DIGITAL SIGNAL PROCESSING

Processamento Digital de Sinais

Prof. Bruno Zatt
Ruhan Conceição

DIGITAL SIGNAL PROCESSING

Aula 6 - Soma de Convolução

Prof. Bruno Zatt
Ruhan Conceição

PREMISSA

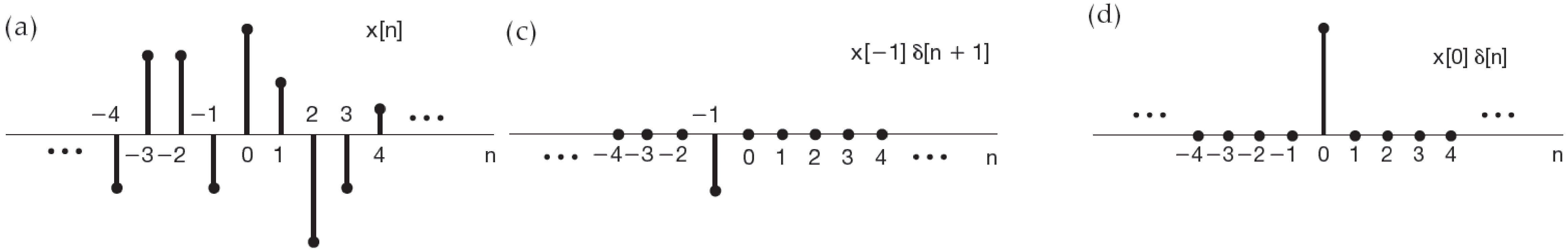
Um sinal discreto x[n] pode ser descrito como uma soma de impulsos deslocados e escalados:

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

PREMISSA

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

$x[-1]\delta[n+1]$ $x[-1]\delta[n-1]$ $x[0]\delta[n]$



PROPRIEDADE SELETIVA

A sequência $\delta[n-k]$ é diferente de zero somente quando

$$k = n$$

Assim, o somatória da equação abaixo "vasculha" a sequência de valores de $x[n]$ e extrai o valor somente onde $k = n$

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

SOMA DE CONVOLUÇÃO

Considerando um sinal de saída y[n] como a resposta de um sistema linear e invariante no tempo a uma dada entrada x[n], temos que:

$$y[n] = H\{x[n]\} = H\left\{\sum_{k=-\infty}^{+\infty} x[k]\delta[n-k]\right\} = \sum_{k=-\infty}^{+\infty} x[k]H\{\delta[n-k]\}$$

SOMA DE CONVOLUÇÃO

A resposta do sistema ao impulso $H\{\delta[n]\}$ pode ser interpretada como $h(n)$, logo:

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

SOMA DE CONVOLUÇÃO

A resposta do sistema ao impulso $H\{\delta[n]\}$ pode ser interpretada como $h(n)$, logo:

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

Soma de convolução

SOMA DE CONVOLUÇÃO

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

Soma de convolução

SOMA DE CONVOLUÇÃO

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

A resposta devida ao impulso $x[k]$ aplicada no instante k é $x[k]h[n-k]$, ou seja, é uma versão ponderada e deslocada (eco) de $h[n]$

