Aplicações de sinais

Prof. Raul T. Rato

DEEC - 2021

Considerações sobre Janelas … 1

Antes de mais:

Apresentação para a próxima aula (slides):

O ficheiro Aula25Mar contém uma risca espectral a que frequência?

Variáveis lá presentes: kRr – Sinal
qTs – Período de amostragem

Apresente não só os resultados como também a listagem do código

Quiz (Questionário) para relembrar técnicas de Octave/Matlab

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    bK= 0:4;
6 -    kS= cos(bK');
7
```

A variável kS contém o sinal:

a) sin(n), n ϵ [1..5]

b) cos(n), n ϵ [1..5]

c) sin(n-1), n ϵ [1..5]

d) cos(n-1), n ϵ [1..5]

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    bK= 0:4;
6 -    kS= cos(bK');
7
```

Tem-se que:

a) A variável kS é um vector linha e a bK é um vector coluna

b) A variável kS é um vector coluna e a bK é um vector linha

c) As variável kS e bK são ambas vectores coluna

d) As variável kS e bK são ambas vectores linha

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    plot(kS);
```

Tem-se que:

a) A variável `btt` começa em zero e acaba em `6.3`

b) A variável `btt` começa em zero e acaba em `6.283185307179586` (2π)

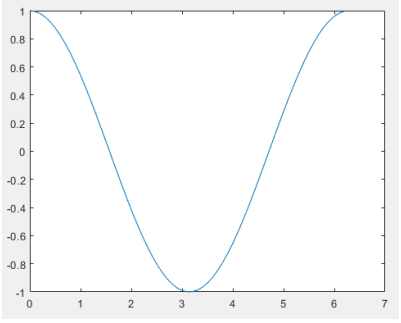
c) A variável `btt` começa em zero e acaba em `6.2`

d) A variável `btt` só pode ser `0` ou `0.1` conforme o sinal algébrico de `2*pi`

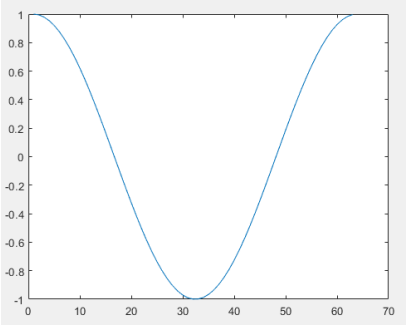
e) Nenhuma das anteriores

```
1 -     clear
2 -     close all
3 -     clc
4
5 -     btt= 0:.1:(2*pi);
6 -     kS= cos(btt');
7 -     plot(kS);
```

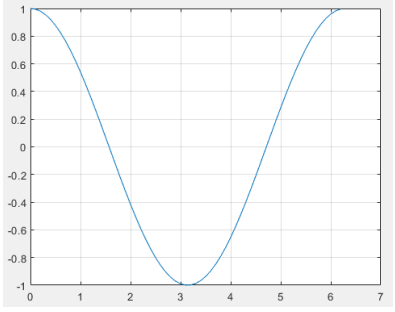
A



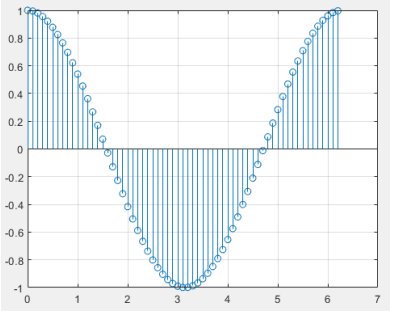
B



C



D



Tem-se que:

a) O código gera a figura A

b) O código gera a figura B

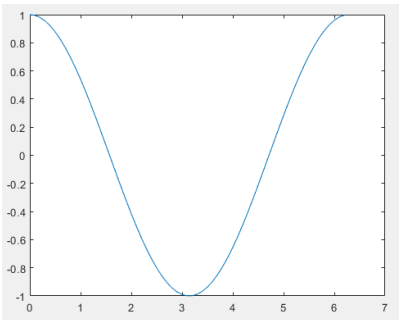
c) O código gera a figura C

d) O código gera a figura D

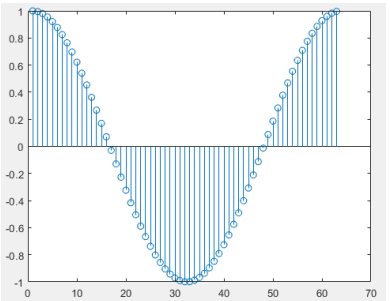
e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    stem(btt, kS);
8 -    grid on
```

