Disciplina: Processamento Digital de Sinais

Material aula 2
Ambiente Blackboard

Apresentação

- 1) Sinais discretos básicos
- 2) Propriedades de sistemas discretos
- 3) Representação por diagrama de blocos

4) Implementação de algoritmos:

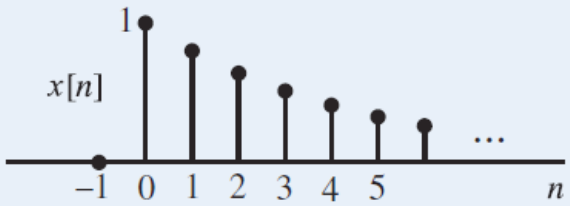
  - Média Móvel

    Delay

    Eco

Representações de um sinal discreto

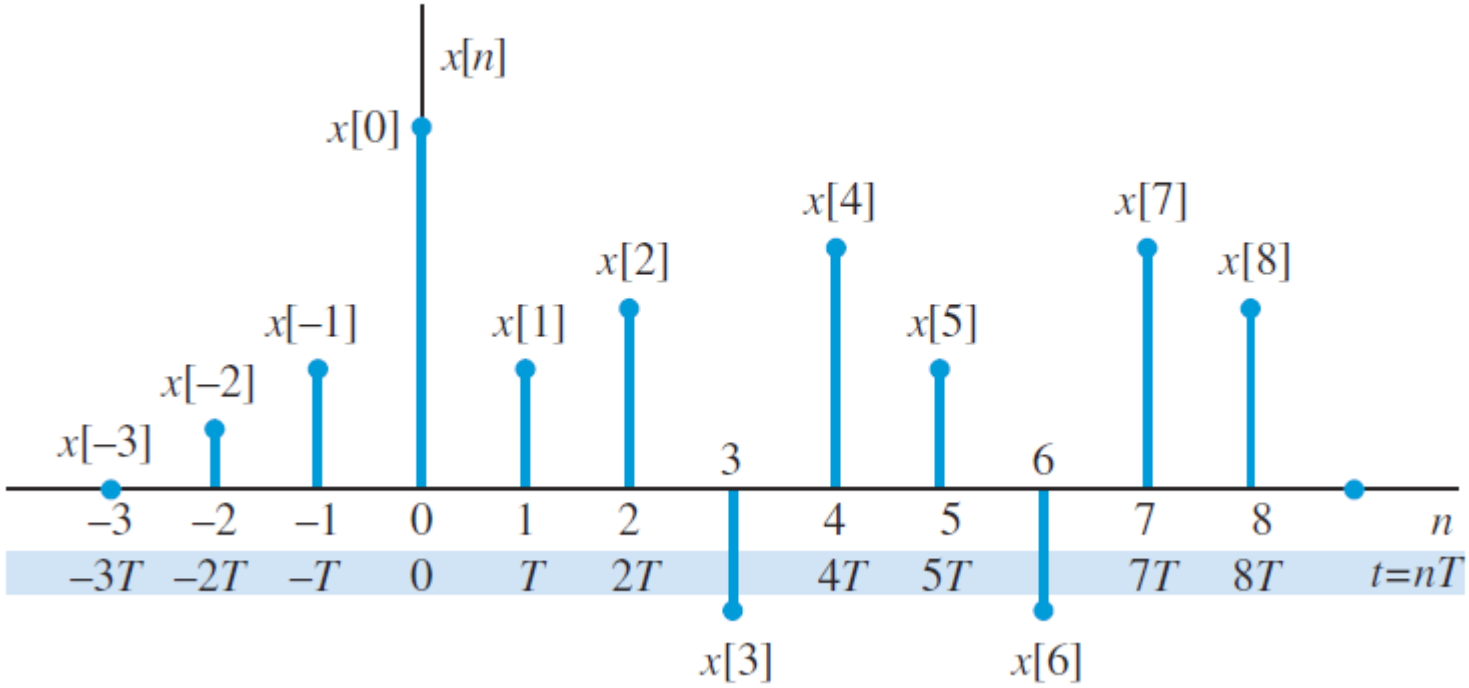
Table 2.1 Discrete-time signal representations.

