DIGITAL SIGNAL PROCESSING

Processamento Digital de Sinais

Prof. Bruno Zatt
Ruhan Conceição

DIGITAL SIGNAL PROCESSING

Aula 4 - Sinais e Sistemas de Tempo Discreto (Part. 2)

Prof. Bruno Zatt
Ruhan Conceição

OPERAÇÕES ENTRE SINAIS

Transformação da Variável Dependente

MUDANÇA DE ESCALA DE AMPLITUDE

$$y[n] = c \times x[n]$$

```
>> y = c * x
%onde x é um vetor e c um escalar
```

ADIÇÃO

$$y[n] = x_1[n] + x_2[n]$$

ADIÇÃO IN MATLAB

$$y[n] = x_1[n] + x_2[n]$$

```
function [y,n] = sigadd(x1,n1,x2,n2)
    n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
    y1 = zeros(1,length(n)); y2 = y1; % initialization
    y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with duration of y
    y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with duration of y
    y = y1+y2;
```

MULTIP. AMOSTRA POR AMOSTRA

$$y[n] = x_1[n] . x_2[n]$$

MULTIP. AMOSTRA POR AMOSTRA IN MATLAB

$$y[n] = x_1[n].\,x_2[n]$$

```
function [y,n] = sigmult(x1,n1,x2,n2)
    n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
    y1 = zeros(1,length(n)); y2 = y1; % initialization
    y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with duration of y
    y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with duration of y
    y = y1.*y2;
```

MULTIP. AMOSTRA POR AMOSTRA IN MATLAB

$$y[n] = x_1[n].\, x_2[n]$$

```
function [y,n] = sigmult(x1,n1,x2,n2)
    n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
    y1 = zeros(1,length(n)); y2 = y1; % initialization
    y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with duration of y
    y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with duration of y
    y = y1.*y2;
```

y1 e y2 são matrizes unidimensionais, logo deve-se utilizar o operador .*
para evitar que o Matlab realize multiplicação matricial entre y1 e y2

