```
% (ADEEEA-ISEL) - Secção de Automação e Electrónica          fevereiro 2019
% Grupo Diciplinar Controlo               Nome do ficheiro: Apoio_Matlab_V6
% Atualizado em 25 de fevereiro de 2019 por Luis Rocha


%O MATLAB é um software destinado a fazer cálculos com matrizes
%(MATLAB = MATrix LABoratory).
% Link importante do MAtlab     http://www.mathworks.com/

% Visualize o pdf com o publisher deste mfile antes de começar a usar
% Nota:o ficheiro devido à sua dimensão e à grande variadade de temas,deve
% ser estudado ponto a ponto, ou seja, Parte 1 depois Parte 2 e assim
% sucessivamente, fazendo para isso Comment ou Uncomment


close all % limpa  todas as figuras existentes antes de correr o mfile
clear     % limpa todas as variaveis que estao na workspace
clc       % limpa o ecran
%%%%%%%%%%%%%%%%%%%%%%%% 1ª Parte - Atribuiçao de valores, modo basico
disp(' '); disp('1ª Parte - Atribuiçao de valores,modo basico'); disp(' ');

U=230;  % Valor eficaz da tensao na rede (e coloca a variavel na workspace)
R=10;   % Resistencia da instalaçao (e coloca a variavel na workspace)
I=U/R   % Calcula e mostra na linha de comandos o valor da corrente
P=R*I^2 % Calcula e mostra na linha de comandos o valor da potência



% %%%%%%%2ª Parte - Atribuiçao de valores com pergunta na linha de comandos
% clc
% disp(' '); disp(' ');
% disp('2ª Parte - Atribuiçao de valores com pergunta na linha de comandos');
% disp(' ');
% U1=input('Qual o valor da tensao simples da fase 1? U1='); % disp(' ')
% R1=input('Qual o valor da resistencia? R=');
% I1=U1/R1; P1=R1*I1^2; % disp(' ')
% fprintf('O valor da corrente na fase 1 e I= %.1f\n', I1); % disp(' ')
% fprintf('O valor da potencia na fase 1 e P= %.1f\n', P1);
%
% pause  % faz uma pausa no programa
```

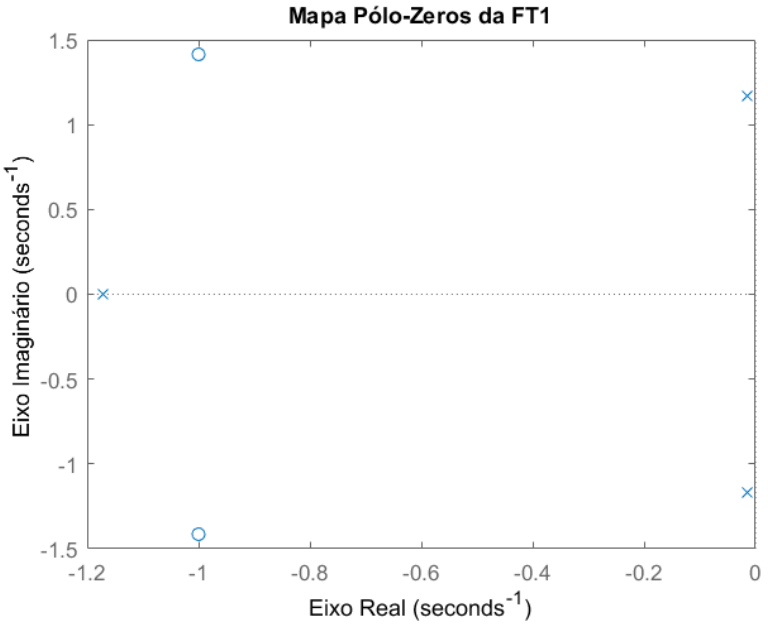
```
1ª Parte - Atribuiçao de valores,modo basico


I =

    23


P =

        5290
```

```
%%%%%%%%%%%%%%%%%%%% 3ª Parte - Funçoes de Transferencia
clc
disp(' '); disp(' '); disp('3ª Parte - Funçoes de Transferencia');
disp(' ');
num=[1 2 3]; den=[5 6 7 8]; disp('FT1');
FT1=tf(num,den) % ou FT1=tf([1 2 3],[5 6 7 8])

%calcula e mostra na linha de comando as raizes caracteristicas FT
Raizes_EQ_caract =eig(FT1 )%calcula as raizes de 1+GH=0 ou os pólos

Zeros_FT1=roots(num) %%calcula e mostra na linha comandos as raizes da FT1
polos_FT1=roots(den)  %calcula e mostra na linha comandos os polos da FT1

% Apresentação de FT na forma factorizada
FT2=zpk([-9 -2],[4 -20 0 0],50);  disp(' ')
disp('Funçao de Transferencia 2, FT2')
FT2
```

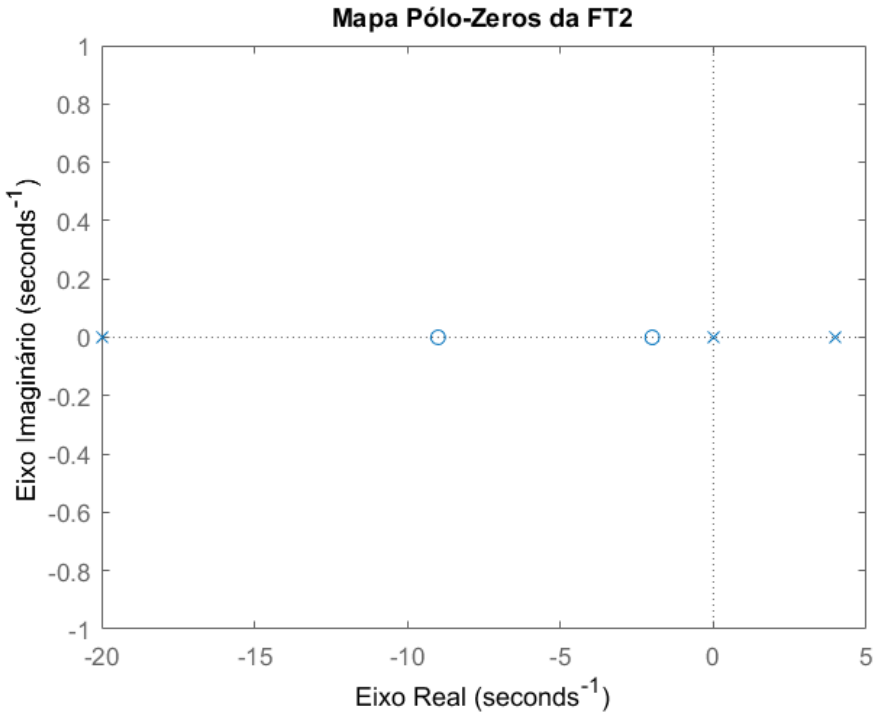
```
3ª Parte - Funçoes de Transferencia

FT1

FT1 =

       s^2 + 2 s + 3
  -----------------------
  5 s^3 + 6 s^2 + 7 s + 8

Continuous-time transfer function.


Raizes_EQ_caract =

  -1.1711 + 0.0000i
  -0.0144 + 1.1688i
  -0.0144 - 1.1688i


Zeros_FT1 =

  -1.0000 + 1.4142i
  -1.0000 - 1.4142i


polos_FT1 =

  -1.1711 + 0.0000i
  -0.0144 + 1.1688i
  -0.0144 - 1.1688i


Funçao de Transferencia 2, FT2

FT2 =

   50 (s+9) (s+2)
  ----------------
  s^2 (s-4) (s+20)
```

```
Continuous-time zero/pole/gain model.
```

```
figure(1)
pzmap(FT1);   % Desenha o mapa-polos zeros da funçao introduzida  A=num  B=den
title('Mapa Pólo-Zeros da FT1');xlabel ('Eixo Real'); ylabel ('Eixo Imaginário');
```




```
figure(2)
pzmap(FT2); title('Mapa Pólo-Zeros da FT2');
xlabel ('Eixo Real'); ylabel ('Eixo Imaginário');
```



```
num_FT3=poly([-1 -1]) % A funçao poly converte raizes em polinomios
den_FT3=poly([-4 -5 0 -6])
disp('Funçao de Transferencia F3 ou FT3=tf(num_FT3,den_FT3)');
FT3=tf(num_FT3,den_FT3)

% A Funçao Series, faz a assosciaçao em cascata, ou em serie funçoes
disp(' '); disp(' '); disp('Associação em cascata de FT2 com FT3');
F=series(FT2,FT3)
```