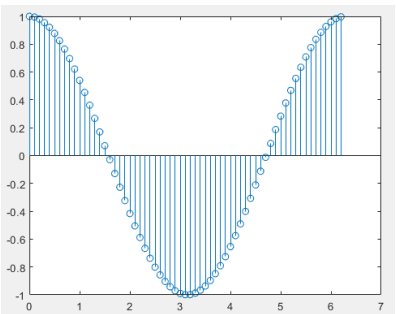
A



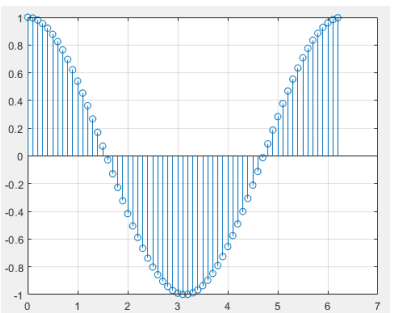
B



C



D



Tem-se que:

a) O código gera a figura A

b) O código gera a figura B

c) O código gera a figura C

d) O código gera a figura D

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    NoiseA= randn(20000,1);
6 -    NoiseB=  rand(1,20000);
```

Tem-se que:

a) `NoiseA` é um vector coluna de valores aleatórios com distribuição normal(0, 1)

b) `NoiseA` é um vector linha de valores aleatórios com distribuição normal(0, 1)

c) `NoiseA` é um vector coluna de valores aleatórios com distribuição uniforme entre(0, 1)

d) `NoiseA` é um vector linha de valores aleatórios com distribuição uniforme entre(0, 1)

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    NoiseA= randn(20000,1);
6 -    NoiseB=  rand(1,20000);
```

Tem-se que:

a) `NoiseA` e `NoiseB` só têm valores positivos

b) `NoiseA` só tem valores positivos

c) `NoiseB` só tem valores positivos

d) Tanto `NoiseA` como `NoiseB` podem ter valores negativos e positivos

e) Nenhuma das anteriores

Obteve-se

```
ans =

     2
     1
```

como resultado.

Então o comando foi:

a) `roots(2 -3 1)`

b) `roots([2 -3 1])`

c) `roots([1 -3 2])'`

d) `roots([1 -3 2])`

e) Nenhuma das anteriores

Pretende-se calcular as raízes de $2z^5+3z^3-2z^2+6z+1$

Então o código deverá ser:

a)
```
ZZ=[2x^5 + 3x^3 -2x^2 +6x +1];
roots(ZZ)
```

b)
```
ZZ= [1 6 -2 3 0 2];
roots(ZZ)
```

c)
```
ZZ=[2 0 3 -2 6 1];
roots(ZZ)
```

d)
```
ZZ=[2 3 -2 6 1];
roots(ZZ)
```

e) Nenhuma das anteriores

Um polinómio tem cinco raízes: 5, -5, 1i, -1i, 7
Pretende-se saber quais os seus coeficientes.

Então o código deverá ser:

a)
```
conv([1 5],  [1 -5],  [1  1i],  [1    -1i],   [1 7]);
```

b)
```
roots(conv(conv(conv([1 -5],  [1 5]), conv(  [1  -1i],  [1    1i])),   [1 -7]))
```

c)
```
ZZ=[5,   -5,    1i,      -1i,    7];
conv(ZZ)
```

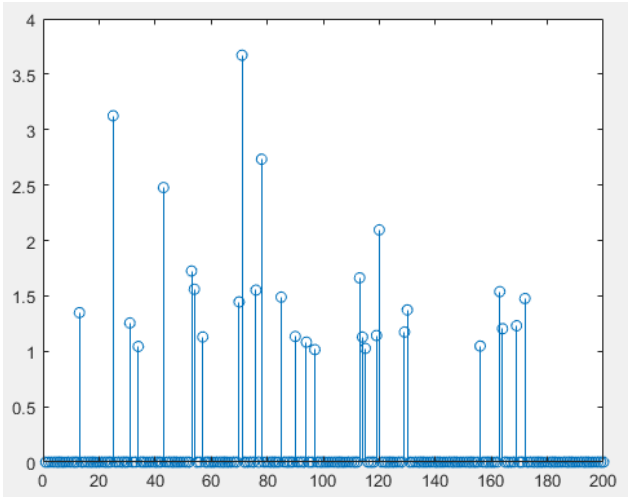
d)
```
ZZ=[5,   -5,    1i,      -1i,    7];
roots(ZZ)
```