DIFERENCIAÇÃO

$$y[n] = x[n] - x[n-1]$$

Integração

$$y[n] = \sum_{i=0}^{n} x[i]$$

OPERAÇÕES ENTRE SINAIS

Transformação da Variável Independente

DESLOCAMENTO NO TEMPO

$$y[n] = x[n - n_0]$$

$n_0 > 0$ deslocamento à direita
$n_0 < 0$ deslocamento à esquerda

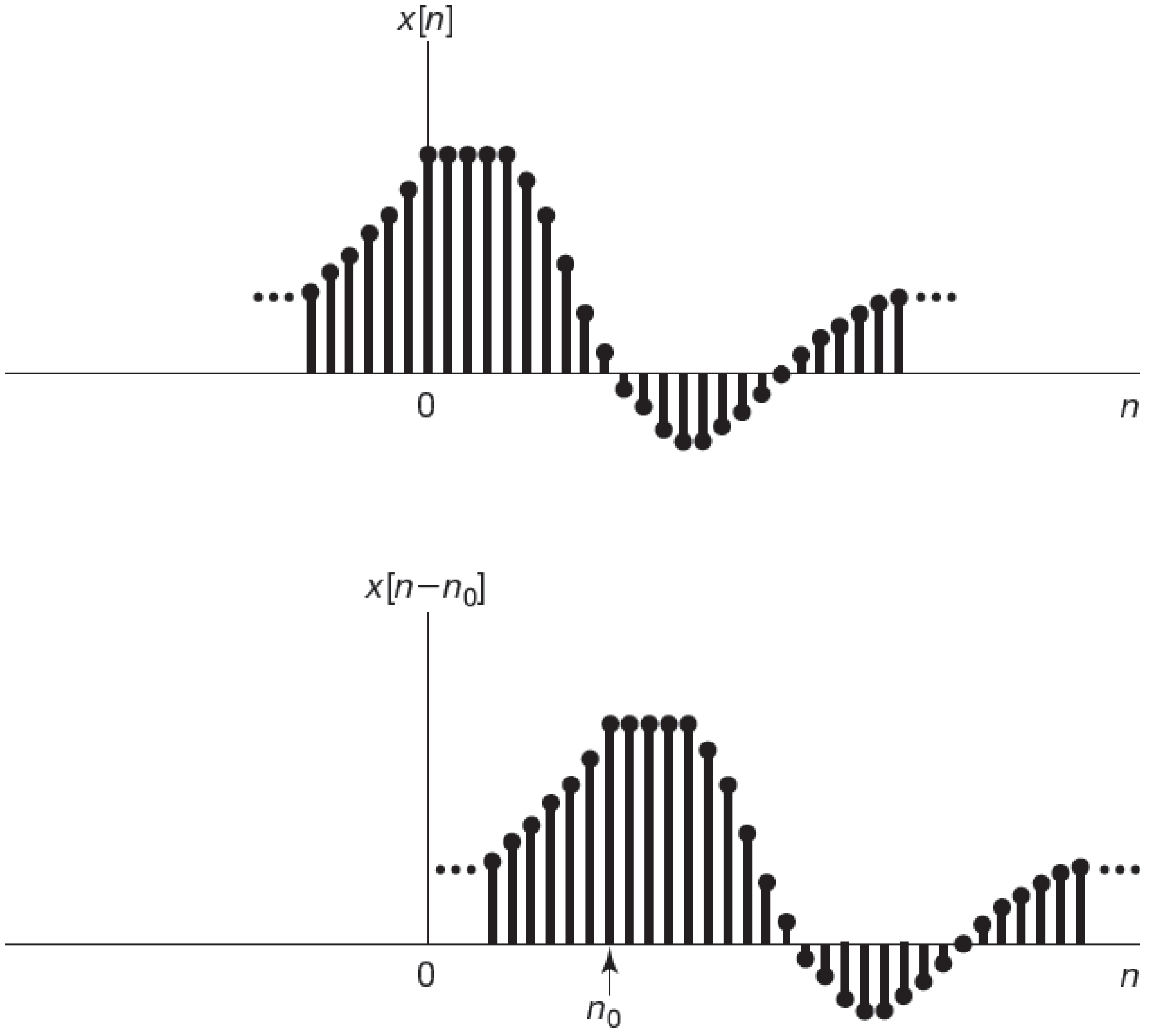
DESLOCAMENTO NO TEMPO IN MATLAB

$$y[n] = x[n - n_0]$$

```
function [y,n] = sigshift(x,m,k)
        n = m+k;
        y = x;
```

DESLOCAMENTO NO TEMPO

Sinal de tempo discreto relacionados por um deslocamento no tempo. Nesta figura, $n_0>0$, de modo que $x[n - n_0]$ é uma versão atrasada de $x[n]$ (isto é, cada ponto em $x[n]$ aparece atrasado em $x[n - n_0]$



