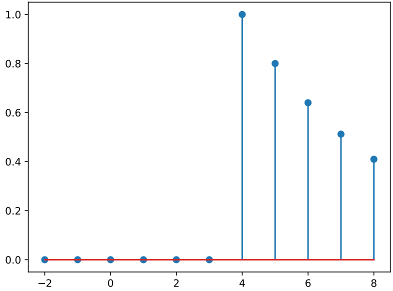


$N = 4$

Building Blocks: Arbitrary delay

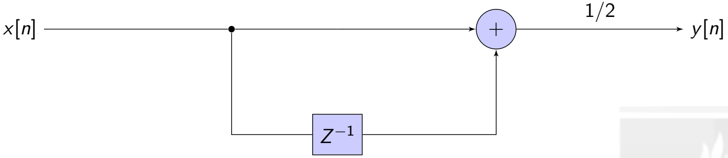
$$x[n] \longrightarrow \boxed{Z^{-N}} \longrightarrow x[n-N]$$





$N = 4$

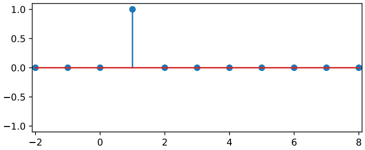
The 2-point Moving Average blocks



$$y[n] = \frac{x[n] + x[n-1]}{2}$$

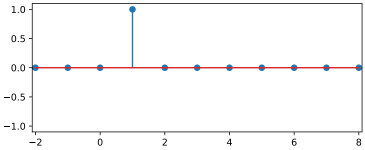
Average - Delta Signal

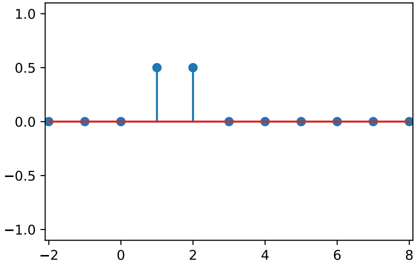
$$x[n] = \delta[n]$$



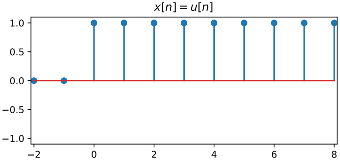
Average - Delta Signal

$$x[n] = \delta[n]$$

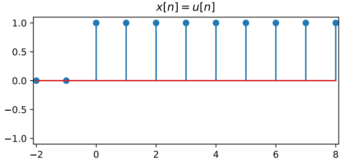


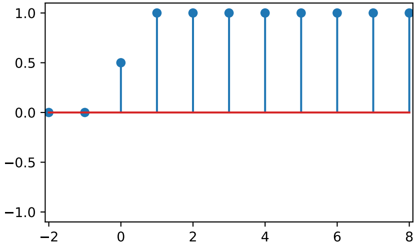


Average - Unit Step



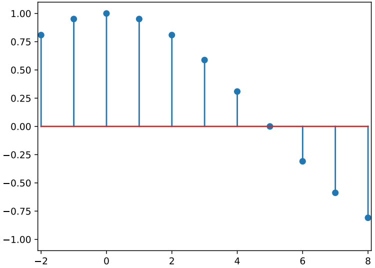
Average - Unit Step





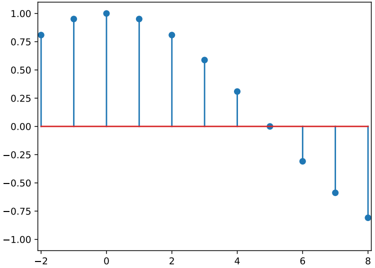
Average - Sinosoid Signals

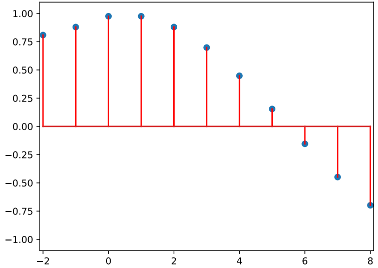
$$x[n] = cos(\omega n),\ \omega = \pi/10$$