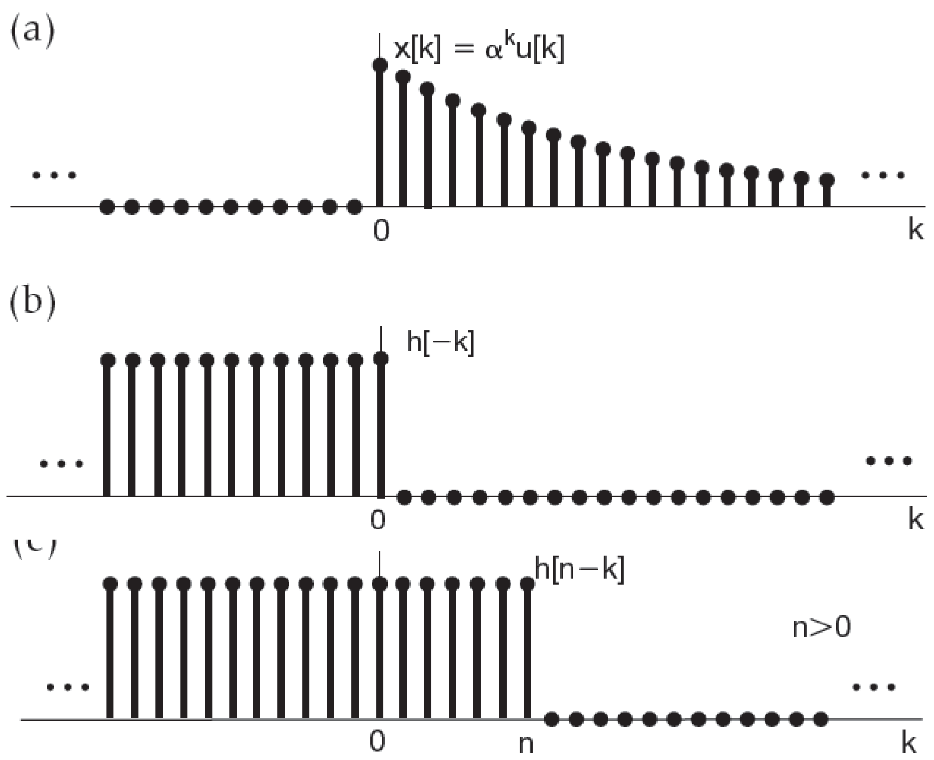
SOMA DE CONVOLUÇÃO

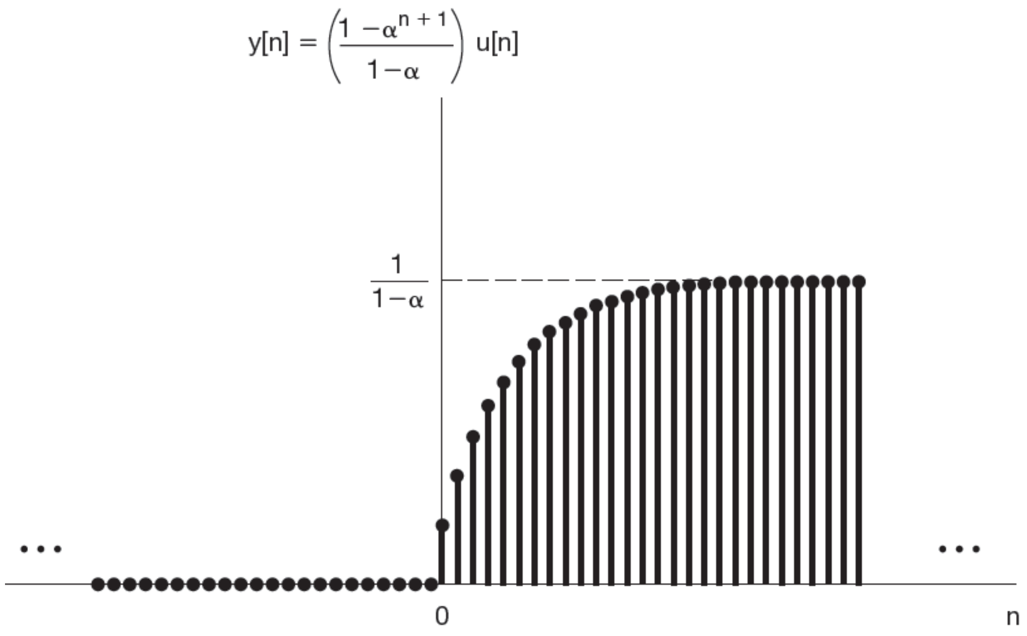
$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]h[n-k] = x[n] * h[n]$$

O resultado da soma de convolução é a superposição de todas essas respostas...