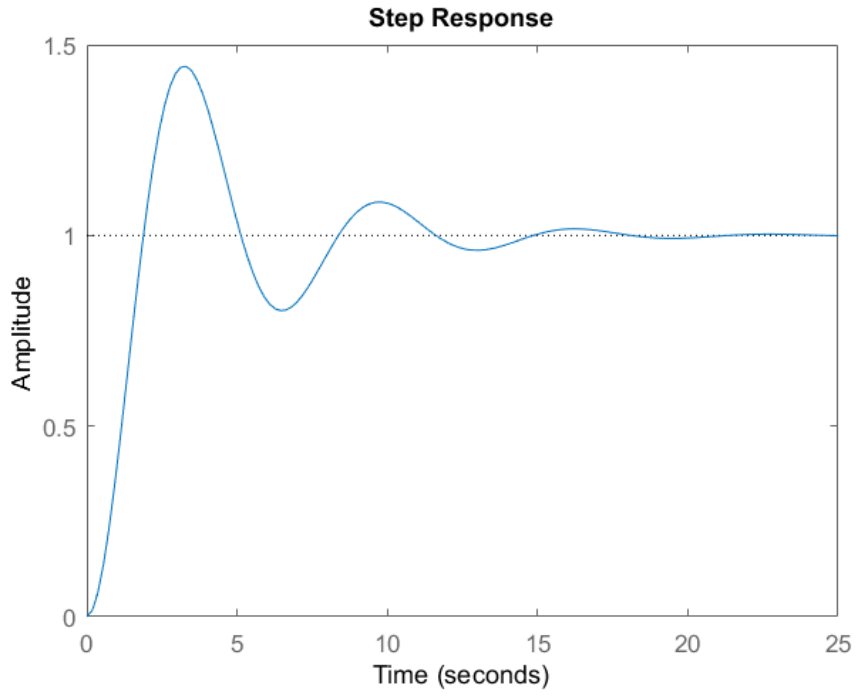
```
num_FT3 =

     1     2     1


den_FT3 =

     1    15    74   120     0

Funçao de Transferencia F3 ou FT3=tf(num_FT3,den_FT3)

FT3 =

          s^2 + 2 s + 1
  ----------------------------
  s^4 + 15 s^3 + 74 s^2 + 120 s

Continuous-time transfer function.



Associação em cascata de FT2 com FT3

F =

        50 (s+1)^2 (s+2) (s+9)
  ---------------------------------
  s^3 (s+6) (s+5) (s+4) (s-4) (s+20)

Continuous-time zero/pole/gain model.
```

```
FT4=tf(4,[1 1]); FT5=zpk([],[],1);

% Faça help na linha de comandos do Matlab para feedback e observe a sintaxe

Z1=feedback(FT4,FT5);  % Retroacçao, de G com H, c/ retroacçao -
Z2=FT4/(1+FT4*FT5);    % Forma manual de calcular G/(1+GH), não simplifica

disp(' '); disp(' ');
% Forma de apresentar informaçao na linha de comandos do Matlab
disp(' Funçao de Transferencia Z1 ou Z1=feedback(FT4,FT5) '); Z1
disp(' '); disp(' ');
disp(' Funçao de Transferencia Z2 ou Z2=FT4/(1+FT4*FT5) '); Z2
disp(' '); disp(' ');
```

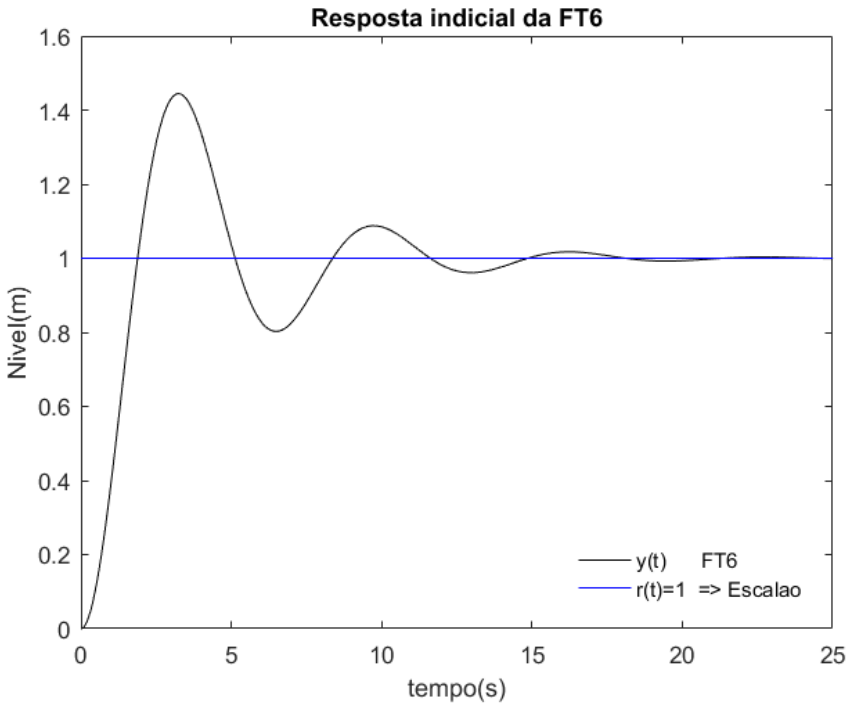
```
 Funçao de Transferencia Z1 ou Z1=feedback(FT4,FT5)

Z1 =

    4
  -----
  (s+5)

Continuous-time zero/pole/gain model.


 Funçao de Transferencia Z2 ou Z2=FT4/(1+FT4*FT5)

Z2 =

    4 (s+1)
  -----------
  (s+1) (s+5)

Continuous-time zero/pole/gain model.
```

```
%%%%%%%%%%%%% 4ª Parte - Respostas temporais e tratamento de graficos
clc
disp(' '); disp(' ');
disp('4ª Parte - Respostas temporais e tratamento de graficos');disp(' ');

figure(3)    % Aplicaçao de x(t)=1 (escalao) sem tratamento de grafico
FT6=tf(1,[1 0.5 1]);
step(FT6)
```