REFLEXÃO NO TEMPO

$$y[n] = x[-n]$$

$n_0 > 0$ deslocamento à direita
$n_0 < 0$ deslocamento à esquerda

REFLEXÃO NO TEMPO

(a) Sinal de tempo discreto x[n]; (b) sua reflexão x[-n] em relação a n = 0

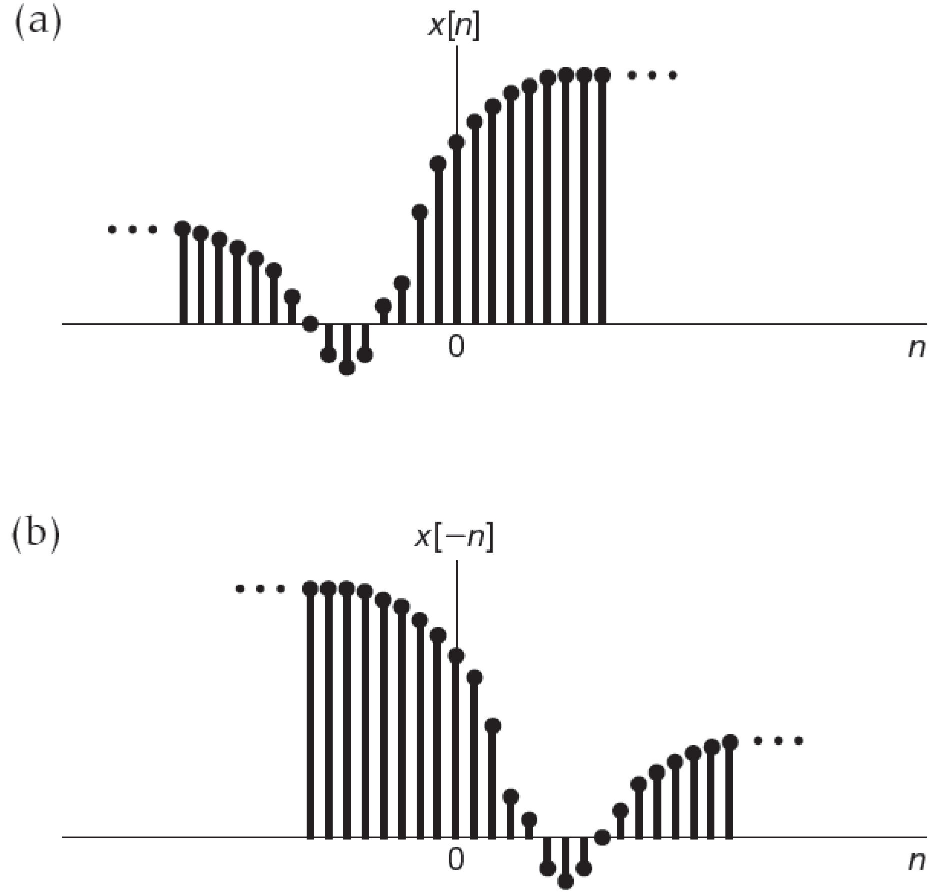


Figura 1.10 (a) Sinal de tempo discreto $x[n]$; (b) sua reflexão $x[-n]$ em relação a $n = 0$.

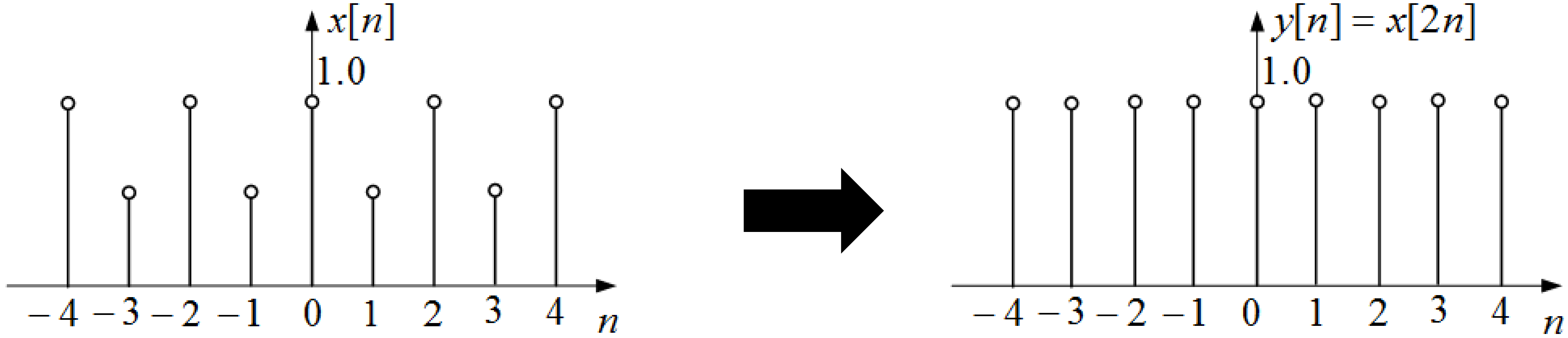
MUDANÇA DE ESCALA DE TEMPO

$$y[n] = x[kn]$$

$k > 1$, y[n] será a versão comprimida de x[n]
0 < k < 1, y[n] será a versão expandida de x[n]*