e) Nenhuma das anteriores

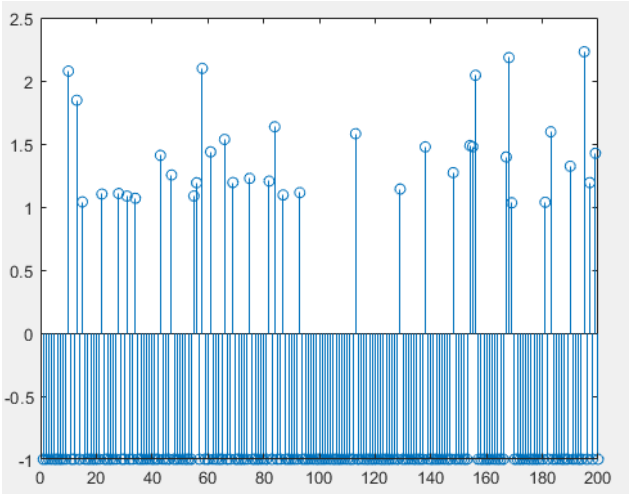
Este código:

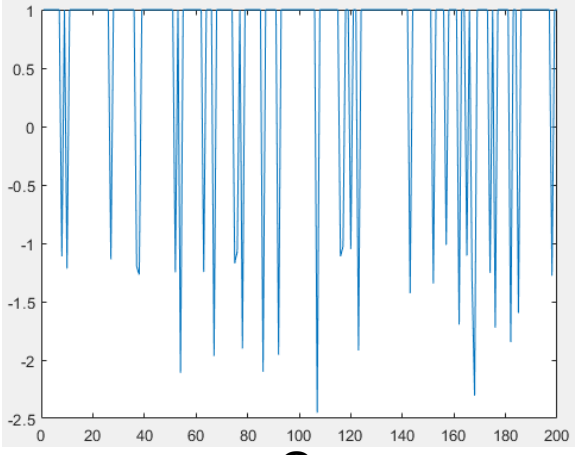
```
clear
close all
clc

Noise= randn(200,1);
Noise(Noise<1)=0;
stem(Noise)
```

A

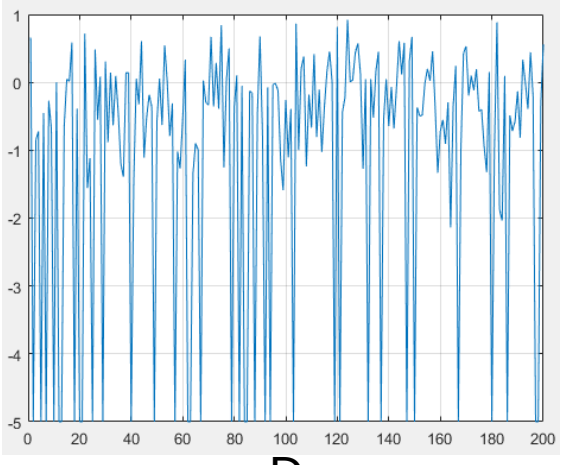


B





C



D

Tem que resultado?

a) A
b) B
c) C
d) D
e) Nenhuma das anteriores

Agora o Quiz com respostas:

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    bK= 0:4;
6 -    kS= cos(bK');
7
```

A variável kS contém o sinal:

a) sin(n), n ϵ [1..5]

b) cos(n), n ϵ [1..5]

c) sin(n-1), n ϵ [1..5]

d) cos(n-1), n ϵ [1..5]

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    bK= 0:4;
6 -    kS= cos(bK');
7
```

A variável kS contém o sinal:

a) sin(n), n ϵ [1..5]

b) cos(n), n ϵ [1..5]

c) sin(n-1), n ϵ [1..5]

d) cos(n-1), n ϵ [1..5]

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    bK= 0:4;
6 -    kS= cos(bK');
7
```

Tem-se que:

a) A variável kS é um vector linha e a bK é um vector coluna

b) A variável kS é um vector coluna e a bK é um vector linha

c) As variável kS e bK são ambas vectores coluna

d) As variável kS e bK são ambas vectores linha

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    bK= 0:4;
6 -    kS= cos(bK');
7
```

Tem-se que:

a) A variável kS é um vector linha e a bK é um vector coluna

b) A variável kS é um vector coluna e a bK é um vector linha

c) As variável kS e bK são ambas vectores coluna

d) As variável kS e bK são ambas vectores linha

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    plot(kS);
```

Tem-se que:

a) A variável `btt` começa em zero e acaba em `6.3`

b) A variável `btt` começa em zero e acaba em `6.283185307179586` (2π)

c) A variável `btt` começa em zero e acaba em `6.2`

d) A variável `btt` só pode ser `0` ou `0.1` conforme o sinal algébrico de `2*pi`

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    plot(kS);
```

Tem-se que:

a) A variável `btt` começa em zero e acaba em `6.3`

b) A variável `btt` começa em zero e acaba em `6.283185307179586` (2π)