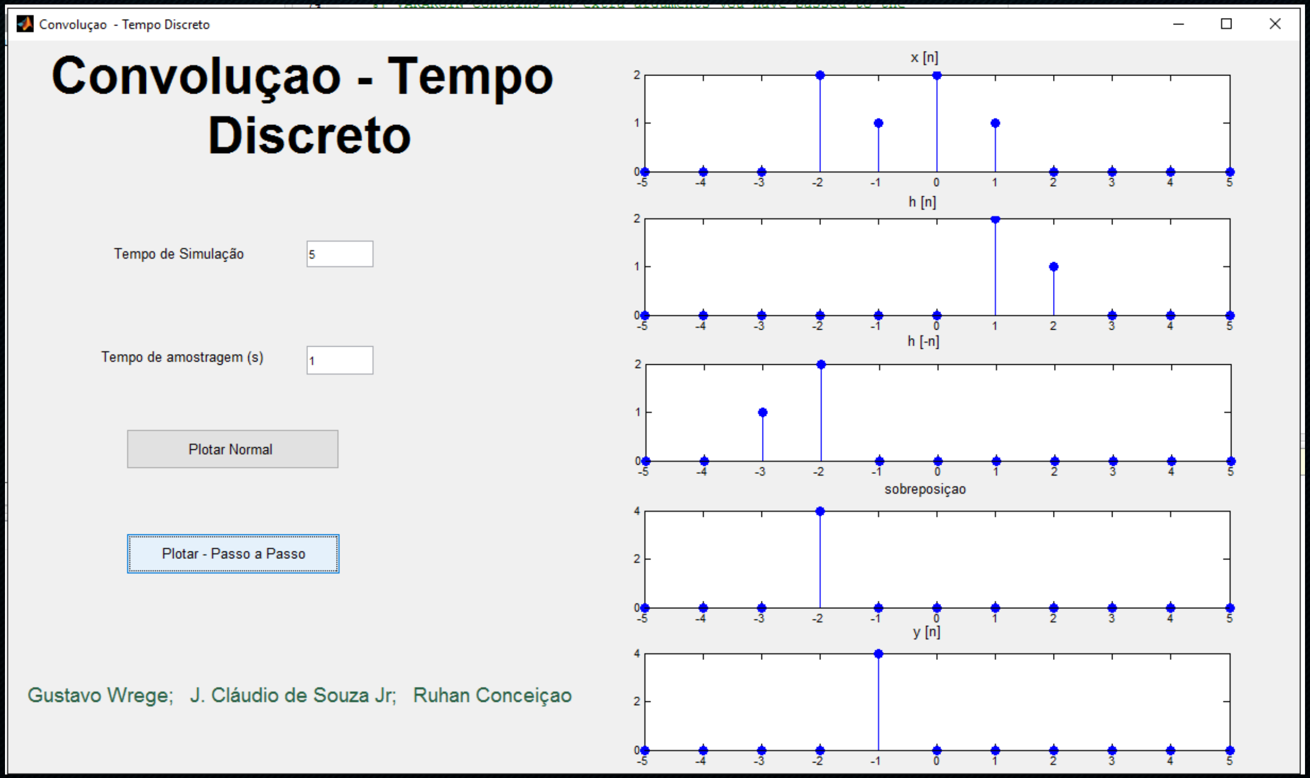
SOMA DE CONVOLUÇÃO

Supondo um sinal $x[n] = a^n u[n]$ como entrada de um sistema com resposta ao impulso $h[n] = u[n]$, o sinal resultante será $y[n] = \sum_{k=-\infty}^{+\infty} a^k u[k]u[n-k]$





SOMA DE CONVOLUÇÃO



Convolucao.m ...