*amostras criadas são igualadas a zero

MUDANÇA DE ESCALA DE TEMPO



OPERAÇÕES ENTRE SINAIS

Potência e Energia de um Sinal

ENERGIA

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

POTÊNCIA

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

SINAIS PARES E ÍMPARES

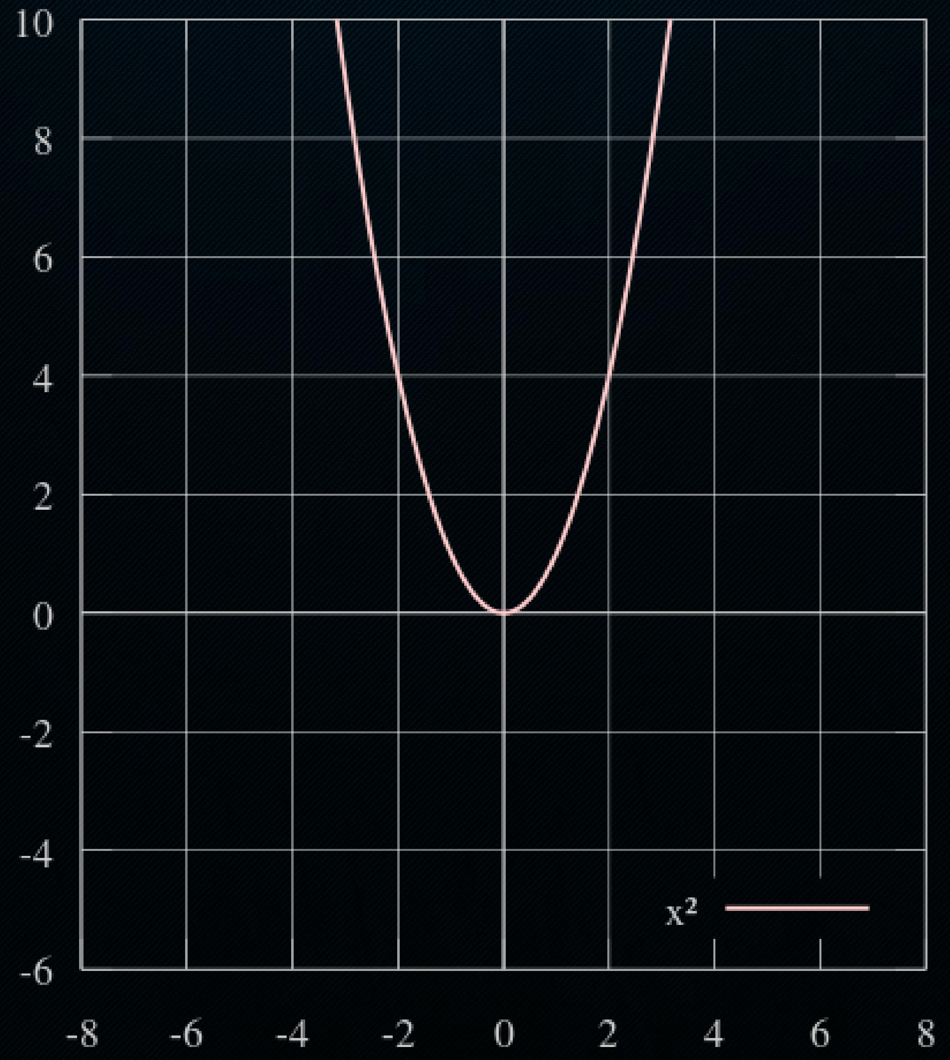
SINAL PAR

$$x[n] = x[-n]$$

Exemplos:

$$x[n] = \cos(\omega_0 n)$$
$$x[n] = n^2$$

SINAL PAR



SINAL ÍMPAR

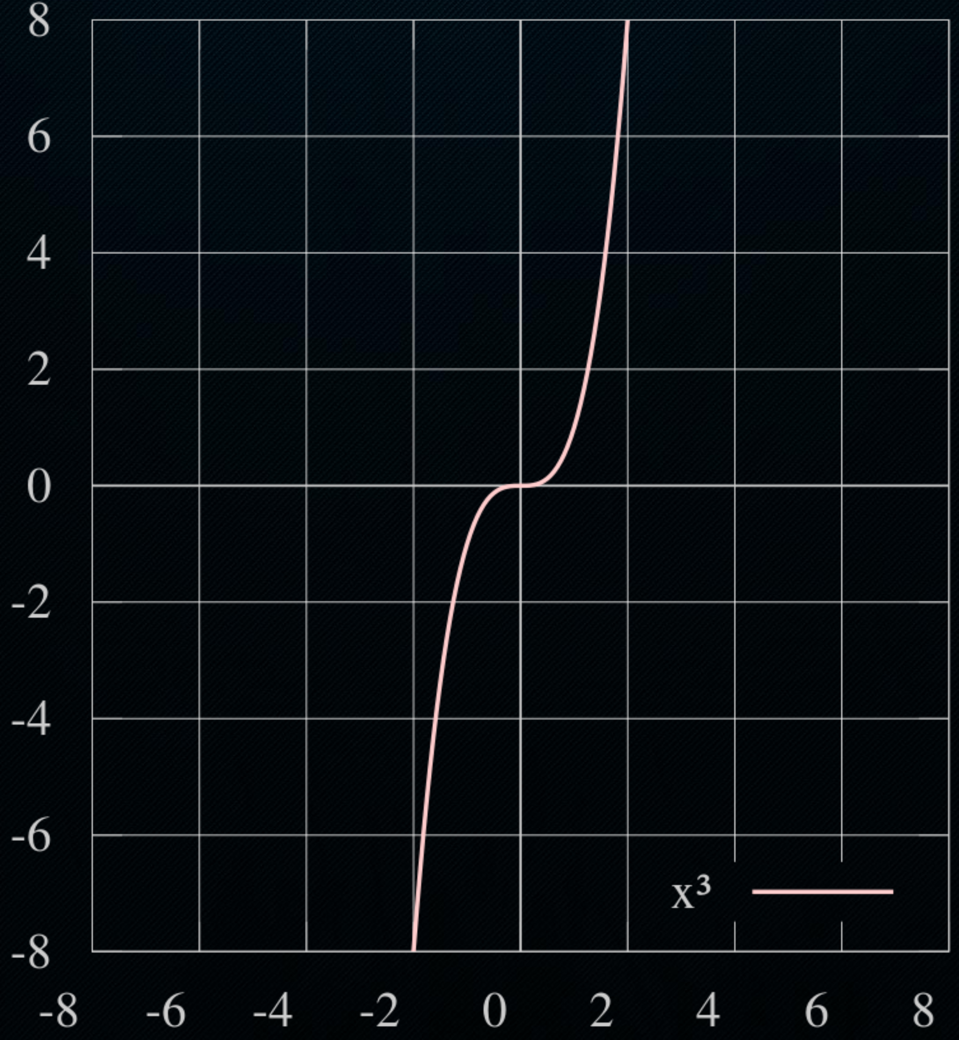
$$x[n] = -x[-n]$$

Exemplos:

$$x[n] = \sin(\omega_0 n)$$

$$x[n] = n^3$$

SINAL ÍMPAR



DECOMPOSIÇÃO DE UM SINAL

$$x[n] = x_e[n] + x_o[n]$$

Um sinal real qualquer $x[n]$ pode ser decomposto em uma componente par (x_e) e uma ímpar (x_o)

DECOMPOSIÇÃO DE UM SINAL

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

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

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
    % Real signal decomposition into even and odd parts
    % --------------------------------------------------
    % [xe, xo, m] = evenodd(x,n)
    %
    if any(imag(x) ~= 0)
        error('x is not a real sequence')
    end
    m = -fliplr(n);
    m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
    nm = n(1)-m(1); n1 = 1:length(n);
    x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
    xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
    % Real signal decomposition into even and odd parts
    % -------------------------------------------
    % [xe, xo, m] = evenodd(x,n)
    %
    if any(imag(x) ~= 0)
        error('x is not a real sequence')
    end
    m = -fliplr(n);
    m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
    nm = n(1)-m(1); n1 = 1:length(n);
    x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
    xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

Somente números reais

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
    % Real signal decomposition into even and odd parts
    % -------------------------------------------------
    % [xe, xo, m] = evenodd(x,n)
    %
    if any(imag(x) ~= 0)
        error('x is not a real sequence')
    end
    m = -fliplr(n);
    m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
    nm = n(1)-m(1); n1 = 1:length(n);
    x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
    xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

Se n = [1 2 3 4 5],

m = [-5 -4 -3 -2 -1]

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
    % Real signal decomposition into even and odd parts
    % ----------------------------------------------
    % [xe, xo, m] = evenodd(x,n)
    %
    if any(imag(x) ~= 0)
        error('x is not a real sequence')
    end
    m = -fliplr(n);
    m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
    nm = n(1)-m(1); n1 = 1:length(n);
    x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
    xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

Se n = [1, 2, 3, 4, 5]

m = [-5, -4, -3, -2, -1]

m1 = -5, m2 = 5

m = [-5, -4, ... 4, 5]

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
    % Real signal decomposition into even and odd parts
    % ---------------------------------------------
    % [xe, xo, m] = evenodd(x,n)
    %
    if any(imag(x) ~= 0)
        error('x is not a real sequence')
    end
    m = -fliplr(n);
    m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
    nm = n(1)-m(1); n1 = 1:length(n);
    x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
    xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