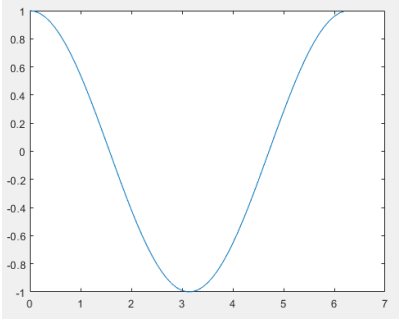
c) A variável `btt` começa em zero e acaba em `6.2`

d) A variável `btt` só pode ser `0` ou `0.1` conforme o sinal algébrico de `2*pi`

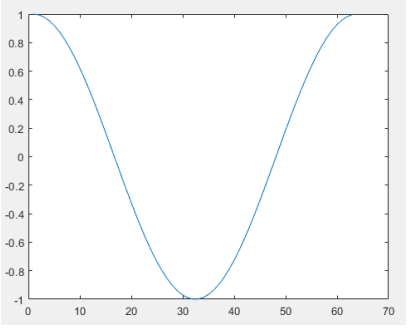
e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    plot(kS);
```

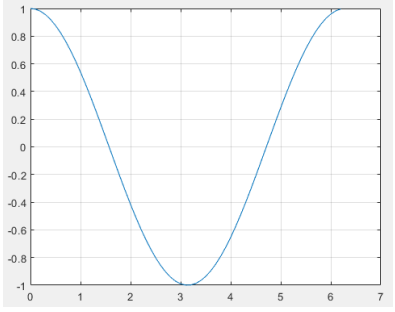
A



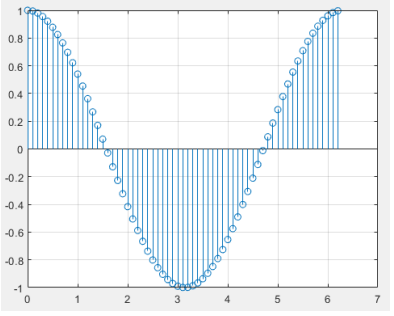
B



C



D



Tem-se que:

a) O código gera a figura A

b) O código gera a figura B

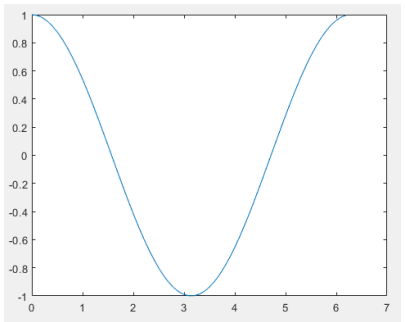
c) O código gera a figura C

d) O código gera a figura D

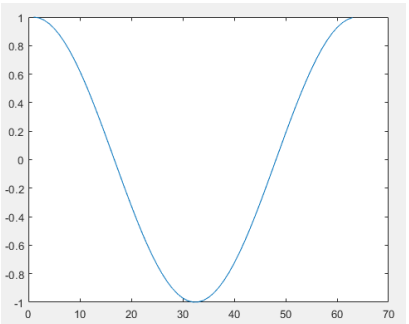
e) Nenhuma das anteriores

```
1 -     clear
2 -     close all
3 -     clc
4
5 -     btt= 0:.1:(2*pi);
6 -     kS= cos(btt');
7 -     plot(kS);
```

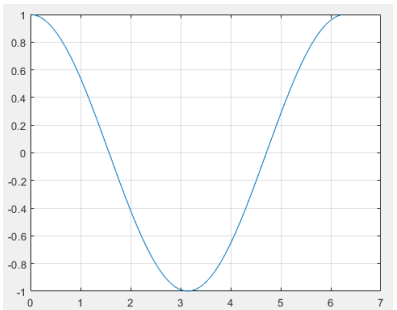
A



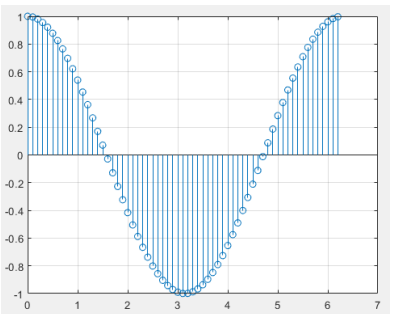
B



C



D



Tem-se que:

a) O código gera a figura A

b) O código gera a figura B

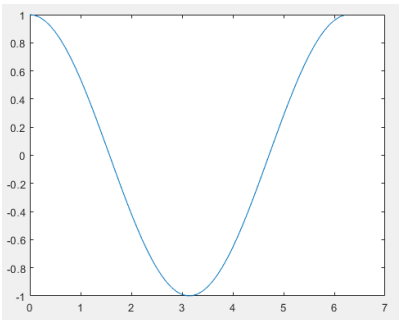
c) O código gera a figura C

d) O código gera a figura D

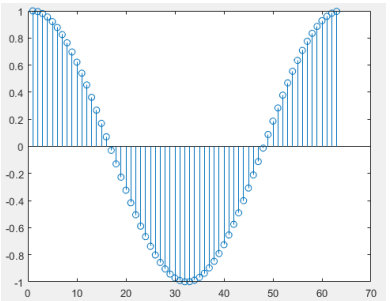
e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    stem(btt, kS);
8 -    grid on
```