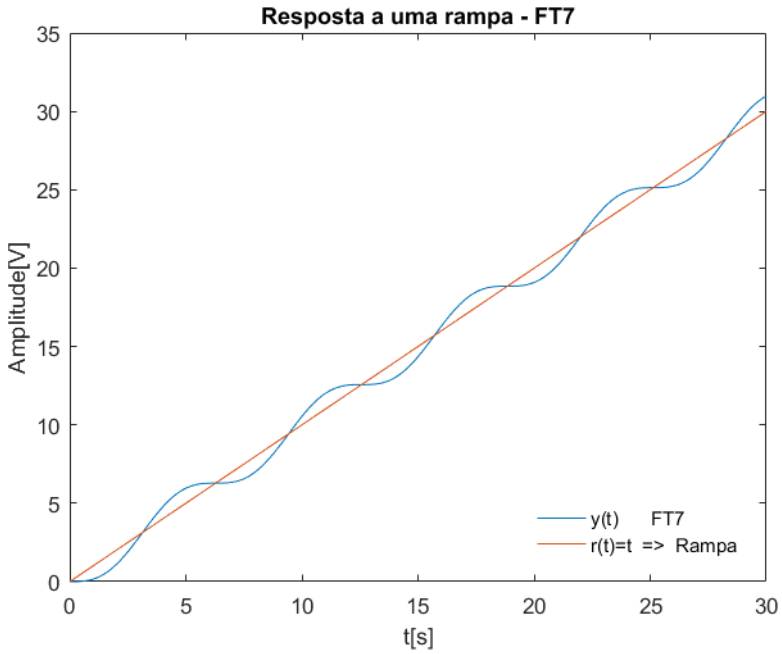
```
4ª Parte - Respostas temporais e tratamento de graficos
```



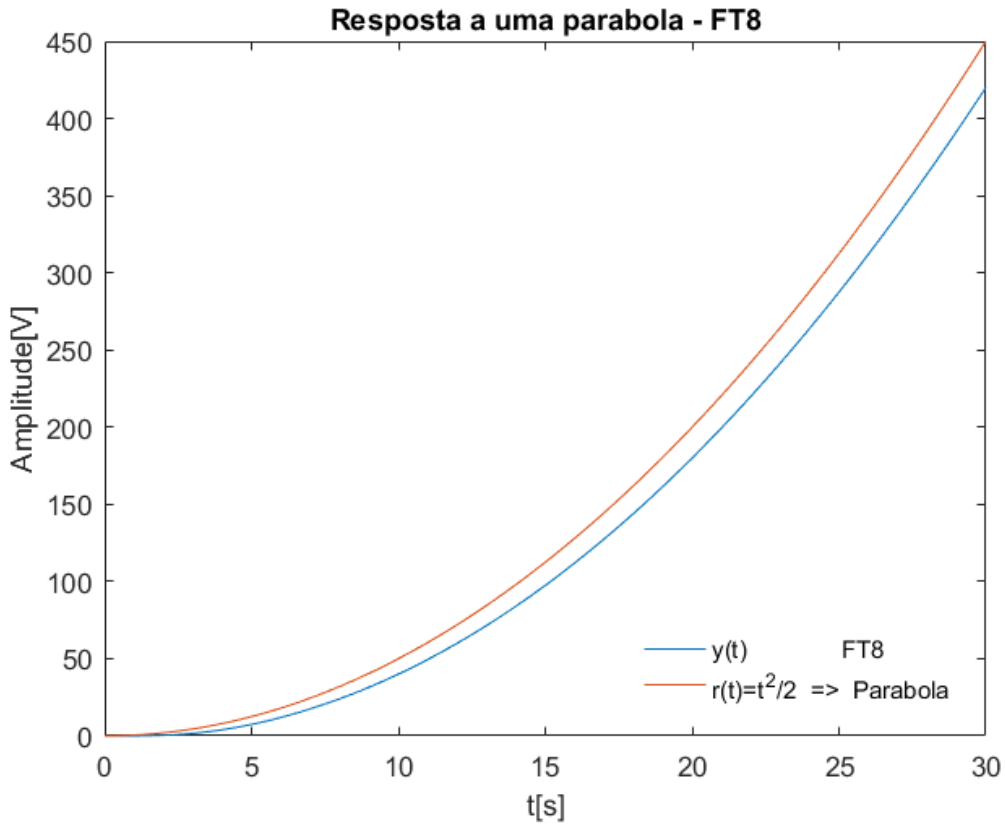
```
figure(4)   % Aplicaçao de x(t)=1 com tratamento de grafico (legendas, etc)
t1=0:0.01:30;
V1=step(FT6,t1);
r1=ones(length(t1),1);  % cria var set-point com dimensão t1
plot(t1,V1,'k-',t1,r1,'b-')
xlabel('tempo(s)'); ylabel('Nivel(m)');
title('Resposta indicial da FT6');
legend('y(t)      FT6','r(t)=1  => Escalao','Location','southeast');
legend('boxoff');
axis([0 25 0 1.6]) % Controla as coordenadas xmin xmax ymin ymax
```



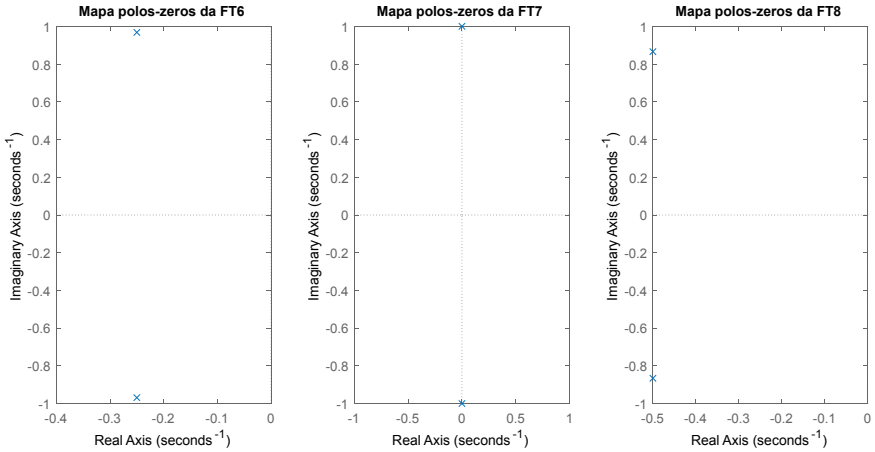

```
figure(5);           % Aplicaçao de uma rampa x(t)=t, usando a funçao lsim
FT7=tf(1,[1 0 1]);
t2=0:0.1:30;
r2=t2;                %entrada do tipo rampa unitaria
V2=lsim(FT7,r2,t2);
plot(t2,V2,'-',t2,r2,'-')
title('Resposta a uma rampa - FT7');
xlabel('t[s]'); ylabel('Amplitude[V]');
legend('y(t)      FT7','r(t)=t  =>  Rampa','Location','southeast')
legend('boxoff')
```



