Experimento

Operações básicas em sinais com utilizando programa de computação simbólica.

Objetivo

Ilustrar com vários exemplos algumas operações básicas para realizar transformação no tempo de sinais, utilizando códigos de programa computacional que são executados sobre GNU OCTAVE (Linguagem de Programação Científica) com o emprego de programa de computação simbólica.

Nota: Normalmente o OCTAVE gera números, não fórmulas. Mas às vezes se deseja uma fórmula em vez de um número. Por exemplo, a derivada de uma função. Além disso, às vezes se deseja ver um número exato, e os números não inteiros no OCTAVE apresentam erros de arredondamento. As fórmulas fornecem a flexibilidade e rapidez para desenvolver equações de sinais e sistemas e obter os resultados melhores. Para tais propósitos, um programa de computação simbólica é necessário. Esse programa está disponível no OCTAVE como *pkg load symbolic*. A instalação de **Symbolic Package para GNU Octave** está em anexo.

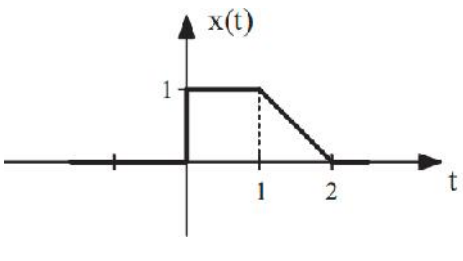
Material Utililizado

1. Linguagem de Programação Científica OCTAVE
2. Computador

Fundamentos

Considerar o seguinte exemplo ilustrativo.

Exemplo: Seja *x(t)* o sinal contínuo representado na figura abaixo. Obter: $x(\frac{1}{2}t+1)$



Soluçao Computacional via OCTAVE

Para calcular a transformação no tempo, deve-se realizar uma operação de escalonamento e deslocamento, ambas em relação ao eixo *x*. A ordem das operações não é importante, contanto que se tenha certeza da execução. Pode-se usar os seguintes comandos do OCTAVE para obter:

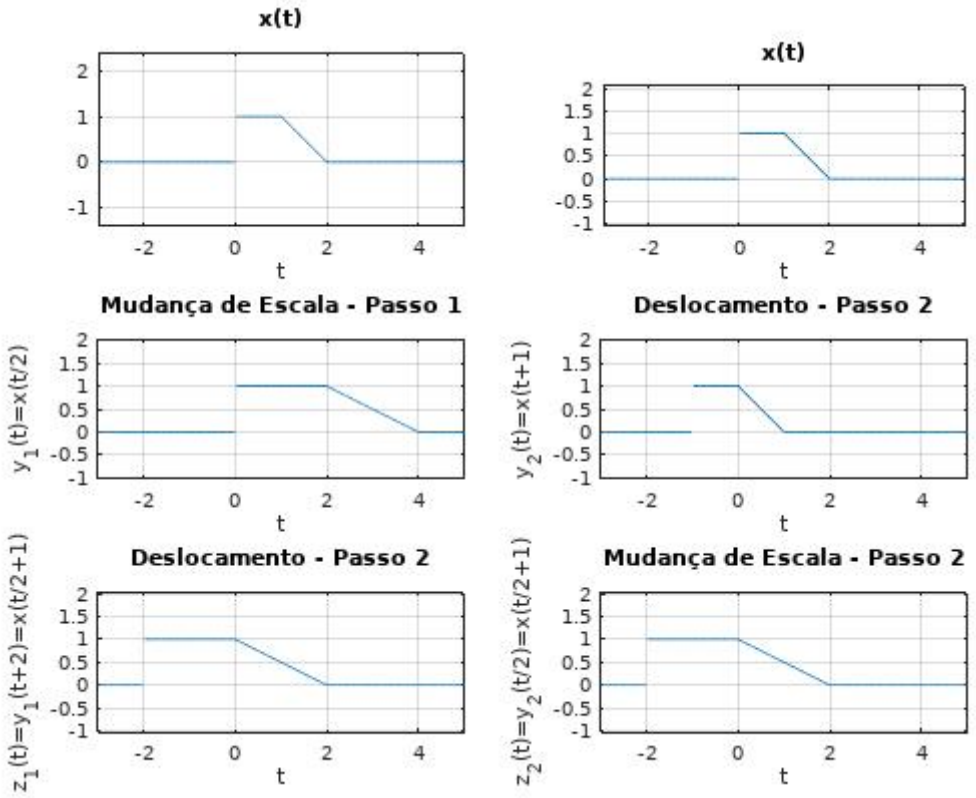
$$x(\frac{1}{2}t+1)$$

a partir de dois procedimentos distintos, a saber:

```
% Exemplo - Solução
close all;
home;
clear;
pkg load symbolic
syms t;

% Gráfico de x(t)
x = heaviside(t)-heaviside(t-1)+(-t+2)*(heaviside(t-1)-heaviside(t-2));
y1 = subs(x,t,t/2);
z1 = subs(y1,t,t+2);
figure(1);
subplot(321)
ezplot(x,[-3,5]);
title('x(t)');
axis equal
grid

pause
subplot(323)
ezplot(y1,[-3,5]);

title('Mudança de Escala - Passo 1');
ylabel('y_1(t)=x(t/2)');
axis equal
grid
pause
subplot(325)
ezplot(z1,[-3,5]);
ylabel('z_1(t)=y_1(t+2)=x(t/2+1)');
title('Deslocamento - Passo 2');
axis equal
grid
pause

y2 = subs(x,t,t+1);
z2 = subs(y2,t,t/2);
subplot(322)
ezplot(x,[-3,5]);
title('x(t)');
```

41. axis equal
42. grid
43. pause
44. subplot(324)
45. ezplot(y2,[-3,5]);
46. title('Deslocamento - Passo 2');
47. ylabel('y_2(t)=x(t+1)');
48. axis equal
49. grid
50.
51. pause
52. subplot(326)
53. ezplot(z2,[-3,5]);
54. ylabel('z_2(t)=y_2(t/2)=x(t/2+1)');
55. title('Mudança de Escala - Passo 2');
56. axis equal
57. grid

Os resultados da execução do programa acima são nos gráficos abaixo.



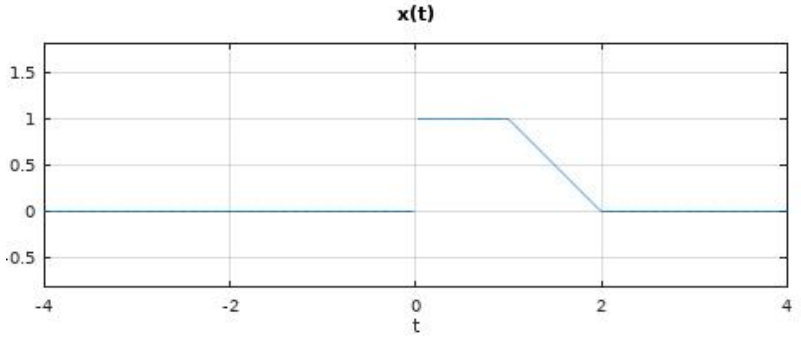
Nota: Lembrar de que um sinal como *x(at + b)*, quando primeiro faz-se a mudança de escala e depois o deslocamento (vide primeira coluna da figura acima), a quantidade da escala será *a*, mas a a quantidade do deslocamento será dada por *b/a*.