Representation	Example
Functional	$x[n] = \begin{cases} \left(\frac{1}{2}\right)^n, & n \geq 0 \\ 0, & n < 0 \end{cases}$
Tabular	$\begin{array}{c\|ccccccc} n & \ldots & -2 & -1 & 0 & 1 & 2 & 3 & \ldots \\ \hline x[n] & \ldots & 0 & 0 & 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} & \ldots \end{array}$
Sequence	$x[n] = \{ \ldots \; 0 \; \underset{\uparrow}{1} \; \frac{1}{2} \; \frac{1}{4} \; \frac{1}{8} \; \ldots \}$
Pictorial	

[1] The symbol ↑ denotes the index $n = 0$; it is omitted when the table starts at $n = 0$.

Sinal discreto no tempo

2.1 Discrete-time signals



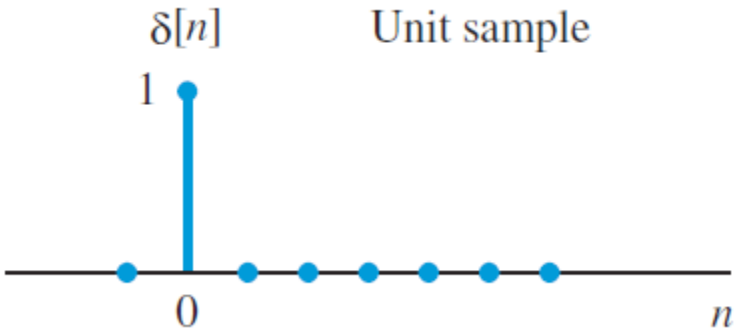
Sinais discretos básicos

- Impulso Unitário
- Degrau Unitário
- Sequência Sinusoidal
- Sequência Exponencial