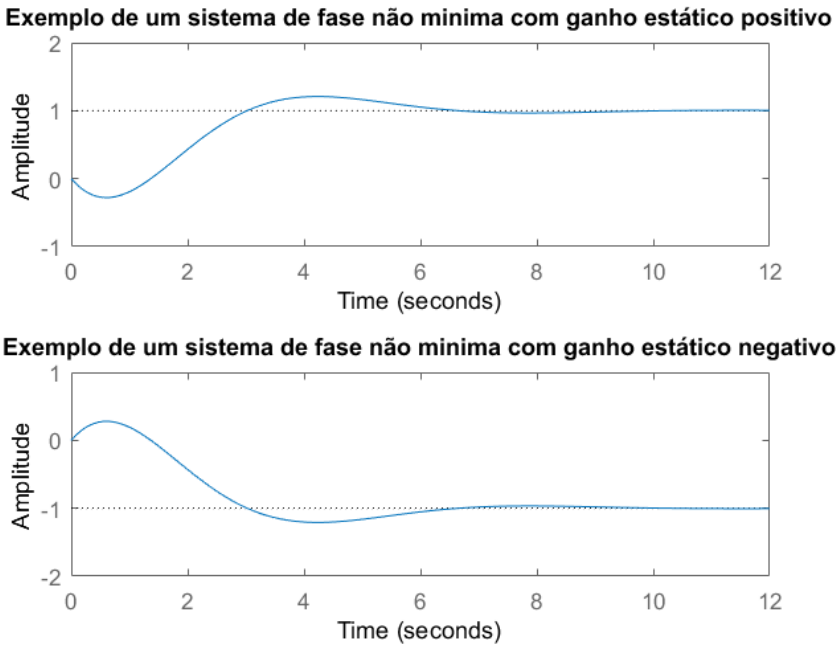

```
figure(6);  % Aplicaçao de uma parabola x(t)=(t^2)/2, usando a funçao lsim
FT8=tf(1,[1 1 1]);
t3=0:0.1:30;  r3=(t3.^2)/2;  % Entrada do tipo parabola unitaria
V3=lsim(FT8,r3,t3);
plot(t3,V3,'-',t3,r3,'-')
title('Resposta a uma parabola - FT8'); xlabel('t[s]');
ylabel('Amplitude[V]');
legend('y(t)              FT8','r(t)=t^2/2  =>  Parabola','Location','southeast')
legend('boxoff')
```



```
figure(7)      % Utilizaçao da funçao Subplot com 1 linha e 3 colunas
subplot(1,3,1)
pzmap(FT6); title('Mapa polos-zeros da FT6');

subplot(1,3,2)
pzmap(FT7); title('Mapa polos-zeros da FT7');

subplot(1,3,3)
pzmap(FT8); title('Mapa polos-zeros da FT8');
```




```
figure(8)     % Utilizaçao da funçao Subplot com 2 linhas e 1 coluna
subplot(2,1,1)
FT_9A=tf([-1 1],[1 1 1]);
step(FT_9A); title('Exemplo de um sistema de fase não minima com ganho estático positivo')

subplot(2,1,2)
FT_9B=tf([1 -1],[1 1 1]);
step(FT_9B); title('Exemplo de um sistema de fase não minima com ganho estático negativo')
```



```
%%%%%%%%%%%%%%%% 5ª Parte - Transformada directa e inversa de laplace
clc
disp(' ');  disp(' ');
disp('5ª Parte - Transformada directa e inversa de laplace') ; disp(' ');

syms s t          % Representaçao simbolica

Y1=4/s^2
y1a=ilaplace(Y1) % output no Matlab com a TL inversa de 4/s^2
```

```
5ª Parte - Transformada directa e inversa de laplace


Y1 =

4/s^2


y1a =

4*t
```

```
y2b=3*t
Y2=laplace(y2b) % output no Matlab a transformada de Laplace de 3t
```

```
 y2b =

3*t


Y2 =

3/s^2
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 6ª Parte - Matrizes
clc
disp(' '); disp('6ª Parte - Matrizes'); disp(' ')

E=[1 0;2 3]
F=[4 5;6 7]
I=E+F
J=E*I
K=sqrt(J)
L=inv(K)
M=L'         % transposta da Matriz L

O=E/F       % 0=E/F é equivalente a O=E*inv(F)     4/2 => 4*inv(2)=2
J=E*inv(F)  % J=O

%Nota2: E preciso cuidado com as operaçoes matriciais, porque para os
%       matematicos nao e possivel dividir matrizes
R=E\F          % R=E/F é equivalente a R=inv(E)*F   4\2 => inv(4)*2=0.5
P=inv(E)*F     % P=R
```