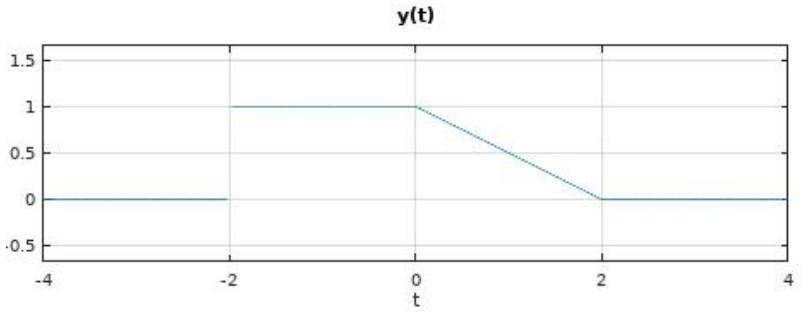
Pode-se automatizar a operação, escrevendo uma função que irá obter e representar qualquer de transformação no tempo de sinais. Seja a seguinte função para OCTAVE:

```
function OperaSinal(x,Ix,y,Iy)

% &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
% .
%   Código -   entrada: um sinal e sua transformação no Tempo
%              - Saída: Gráficos do sinal e da sua transformação
%
% ARGUMENTOS:
%  x  - Sinal Original  (Entrada).
% Ix  - Intervalo de Visualização  do Sinal Original.
% y   - Sinal Transformado.
% Iy  - Intervalo de Visualização  do Sinal.
%
% Observação:

% O sinal deve ser uma função simbólica, com uma variável independente única,
% que tem que ser t.
%
% &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

syms t;
F = figure(1);
set(F,'name','OPERAÇÕES BÁSICAS COM SINAIS CONTÍNUOS');
subplot(211)
ezplot(x,Ix);

title('x(t)');
axis equal
grid
subplot(212)
ezplot(y,Iy);

title('y(t)');
axis equal
grid
```

Com o auxílio da função anterior, para resolver o exemplo acima, precisa-se apenas digitar o as seguintes linhas de código:

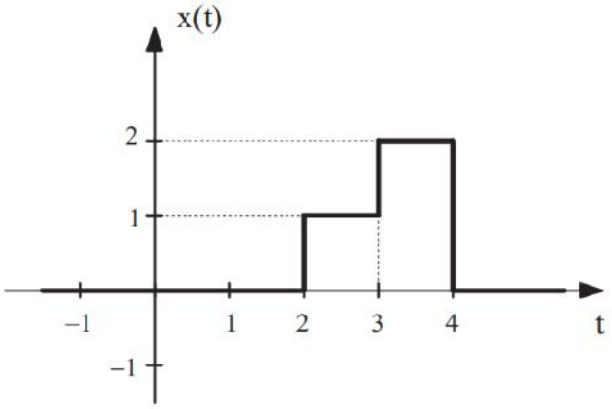
1. close all;
2. home;
3. clear;
4. pkg load symbolic
5. syms t;
6. x = heaviside(t)-heaviside(t-1)+(-t+2)*(heaviside(t-1)-heaviside(t-2));
7. y = subs(x,t,t/2+1);
8. OperaSinal(x,[-4 4],y,[-4 4])





Problema Proposto

Seja x*(t)* o sinal contínuo representado na figura abaixo:



Pede-se:

(a) Obter as seguintes transformações de sinal de *x(t)*:

$$1)\ x(-t+1) \quad ; \quad 2)\ x(2t-1) \quad ; \quad 3)\ x(-2t+4) \quad ; \quad 4)\ x(-t-2)$$

$$5)\ x\left(\frac{t+6}{3}\right) \quad ; \quad 6)\ -\frac{1}{2}x\left(-\frac{1}{3}t+2\right)-1$$

(b) Resolver o problema proposto com a ajuda da função *OperaSinal* e verificar os mesmos resultados obtidos no item (a).

Anexo

Symbolic Package para GNU Octave

Uma implementação de uma *toolbox* para computação simbólica usando *SymPy*.

Como instalar

1. Baixar o arquivo *symbolic-win-py-bundle-2.9.0.tar.gz* de: (https://github.com/cbm755/octsympy/releases)

1. Iniciar o Octave.

2. No prompt do Octave, digitar *pkg install symbolic-win-py-bundle-2.9.0.tar.gz*

3. No prompt do Octave, digitar *pkg load symbolic.*

4. No prompt do Octave, digitar *syms x*, então f = *(sin (x / 2)) ^ 3*, *diff (f, x)*, etc. Verificar os resultados.

Nota: O pacote *symbolic-win-py-bundle* não deve ter dependências além de Octave (inclui SymPy e um interpretador Python). Alternativamente, você mesmo pode instalar Python e SymPy e usar o comando simbólico padrão *pkg install -forge*.