Sinais discretos básicos

- Impulso Unitário

$$\delta[n] \triangleq \begin{cases} 1, & n = 0 \\ 0. & n \neq 0 \end{cases}$$

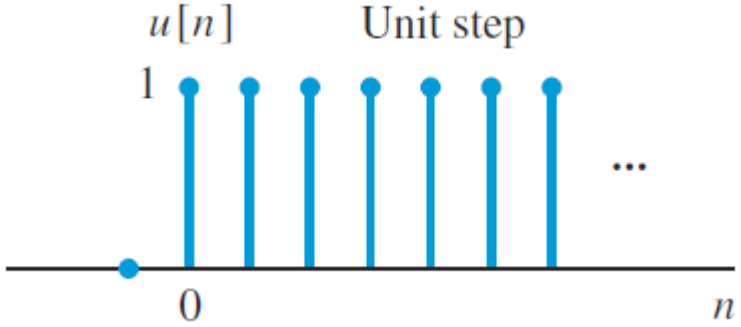


OBS: Implementar a função impulso unitário em Matlab

Sinais discretos básicos

Degrau Unitário

$$u[n] \triangleq \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases}$$

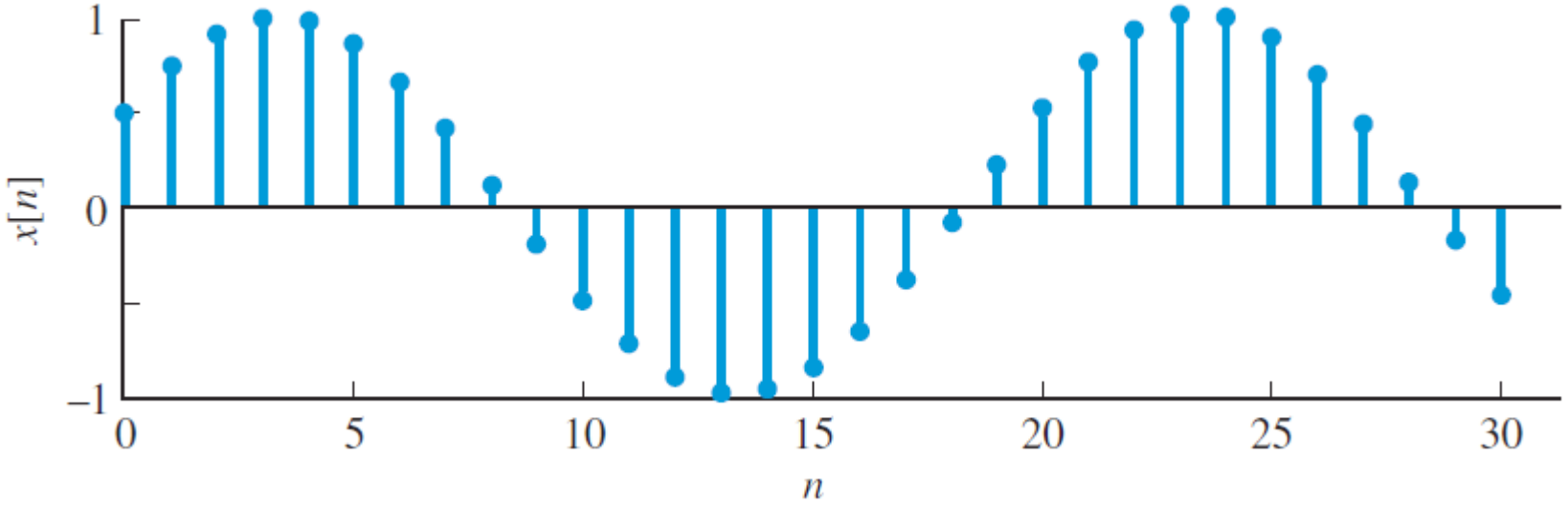


OBS: Implementar a função degrau em Matlab

Sinais discretos básicos

Sequência Sinusoidal

$$x[n] = A\cos(\omega_0 n + \phi), \quad -\infty < n < \infty$$

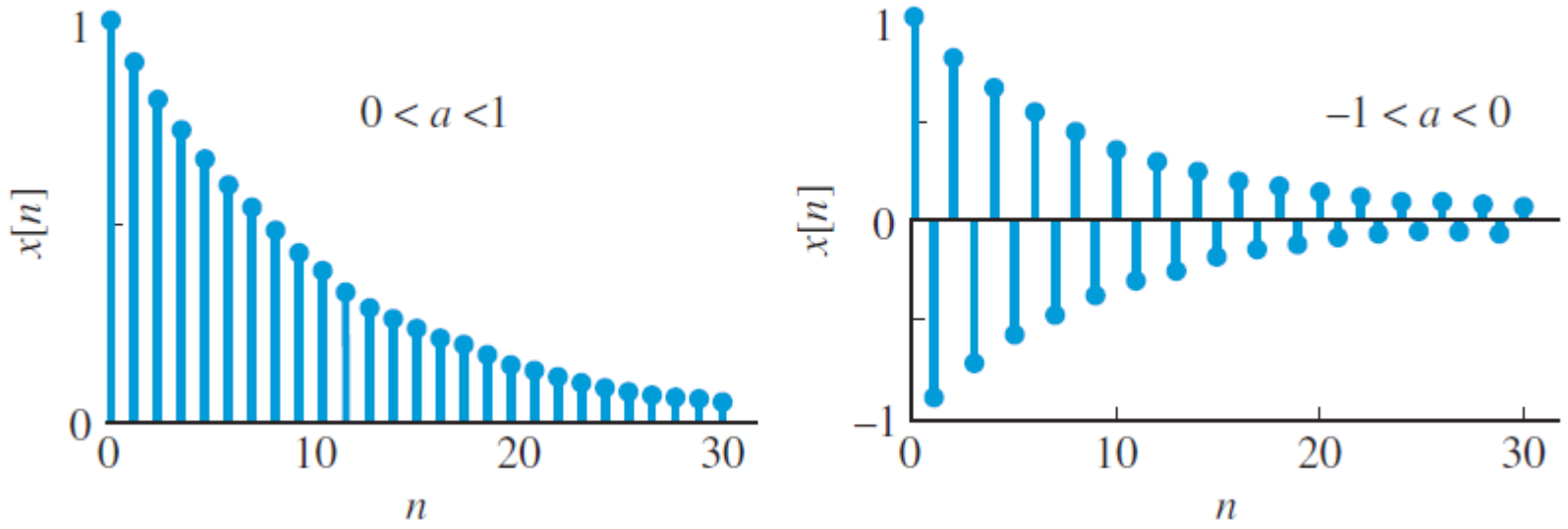


OBS: Implementar a função senoidal em Matlab com fo = 100Hz e FS = 8kHz e duração de 1 segundo

Sinais discretos básicos

Sequência Exponencial

$$x[n] \triangleq Aa^n, \quad -\infty < n < \infty$$



OBS: Implementar a sequência exponencial em Matlab com:
A = 1, a = 0, 5, a = -0,5 e a = 2