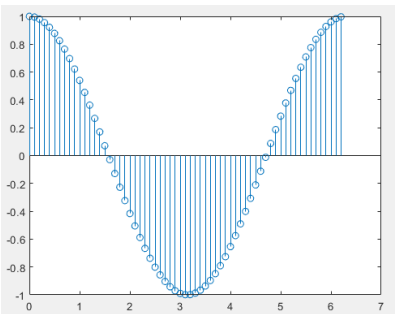
A



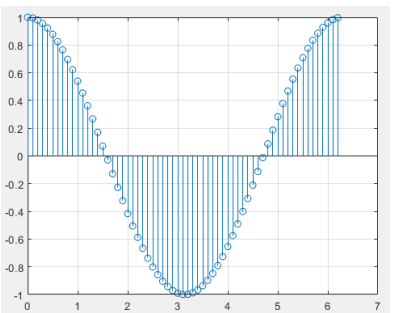
B



C



D



Tem-se que:

a) O código gera a figura A

b) O código gera a figura B

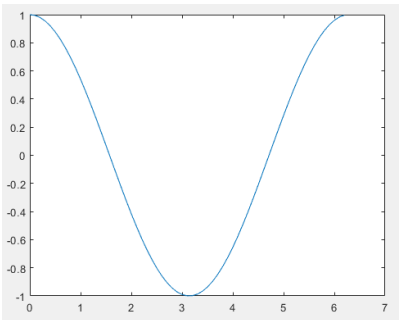
c) O código gera a figura C

d) O código gera a figura D

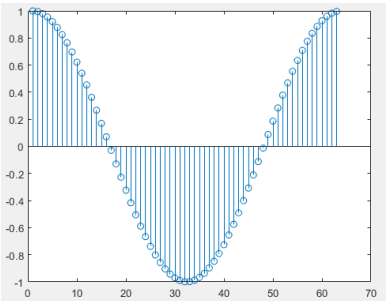
e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    btt= 0:.1:(2*pi);
6 -    kS= cos(btt');
7 -    stem(btt, kS);
8 -    grid on
```

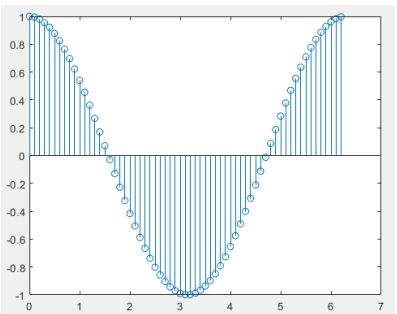
A



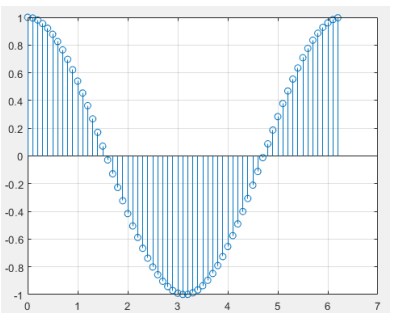
B



C



D



Tem-se que:

a) O código gera a figura A

b) O código gera a figura B

c) O código gera a figura C

d) O código gera a figura D

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    NoiseA= randn(20000,1);
6 -    NoiseB=  rand(1,20000);
```

Tem-se que:

a) `NoiseA` é um vector coluna de valores aleatórios com distribuição normal(0, 1)

b) `NoiseA` é um vector linha de valores aleatórios com distribuição normal(0, 1)

c) `NoiseA` é um vector coluna de valores aleatórios com distribuição uniforme entre(0, 1)

d) `NoiseA` é um vector linha de valores aleatórios com distribuição uniforme entre(0, 1)

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    NoiseA= randn(20000,1);
6 -    NoiseB=  rand(1,20000);
```

Tem-se que:

a) `NoiseA` é um vector coluna de valores aleatórios com distribuição normal(0, 1)

b) `NoiseA` é um vector linha de valores aleatórios com distribuição normal(0, 1)

c) `NoiseA` é um vector coluna de valores aleatórios com distribuição uniforme entre(0, 1)

d) `NoiseA` é um vector linha de valores aleatórios com distribuição uniforme entre(0, 1)

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    NoiseA= randn(20000,1);
6 -    NoiseB=  rand(1,20000);
```