6ª Parte - Matrizes

E =

```
     1     0
     2     3
```

F =

```
     4     5
     6     7
```

I =

```
     5     5
     8    10
```

J =

```
     5     5
    34    40
```

K =

```
    2.2361    2.2361
    5.8310    6.3246
```

L =

```
    5.7302   -2.0259
   -5.2829    2.0259
```

M =

```
    5.7302   -5.2829
   -2.0259    2.0259
```

O =

```
   -3.5000    2.5000
    2.0000   -1.0000
```

J =

```
   -3.5000    2.5000
    2.0000   -1.0000
```

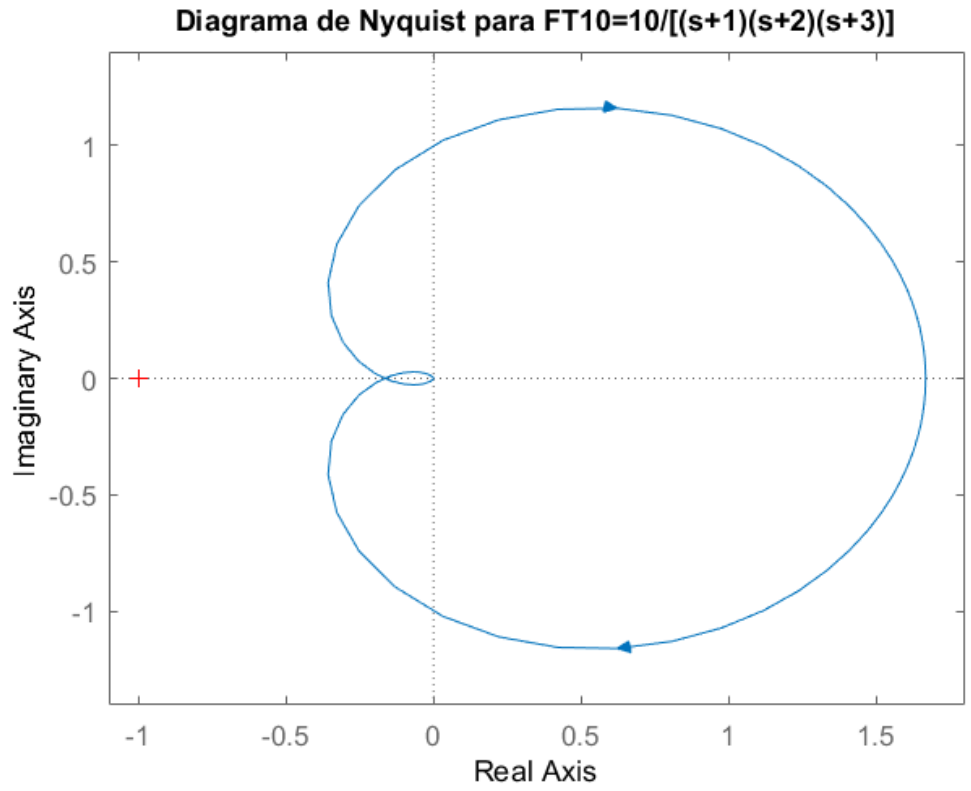
R =

```
    4.0000    5.0000
```

```
   -0.6667   -1.0000

P =

    4.0000    5.0000
   -0.6667   -1.0000
```

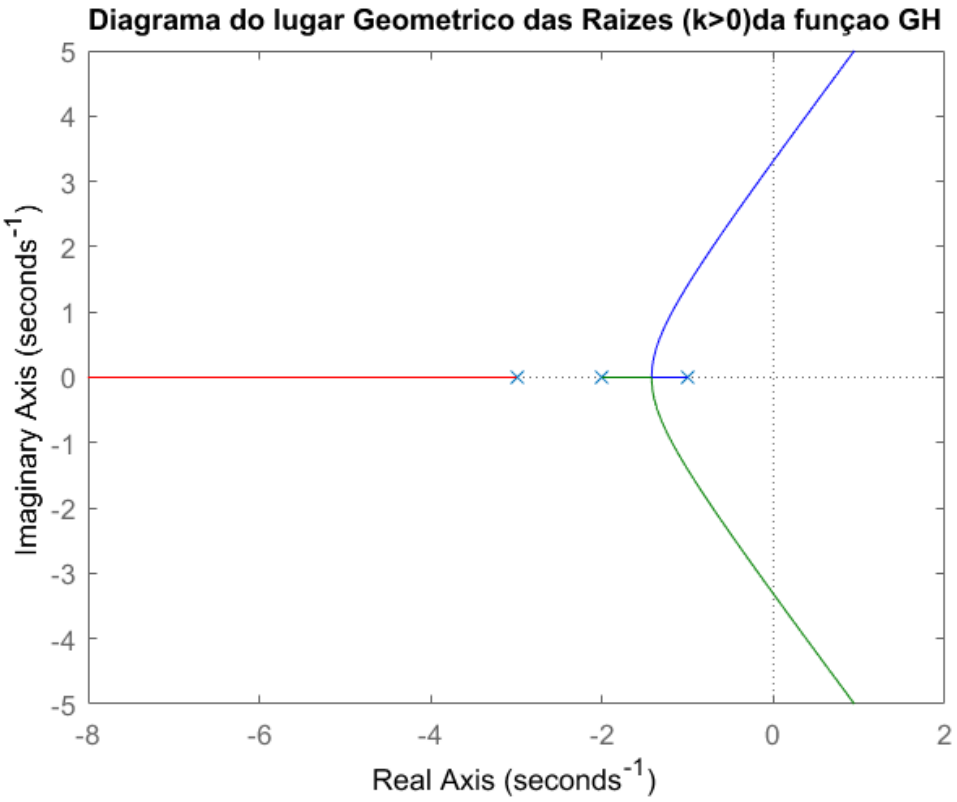
```
%%%%%%%%%7ª Parte - Funçoes para o estudo da estabilidade no MATLAB

%Nota: A função de entrada nos 3 algoritmos de estudo da estabilidade é
%a função de transferência em cadeia aberta (GH), porque 1+GH=0 e depois o
%algoritmo trabalha com GH em função do ponto -1, ou seja GH=-1

clc
disp(' '); disp(' '); disp('7ª Parte - Funçoes de Controlo no MATLAB');
disp('Nyquist, Root Locus e Diagramas de Bode'); disp(' ');

% Diagramas de Nyquist
figure(9)
FT10=zpk([],[-1 -2 -3],10);
nyquist(FT10)
title('Diagrama de Nyquist para FT10=10/[(s+1)(s+2)(s+3)]')
v=[-1.1 1.8 -1.4 1.4]; axis(v)
```

```
7ª Parte - Funçoes de Controlo no MATLAB
Nyquist, Root Locus e Diagramas de Bode
```



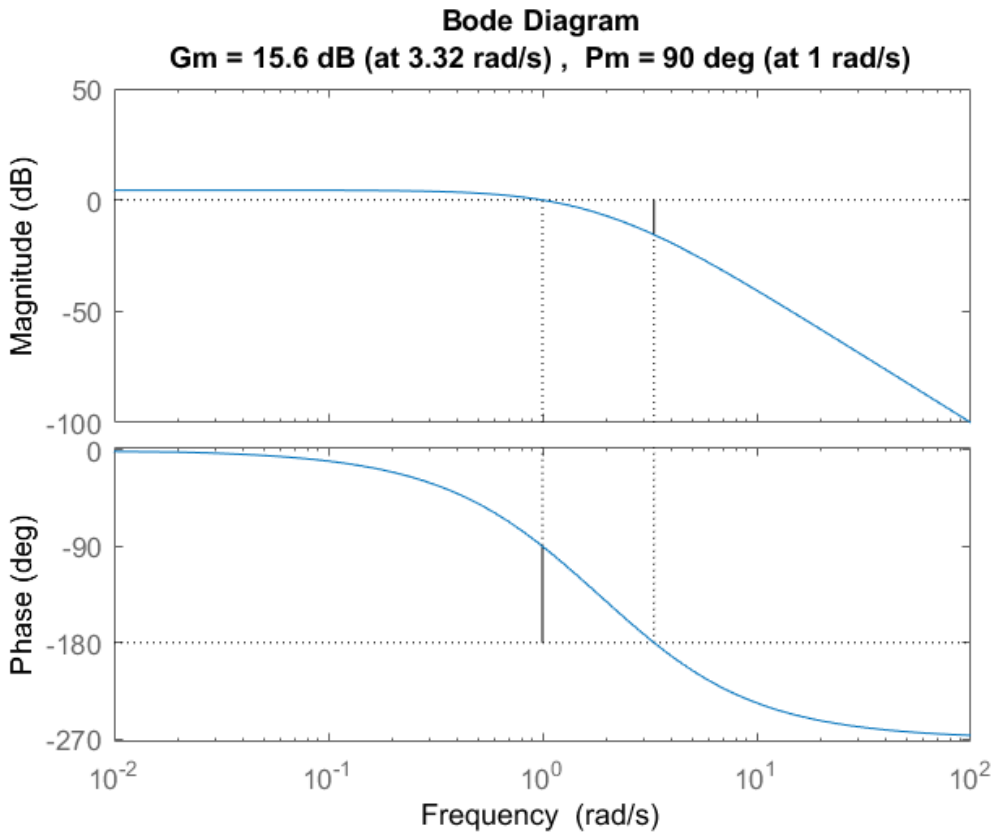
Diagramas do lugar geometrico das raizes - Root Locus

```
figure(10)
rlocus(FT10)
title('Diagrama do lugar Geometrico das Raizes (k>0)da funçao GH')
```