Sistema Discreto

2.3 Discrete-time systems

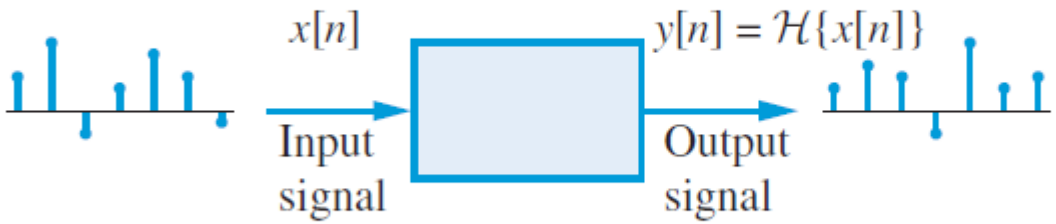


Figure 2.5 Block diagram representation of a discrete-time system.

Propriedades de sistemas discretos

- Causalidade
- Estabilidade
- Linearidade
- Invariante no tempo

Propriedades de sistemas discretos

- Causalidade
- **Definition**: A system is called *causal if the present value of the output does not depend* on future values of the input, that is, *y[n0] is determined by the values of x[n] for n ≤ n0,* only.

Causal

$$y[n] = \frac{1}{3}\{x[n] + x[n-1] + x[n-2]\}$$

Não causal

$$y[n] = \text{median}\{x[n-1], x[n-2], x[n], x[n+1], x[n+2]\}.$$

Propriedades de sistemas discretos

- Estabilidade
- Definition : A system is said to be *stable, in the Bounded-Input Bounded-Output* (BIBO) sense, if every bounded input signal results in a bounded output signal, that is
  $|x[n]| \leq Mx < \infty \Rightarrow |y[n]| \leq My < \infty$.

Estável $$y[n] = \frac{1}{3}\{x[n] + x[n-1] + x[n-2]\}$$

Instável $$y[n] = \sum_{k=0}^{\infty} x[n-k]$$

Para x[n] = u[n]

Propriedades de sistemas discretos

- Linearidade
- Definition: A system is called *linear if and only if for every real or complex constant a1, a2 and every input signal x1[n] and x2[n]*
- *H{a1x1[n] + a2x2[n]} = a1H{x1[n]} + a2H{x2[n]},*

- *Exemplo não linear: y[n] = x^2[n].*

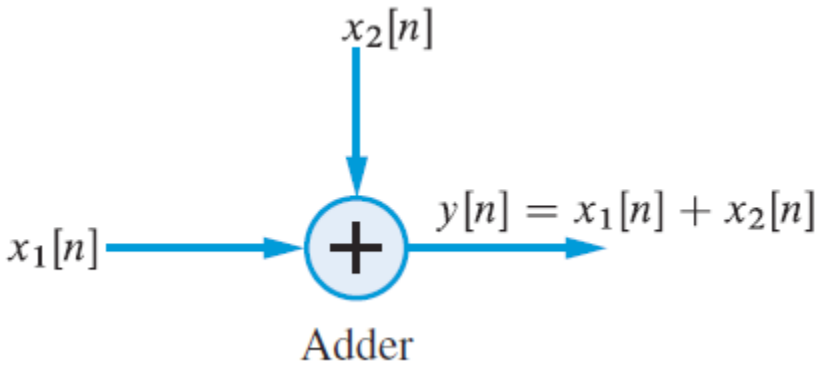
Propriedades de sistemas discretos

- Invariante no tempo
- Definition: A system is called *time-invariant or fixed if and only if*
- $y[n] = H\{x[n]\} \Rightarrow y[n - n0] = H\{x[n - n0]\}$,

- for every input *x[n] and every time shift n0. That is, a time shift in the input results in a* corresponding time shift in the output.