Tem-se que:

a) `NoiseA` e `NoiseB` só têm valores positivos

b) `NoiseA` só tem valores positivos

c) `NoiseB` só tem valores positivos

d) Tanto `NoiseA` como `NoiseB` podem ter valores negativos e positivos

e) Nenhuma das anteriores

```
1 -    clear
2 -    close all
3 -    clc
4
5 -    NoiseA= randn(20000,1);
6 -    NoiseB=  rand(1,20000);
```

Tem-se que:

a) `NoiseA` e `NoiseB` só têm valores positivos

b) `NoiseA` só tem valores positivos

c) `NoiseB` só tem valores positivos

d) Tanto `NoiseA` como `NoiseB` podem ter valores negativos e positivos

e) Nenhuma das anteriores

Obteve-se

```
ans =

     2
     1
```

como resultado.

Então o comando foi:

a) `roots(2 -3 1)`

b) `roots([2 -3 1])`

c) `roots([1 -3 2])'`

d) `roots([1 -3 2])`

e) Nenhuma das anteriores

Obteve-se

```
ans =

     2
     1
```

como resultado.

Então o comando foi:

a) `roots(2 -3 1)`

b) `roots([2 -3 1])`

c) `roots([1 -3 2])'`

d) `roots([1 -3 2])`

e) Nenhuma das anteriores

Pretende-se calcular as raízes de $2z^5+3z^3-2z^2+6z+1$

Então o código deverá ser:

a)
```
ZZ=[2x^5 + 3x^3 -2x^2 +6x +1];
roots(ZZ)
```

b)
```
ZZ= [1 6 -2 3 0 2];
roots(ZZ)
```

c)
```
ZZ=[2 0 3 -2 6 1];
roots(ZZ)
```

d)
```
ZZ=[2 3 -2 6 1];
roots(ZZ)
```

e) Nenhuma das anteriores

Pretende-se calcular as raízes de $2z^5+3z^3-2z^2+6z+1$

Então o código deverá ser:

a)
```
ZZ=[2x^5 + 3x^3 -2x^2 +6x +1];
roots(ZZ)
```

b)
```
ZZ= [1 6 -2 3 0 2];
roots(ZZ)
```

c)
```
ZZ=[2 0 3 -2 6 1];
roots(ZZ)
```

d)
```
ZZ=[2 3 -2 6 1];
roots(ZZ)
```

e) Nenhuma das anteriores

Um polinómio tem cinco raízes: 5, -5, 1i, -1i, 7
Pretende-se saber quais os seus coeficientes.

Então o código deverá ser:

a)
```
conv([1 5],  [1 -5],  [1  1i],  [1    -1i],   [1 7]);
```

b)
```
roots(conv(conv(conv([1 -5],  [1 5]), conv(  [1  -1i],  [1    1i])),   [1 -7]))
```

c)
```
ZZ=[5,   -5,    1i,      -1i,    7];
conv(ZZ)
```

d)
```
ZZ=[5,   -5,    1i,      -1i,    7];
roots(ZZ)
```

e) Nenhuma das anteriores

Um polinómio tem cinco raízes: 5, -5, 1i, -1i, 7
Pretende-se saber quais os seus coeficientes.

Então o código deverá ser:

a)
```
conv([1 5],  [1 -5],  [1  1i],  [1    -1i],   [1 7]);
```

b)
```
roots(conv(conv(conv([1 -5],  [1 5]), conv(  [1  -1i],  [1    1i])),   [1 -7]))
```

c)
```
ZZ=[5,   -5,    1i,      -1i,    7];
conv(ZZ)
```

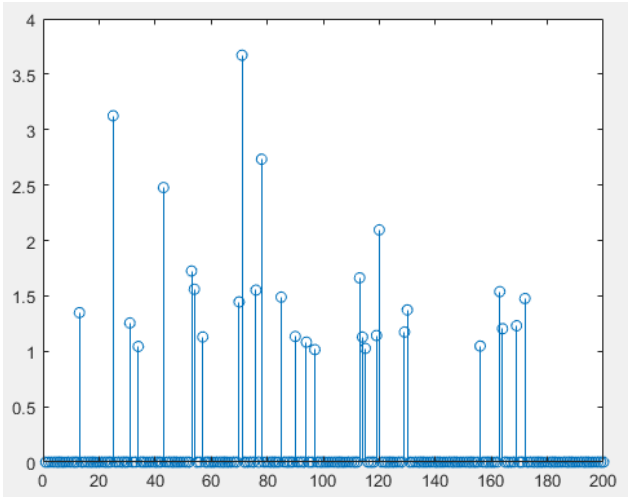
d)
```
ZZ=[5,   -5,    1i,      -1i,    7];
roots(ZZ)
```