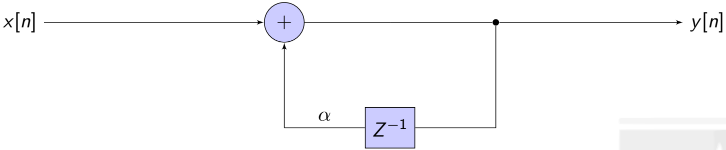
Average - Sinosoid Signals

$$x[n] = cos(\omega n),\ \omega = \pi/10$$



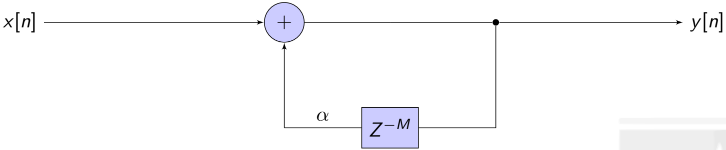


Inverse Loop



$$y[n] = x[n] + \alpha y[n-1]$$

Generalised Inverse Loop



$$y[n] = x[n] + \alpha y[n - M]$$

Make music with reverse Loop - Karplus Strong

- build a recursion loop with a delay of M
- choose a finite support signal $\bar{x}[n]$ that is nonzero only for $0 \leq n \leq M$
- choose a decay factor
- input $\bar{x}[n]$ to the system
- play the output

Make music with reverse Loop - Karplus Strong

- build a recursion loop with a delay of M
- choose a finite support signal $\bar{x}[n]$ that is nonzero only for $0 \leq n \leq M$
- choose a decay factor
- input $\bar{x}[n]$ to the system
- play the output

Make music with reverse Loop - Karplus Strong

- build a recursion loop with a delay of M
- choose a finite support signal $\bar{x}[n]$ that is nonzero only for $0 \leq n \leq M$
- choose a decay factor
- input $\bar{x}[n]$ to the system
- play the output

Make music with reverse Loop - Karplus Strong

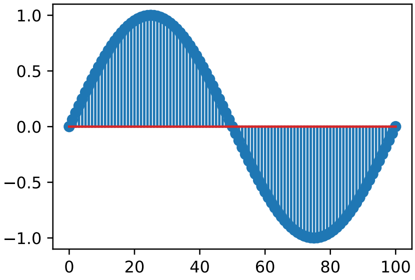
- build a recursion loop with a delay of M
- choose a finite support signal $\bar{x}[n]$ that is nonzero only for $0 \leq n \leq M$
- choose a decay factor
- input $\bar{x}[n]$ to the system
- play the output

Make music with reverse Loop - Karplus Strong

- build a recursion loop with a delay of M
- choose a finite support signal $\bar{x}[n]$ that is nonzero only for $0 \leq n \leq M$
- choose a decay factor
- input $\bar{x}[n]$ to the system
- play the output

Karplus-Strong - Playing a sine wave

$M = 100$, $\alpha = 1$, $\bar{x}[n] = sin(2\pi n/100)$ for $0 \leq n \leq 100$ and zero elsewhere

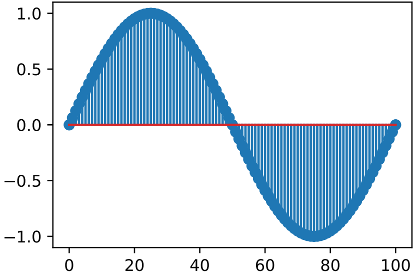


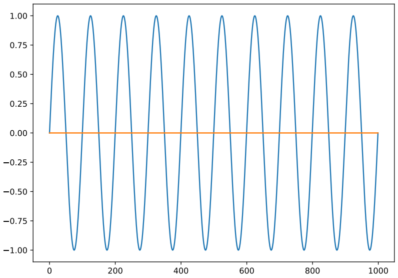
$w = \frac{2\pi}{100}$

$F_s = 48KHz,\ M = \frac{F_s}{f_{real}},\ f_{real} = 480Hz$

Karplus-Strong - Playing a sine wave

$M = 100$, $\alpha = 1$, $\bar{x}[n] = sin(2\pi n/100)$ for $0 \leq n \leq 100$ and zero elsewhere





$w = \frac{2\pi}{100}$

$F_s = 48KHz,\ M = \frac{F_s}{f_{real}},\ f_{real} = 480Hz$

Karplus-Strong - Making realistic

- M controls frequency (pitch)
- α controls envelope (decay)
- $\bar{x}[n]$ controls color (timbre)

PYTHON - PRACTICE

- Karplus-Strong Algorithm