Se n = [1, 2, 3, 4, 5]

m = [-5, -4, -3, -2, -1]

m1 = -5, m2 = 5

m = [-5, -4, ... 4, 5]

nm = 5, n1 = [1 ... 5]

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
        % Real signal decomposition into even and odd parts
        % ---------------------------------------------
        % [xe, xo, m] = evenodd(x,n)
        %
        if any(imag(x) ~= 0)
                error('x is not a real sequence')
        end
        m = -fliplr(n);
        m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
        nm = n(1)-m(1); n1 = 1:length(n);
        x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
        xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

Se n = [1, 2, 3, 4, 5]

m = [-5, -4, -3, -2, -1]

m1 = -5, m2 = 5

m = [-5, -4, ... 4, 5]

nm = 6, n1 = [1 ... 5]

x1 = [0 0 0 ... 0] (11 zeros)

x1[6+n[1:5]] = x[n[1:5]]

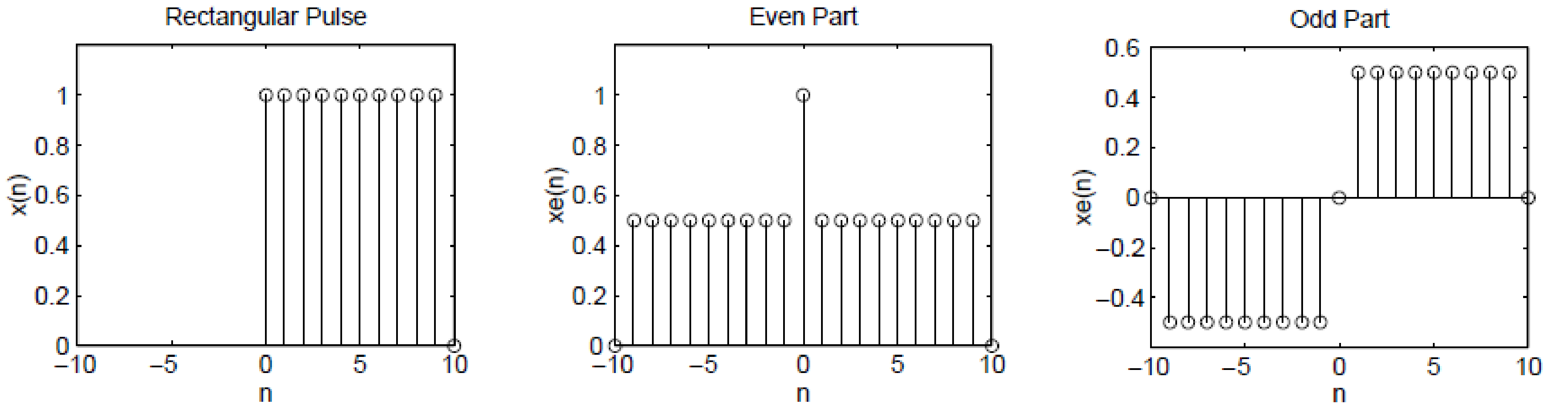
x = x1

DECOMPOSIÇÃO DE UM SINAL

```
function [xe, xo, m] = evenodd(x,n)
    % Real signal decomposition into even and odd parts
    % -----------------------------------------------
    % [xe, xo, m] = evenodd(x,n)
    %
    if any(imag(x) ~= 0)
        error('x is not a real sequence')
    end
    m = -fliplr(n);
    m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
    nm = n(1)-m(1); n1 = 1:length(n);
    x1 = zeros(1,length(m)); x1(n1+nm) = x; x = x1;
    xe = 0.5*(x + fliplr(x)); xo = 0.5*(x - fliplr(x));
```

Aplica fórmula da decomposição em sinais par e ímpar

DECOMPOSIÇÃO DE UM SINAL



EXERCÍCIOS PROPOSTOS

Matlab

- *Dada a função* $x[n] = 3\delta[n+2] - 2u[n-5] - u[2n]$, *faça:*
  - *a) implemente em Matlab a função x[n], usando subplot (1 para cada um dos termos de x[n], e um para o próprio x[n]). Use o comando stem*
  - *b) plote (stem) a derivada (diferença) deste sinal*
  - *C) plote (stem) a integral deste sinal*

“That’s all Folks!”