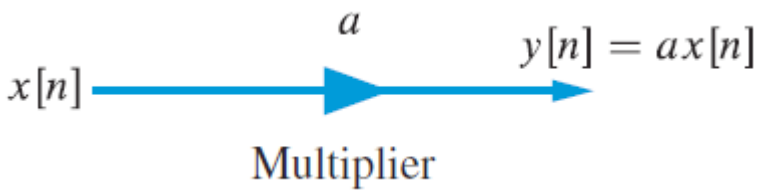
Representação por diagrama de blocos

- Soma



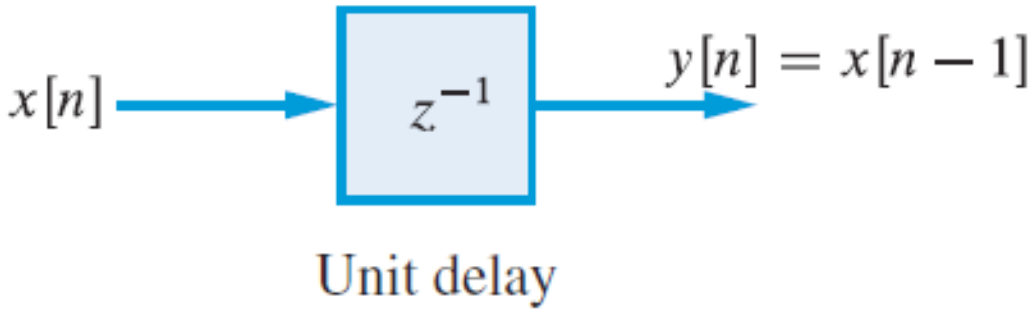
Representação por diagrama de blocos

- Multiplicação por uma constante



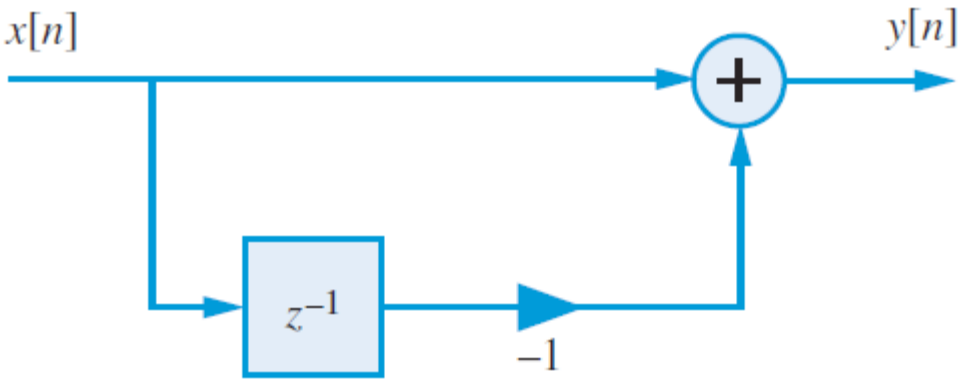
Representação por diagrama de blocos

- Atraso unitário



Representação por diagrama de blocos

- Sistema discreto



Implementação de algoritmos

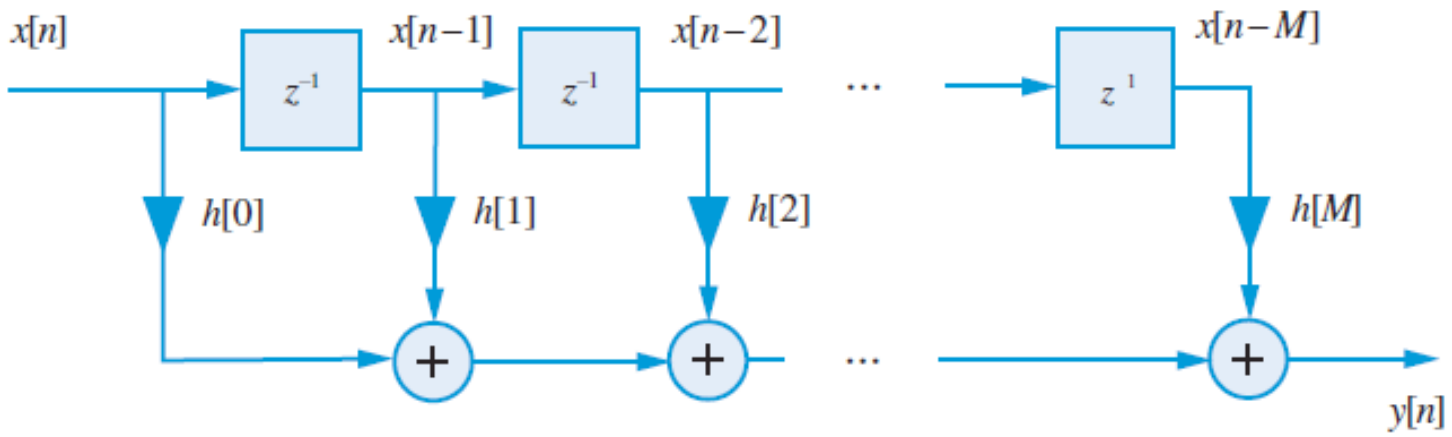
- Média Móvel de k elementos da entrada
- Por exemplo, para k = 4, temos:

$$y[n] = \frac{x[n] + x[n-1] + x[n-2] + x[n-3]}{4}$$

$$y[n] = 1/4 \,.\, x[n] + 1/4 \,.\, x[n-1] + 1/4 \,.\, x[n-2] + 1/4 \,.\, x[n-3]$$