SOMA DE CONVOLUÇÃO IN MATLAB

Por padrão, o Matlab já implementa a função de Soma de Convolução, tanto 1D quanto 2D

y = conv(x,h)
Y = conv2(X,H)

SOMA DE CONVOLUÇÃO IN MATLAB

Por padrão, o Matlab já implementa a função de Soma de Convolução, tanto 1D quanto 2D

Não abordaremos a teoria

```
y = conv(x,h)
Y = conv2(X,H)
```

SOMA DE CONVOLUÇÃO IN MATLAB

A função *conv* considera que ambas sequências começam em n=0, não tratando informações de deslocamento no tempo. Visando remediar este fato, a função *conv_m* gera um vetor para a variável independente do sinal convuluído baseado nas informações dos sinais de entrada.

```
function [y,ny] = conv_m(x,nx,h,nh)
    % Modified convolution routine for signal processing
    % ----------------------------------------------------
    % [y,ny] = conv_m(x,nx,h,nh)
    % [y,ny] = convolution result
    % [x,nx] = first signal
    % [h,nh] = second signal
    %
    nyb = nx(1)+nh(1); nye = nx(length(x)) + nh(length(h));
    ny = [nyb:nye]; y = conv(x,h);
```

EXERCÍCIO PROPOSTO Nº 1

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

1. Observe a resposta $y_{imp}[n]$ deste sistema ao impulso
2. Observe a resposta $y_1[n]$ deste sistema à um sinal $x_1[n]$ qualquer
3. Compare o sinal $y_1[n]$ com $x_1 * y_{imp}$

EXERCÍCIO PROPOSTO Nº 2

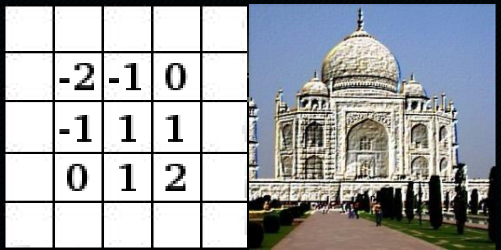
Matlab

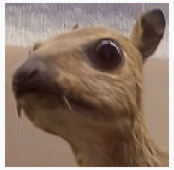
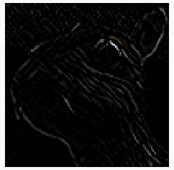
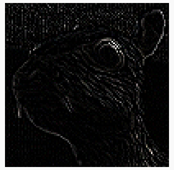
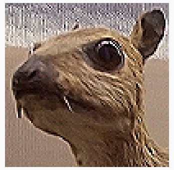
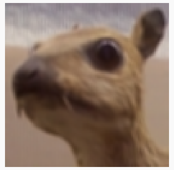
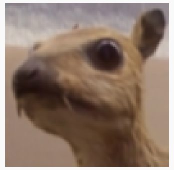
- *Considerando um sinal de áudio x que possua uma taxa de amostragem Fs; e um sinal $h[n] = \delta[n] + \frac{1}{2}\delta[n-a] + \frac{1}{4}\delta[n-2a]$, onde $a = \frac{Fs}{4}$, responda:*
  - *Qual será o resultado audível da convolução entre x[n] e h[n]?*
  - *Implemente em Matlab e confira a resposta*

EXERCÍCIO PROPOSTO Nº 3

Matlab (Convolução 2D)

- *Importe uma imagem em Matlab e convolua com diferentes matrizes Kernel:*



Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
5×5 Unsharp (with no image mask)	$\frac{-1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & -476 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	

HOMEWORK ENTREGAR

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

Implemente em Matlab a função que realiza a soma de convolução entre dois sinais x[n] e h[n] quaisquer, gerando o sinal y[n].

Não utilizar a função conv nem conv_m

```
[y, n] = convolucao(x,nx,h,nh)
```

DICAS

Descolorir imagem (Apenas Luminância)

```
y = imread('imagem.ext'); %importa imagem
y2 = rgb2ycbcr(y); %altera espaço de cores de RGB para YCbCr
y_cinza = y(:,:,1); %atribui apenas matriz de luminância da imagem
imshow(y_cinza); %mostra imagem em escala de cinza
```

Easter Egg 2

```
>> xpbombs
```

“That’s all Folks!”