Diagramas de Bode de Amplitude e de fase

```
figure(11)
margin(FT10); % Este comando alem de gerar os 2 diagramas (amplitude e fase),
%apresenta no titulo da figura, a margem de ganho e de fase, estes 2
% indicadores servem para estudar a estabilidade relativa de um sistema

% Existe outro comando para desejar diagramas de Bode => bode(num12,den12)
% no entanto não apresenta na fig. o valor das margens (ganho e de fase)
```




```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 8ª Parte - Variaveis de Estado
clc
disp(' '); disp(' '); disp('8ª Parte - Variaveis de Estado'); disp(' ');

A=[-2 2;1.5 -4]; B=[10/3;0]; C=[0 1]; D=0;
ME1=ss(A,B,C,D) % Função para definir um modelo de estado
```

```
8ª Parte - Variaveis de Estado


ME1 =

  A =
        x1   x2
   x1   -2    2
   x2  1.5   -4

  B =
         u1
   x1  3.333
   x2      0

  C =
       x1  x2
   y1   0   1

  D =
       u1
   y1   0
```

```
Continuous-time state-space model.
```

```
[num11,den11]=ss2tf(A,B,C,D); % Converte um modelo de estado para FT's
FT11=tf(num11,den11)
raizes_ME1=eig(FT11)
```

```
FT11 =

        5
  -------------
  s^2 + 6 s + 5

Continuous-time transfer function.


raizes_ME1 =

   -5.0000
   -1.0000
```

confirmaçao inversa. A partir da FT obter o modelo de estado

```
[A2,B2,C2,D2]=tf2ss(num11,den11); % Converte uma FT em modelo de estado

disp('Modelo de estado 2 => ME2,  diferente do 1º=> ME1');
Sistema2=ss(A2,B2,C2,D2)
% Resposta da confirmaçao: Origina um modelo de estado diferente do 1º
```

```
Modelo de estado 2 => ME2,  diferente do 1º=> ME1

Sistema2 =

  A =
       x1  x2
   x1  -6  -5
   x2   1   0

  B =
       u1
   x1   1
   x2   0

  C =
       x1  x2
   y1   0   5

  D =
       u1
   y1   0

Continuous-time state-space model.
```

Novo objectivo Voltar a obter a FT0=Num0/Den0

```
[num12,den12]=ss2tf(A2,B2,C2,D2);
FT12=tf(num12,den12)
```

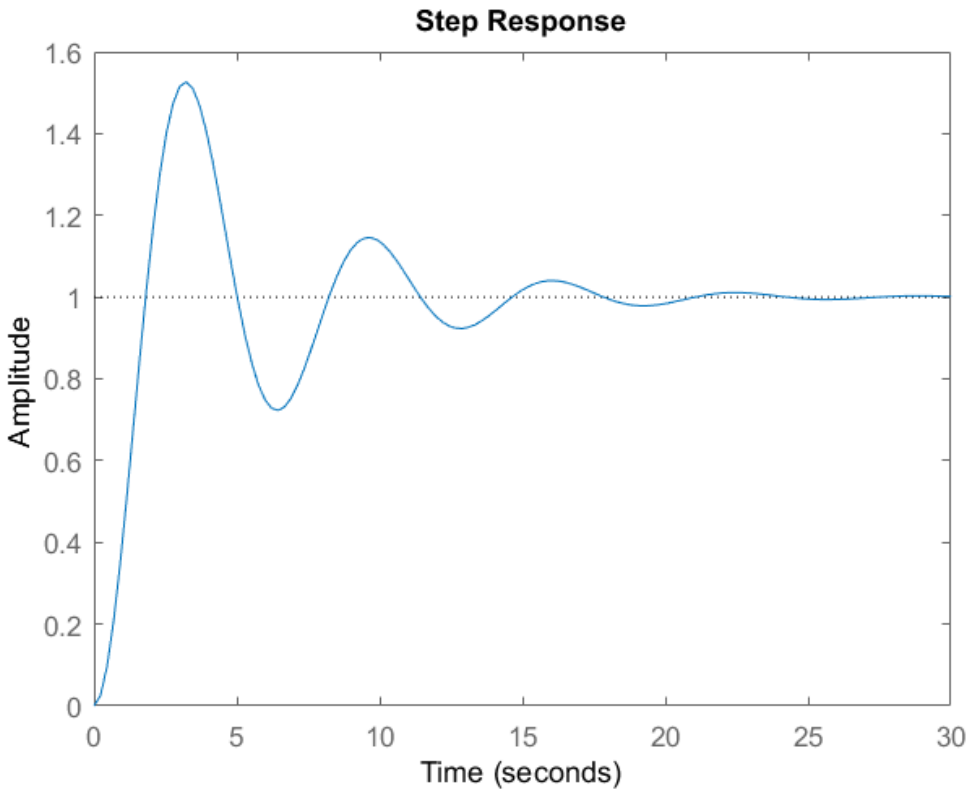
```
FT12 =

        5
  -------------
  s^2 + 6 s + 5

Continuous-time transfer function.
```

```
%%%%%%%%%%%%%%%%%% 9ª Parte - Comando ord2, rmodel e

%comando ord2, permite obter as 4 matrizes para modelo de estado ou uma FT
%em função de um coeficiente de amortecimento z e uma frequência Wn

%%[A,B,C,D]=ord2(Wn, Z) ou %[NUM,DEN]=ord2(Wn,Z)
[A3,B3,C3,D3]=ord2(2,sqrt(2)/2)
[num13,den13]=ord2(2,sqrt(2)/2)
```

```
A3 =

         0    1.0000
   -4.0000   -2.8284


B3 =

     0
     1


C3 =

     1     0


D3 =

     0


num13 =

     1


den13 =

    1.0000    2.8284    4.0000
```