Implementação de algoritmos

- Média Móvel de k elementos da entrada



Implementação Média Móvel

Arquivo de entrada

10
2
8
-4
-2
3

Vetor x[n] dimensão k,1

0	10	2	8	-4	-2	3
0	0	10	2	8	-4	-2
0	0	0	10	2	8	-4
0	0	0	0	10	2	8

Vetor Coefs Dimensão k,1

1/k
1/k
1/k
1/k

Arquivo de saída

2,5
3
5
4

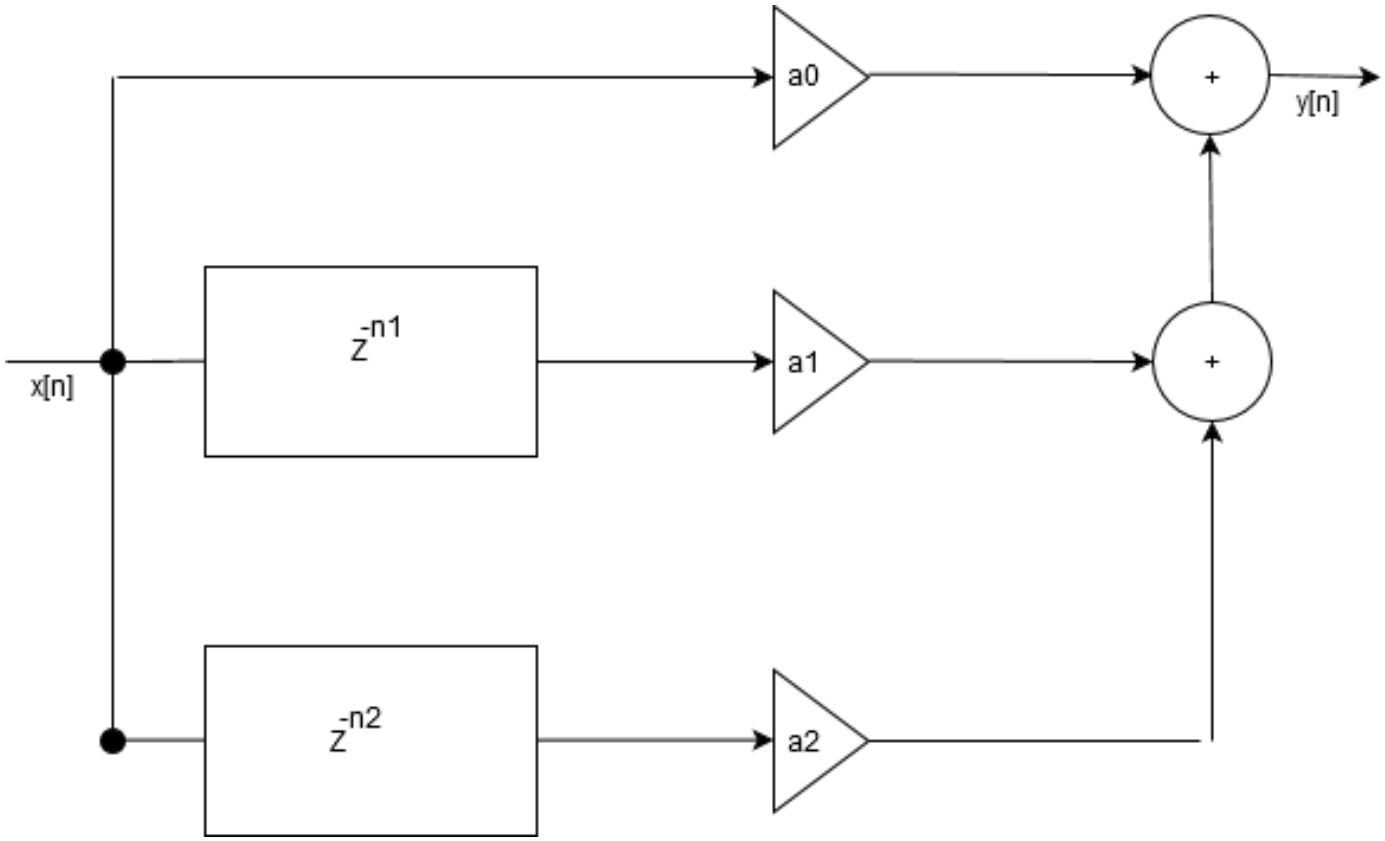
Implementação de algoritmos:

- Média Móvel - Matlab /Python
- Média Móvel - Linguagem C
- Usar como entrada um sweep(20Hz a 3400Hz) gerado no ocenaudio
- Testar com diferentes tamanhos de média e salvar o arquivo de saída, por exemplo, k = 4, 8, 16, 32, etc

Implementação de algoritmos:

- Média Móvel - Matlab /Python
- Usar como entrada o impulso unitário e obtenha a correspondente saída.
- Usar como entrada o degrau unitário e obtenha a correspondente saída.

Implementação do Delay



Implementação do Delay

- Implemente um programa em Matlab/Python que execute a equação do diagrama de blocos.

$$y[n] = a0.x[n] + a1.x[n - n1] + a2.x[n - n2]$$

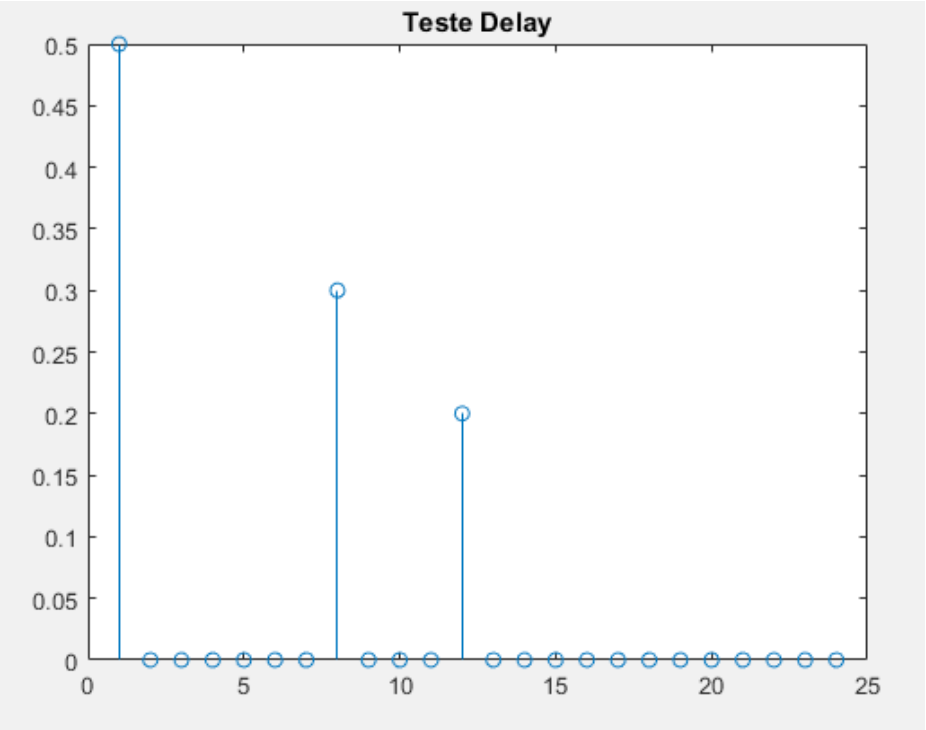
Implementação do Delay

- Utilize Fs= 8k, n1 = correspondendo a um atraso de 10ms e n2 de 15ms.
- Fs = 8k, Ts = 125us.
- Portanto, n1 corresponde a 80 amostras de delay e n2 120 amostras.
-
- Use a0 = 0,5; a1 = 0,3 e a2 = 0,2.