e) Nenhuma das anteriores

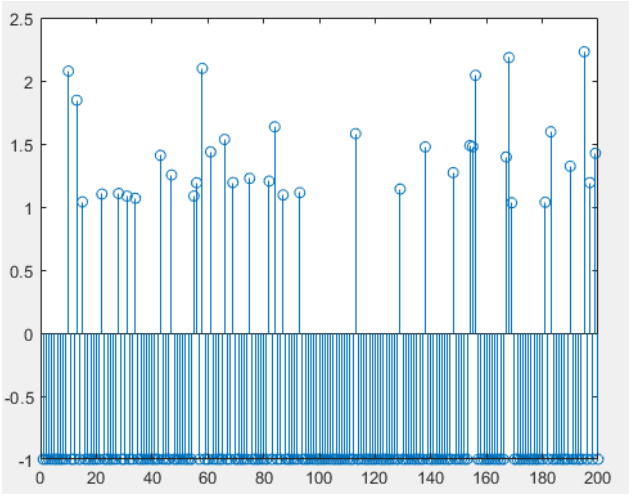
Este código:

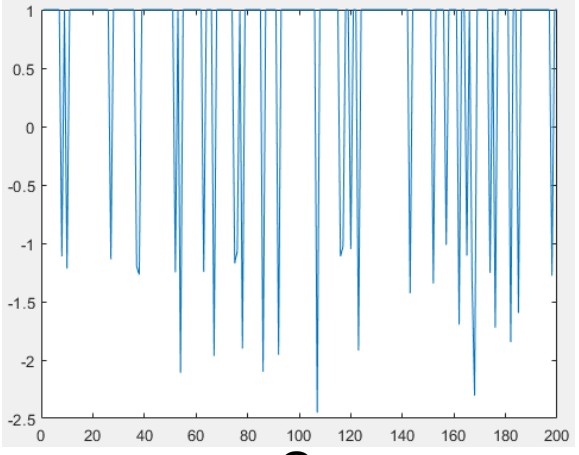
```
clear
close all
clc

Noise= randn(200,1);
Noise(Noise<1)=0;
stem(Noise)
```

A

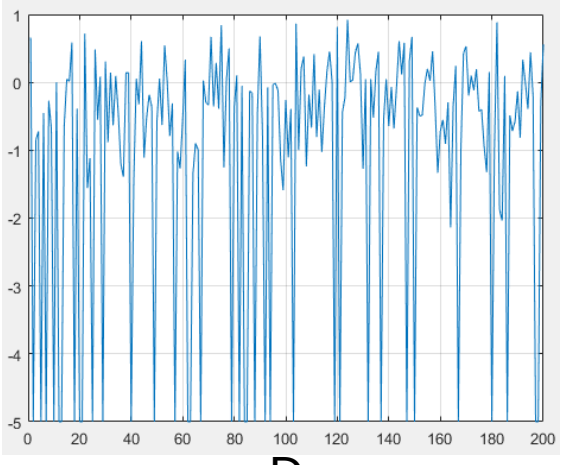


B





C



D

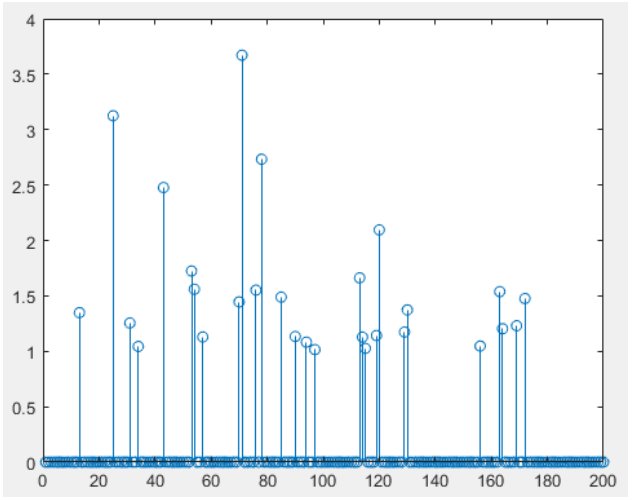
Tem que resultado?