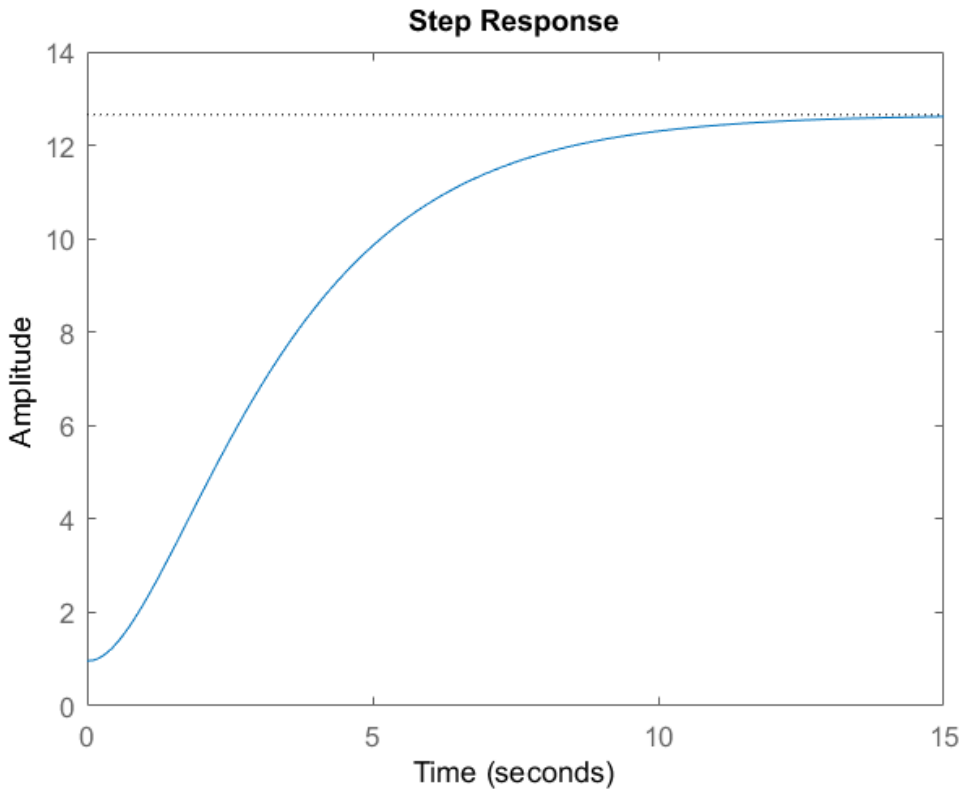
```
clc
disp(' '); disp(' ');
disp('9ª Parte - comandos ord 2, rmodel e damp'); disp(' ');

% % Pergunta e resposta para obter a resposta de um sistema de 2ª ordem
% Z=input('Qual o valor do coeficiente de amortecimento pretendido? Z=');
% disp(' ')
% Wn=input('Qual o valor da frequência natural não amortecida pretendida? Wn=');
Z=0.2;
Wn=1;

% Gera um num e um den de 2ª ordem recebendo um Wn e um Z (coef. de amort.)
[a,b] = ord2(Wn,Z);
FT14=tf(a,b)
figure(90)
step(FT14)
```

```
9ª Parte - comandos ord 2, rmodel e damp


FT14 =

        1
  ---------------
  s^2 + 0.4 s + 1

Continuous-time transfer function.
```



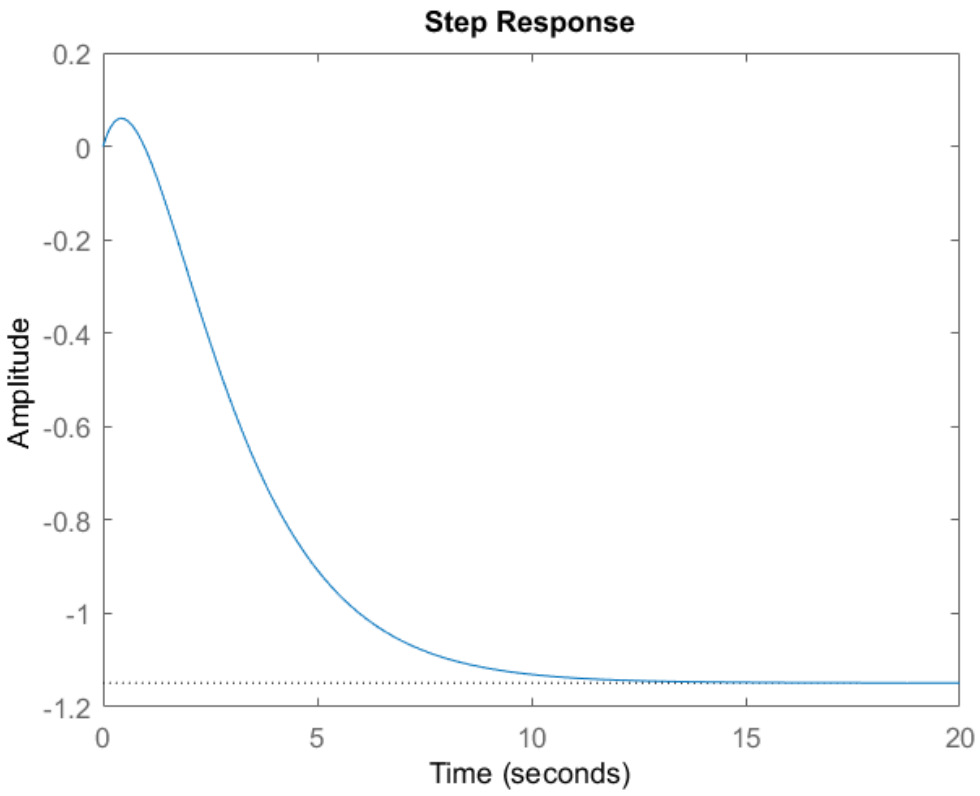
```
disp(' '); disp(' ');
disp('Comando RMODEL  para gerar modelos de 2ª ordem aleatório')
% comando RMODEL  % Geral modelos de 2ª ordem aleatório
for i=2:10
    i
[NUM,DEN]=rmodel(2);
FT15=tf(NUM,DEN); % ou FT15=tf(1,Den)
figure(100+i)
step(FT15)
FT15
```

```
i =

     2


FT15 =

  0.9642 s^2 + 1.029 s + 4.142
  ----------------------------
     s^2 + 1.187 s + 0.3273

Continuous-time transfer function.
```