- Valide o algoritmo para uma entrada impulso unitário

Implementação do Delay

- Para uma entrada impulso unitário, n1 de 8 amostras de delay e n2 de 12 amostras, temos:



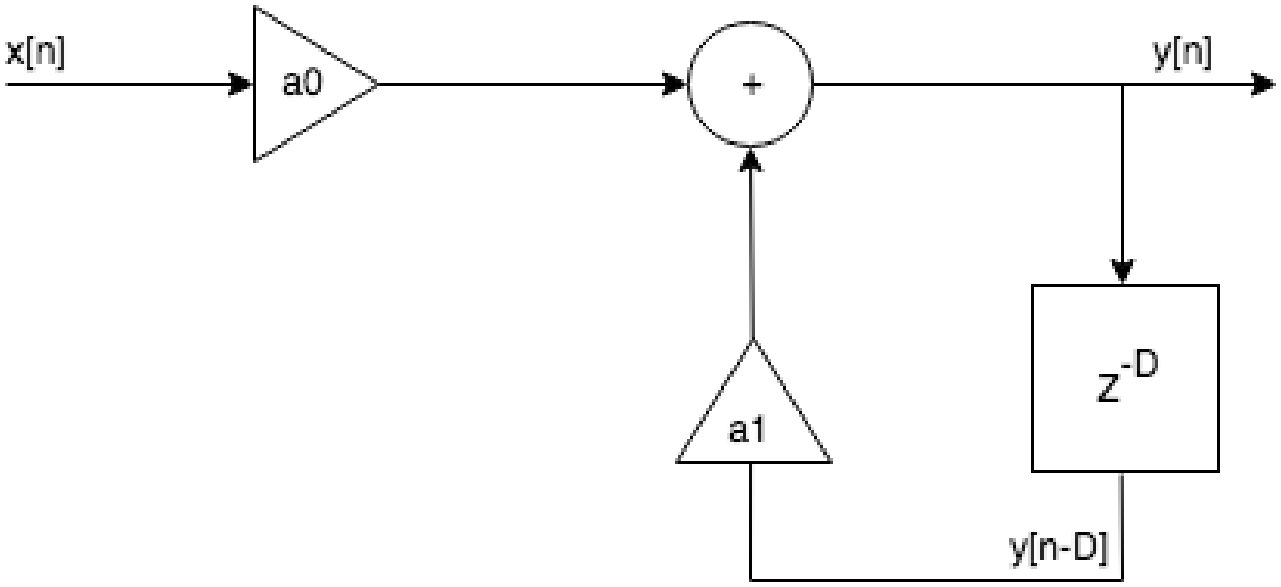
Implementação do Delay

```
clear all; close all; clc;
Fs = 8000;
t1 = 1.0*10^-3; t2 = 1.5*10^-3;
% Exemplo delay
n1 = t1*Fs; n2 = t2*Fs
% Definição dos ganhos
a0 = .5; a1 = .3; a2 = .2;
tama_delay = n2;
vetor_delay = zeros(tama_delay ,1);
% Definindo a entrada
entrada = zeros(2*tama_delay,1);


entrada(1,1) = 1; % definindo o impulso     unitário

tama_loop = length(entrada);
vet_saida = zeros(tama_loop,1);
for j = 1: tama_loop
    input = entrada(j,1);
    vetor_delay(1,1) = input;
    y = a0* vetor_delay(1,1) + a1 *  vetor_delay(n1,1) + a2 *  vetor_delay(n2,1);
    % Desloca o vetor de delay
    for k = 0: tama_delay-2
            vetor_delay(tama_delay-k,1) =  vetor_delay(tama_delay-k-1,1);
    end
  vet_saida(j,1) = y;
end


stem(vet_saida)

title('Teste Delay');
```

Implementação do Delay

TAREFA: Modifique o programa implementado para gerar delay em um arquivo de aúdio com atraso bem maior (D > 100ms) de forma que seja perceptivel ao ouvir (Ocenaudio).

Implementação do eco



$$y[n] = a0\ x[n] + a1\ y[n-D]$$

Implementação do Eco

- Utilize Fs= 8k e D um atraso de 1ms.
- Fs = 8k, Ts = 125us.
-
- Portanto:
- D corresponde a 8 amostras de delay
- Use a0 = 1,0; a1 = 0,5.

- Valide o algoritmo para uma entrada impulso unitário

Implementação do Eco

TAREFA: Modifique o programa implementado para gerar eco em um arquivo de aúdio com atrasos bem maiores de forma que sejam perceptiveis ao ouvir (Ocenaudio).

Disciplina: Processamento Digital de Sinais

Material aula 2
Ambiente Blackboard