a) A
b) B
c) C
d) D
e) Nenhuma das anteriores

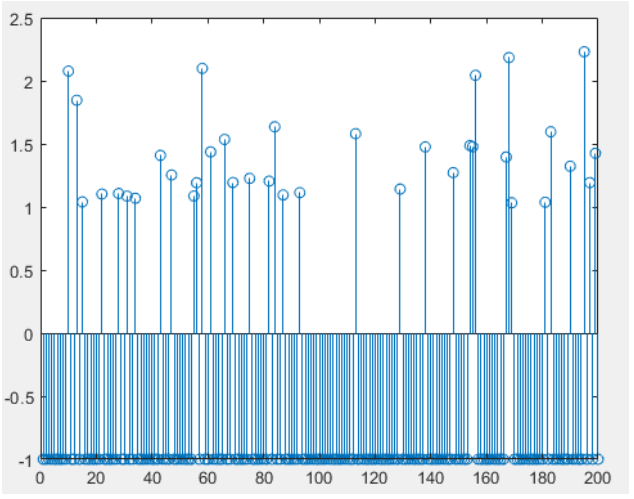
Este código:

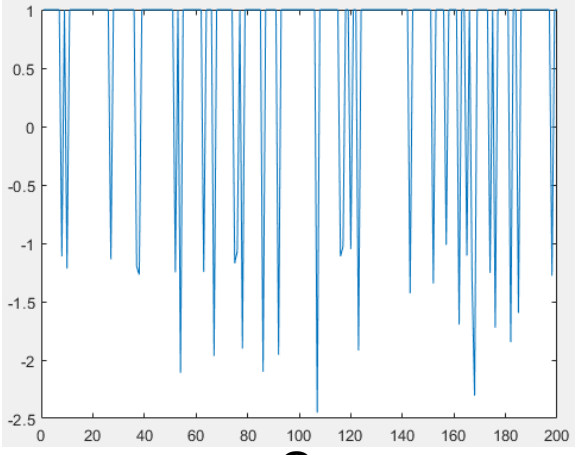
```
1 -     clear
2 -     close all
3 -     clc
4
5 -     Noise= randn(200,1);
6 -     Noise(Noise<1)=0;
7 -     stem(Noise)
```

A

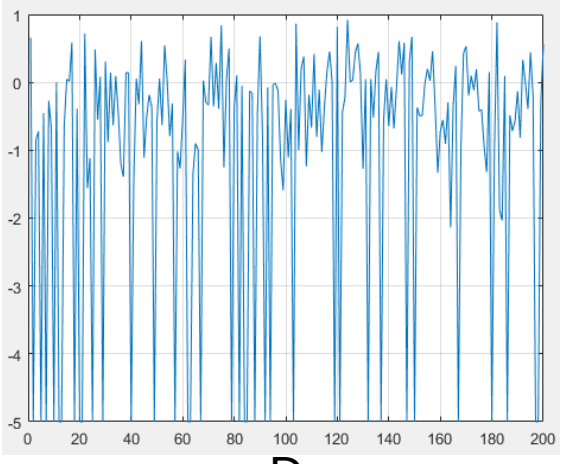


B





C



D

Tem que resultado?

a) A
b) B
c) C
d) D
e) Nenhuma das anteriores

Considerações sobre Janelas … 1

Vamos agora analisar este código:

```
clear
close all
clc


%% Time base setup

qTs= 0.001;              % 1 ms
qTT= 9.65;               % Sample duration - seconds

btt= 0:qTs:qTT;          %Time base
qNN= numel(btt);         % Sample count

%% Signal setup  (Sent signal)
qSigF= 50;               % Signal frequency
qSigT= 1/qSigF;

qPhase= pi*65/180;       % Phase is 65

kSig= sin(2*pi*qSigF*btt');

%   plot(btt, kSig)    %Visual check

%% Noise setup
qN0= 0.4;                %Noise power

kNoise= qN0*randn(qNN, 1);

%% Noisy signal (Received signal)

kRr= kNoise + kSig;

%   plot(btt, kRr)    %Visual check

%% Spectral analysis

kRrAf= abs(fft(kRr));

bffRaw= 0:(qNN-1);
kff= (1/qNN)*(1/qTs)*bffRaw';

plot(kff, 10*log10(kRrAf));
grid on;

%% Spectral analysis II

close all

stem(kff, 10*log10(kRrAf));   % Please note that there is no stem exactly at 50Hz
grid on;


%% Spectral analysis III
close all

qNNB= 10*qNN;
kRrAfB= abs(fft(kRr, qNNB));

bffRawB= 0:(qNNB-1);
kffB= (1/qNNB)*(1/qTs)*bffRawB';

stem(kffB, 10*log10(kRrAfB));   %  Please note that there is no stem exactly at 50Hz
grid on;
```

Considerações sobre Janelas … 1

Antes de mais:

Apresentação para a próxima aula (slides):

O ficheiro Aula25Mar contém uma risca espectral a que frequência?

Variáveis lá presentes: kRr – Sinal
qTs – Período de amostragem

Apresente não só os resultados como também a listagem do código

Considerações sobre Janelas … 1

Janelas há muitas….

OBRIGADO