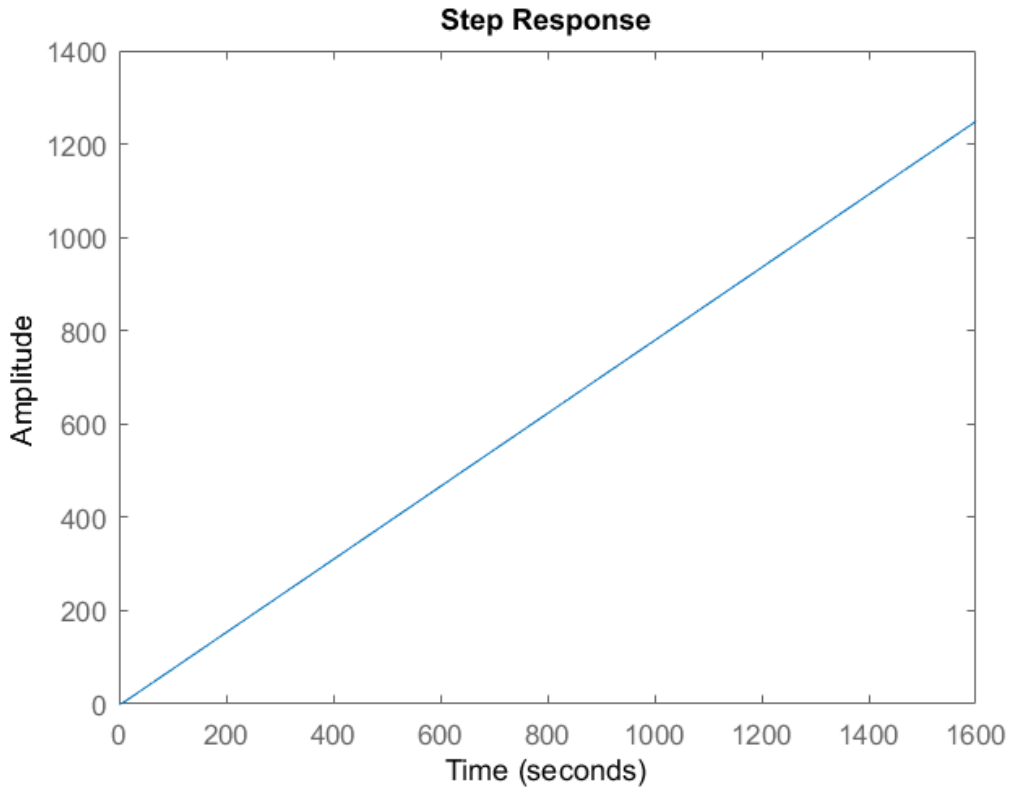



```
i =

     3
```

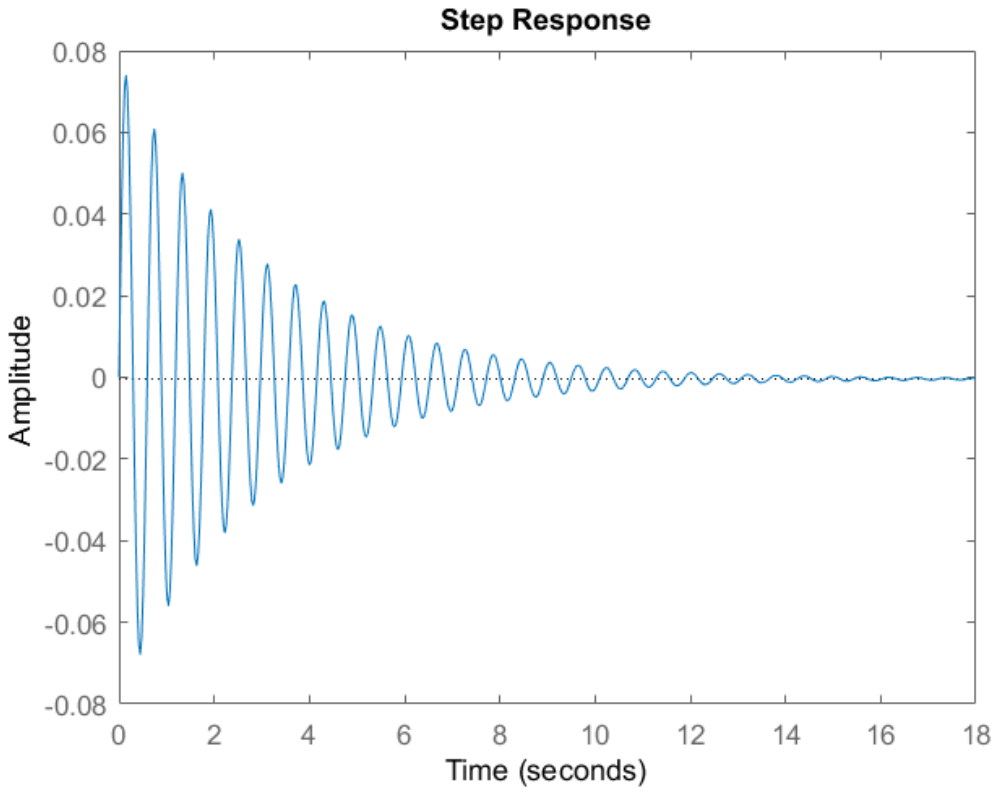
```
FT15 =

    0.3075 s - 0.5091
  ----------------------
  s^2 + 1.354 s + 0.4432

Continuous-time transfer function.
```



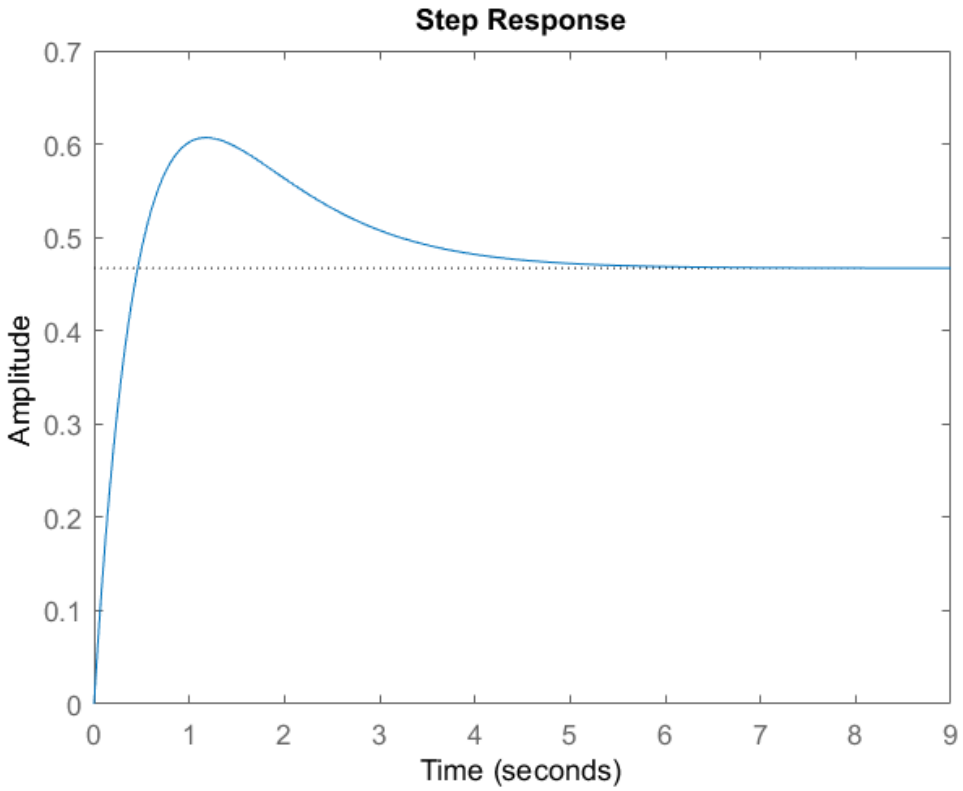

```
i =

     4


FT15 =

      0.1837
  --------------
  s^2 + 0.2348 s

Continuous-time transfer function.
```



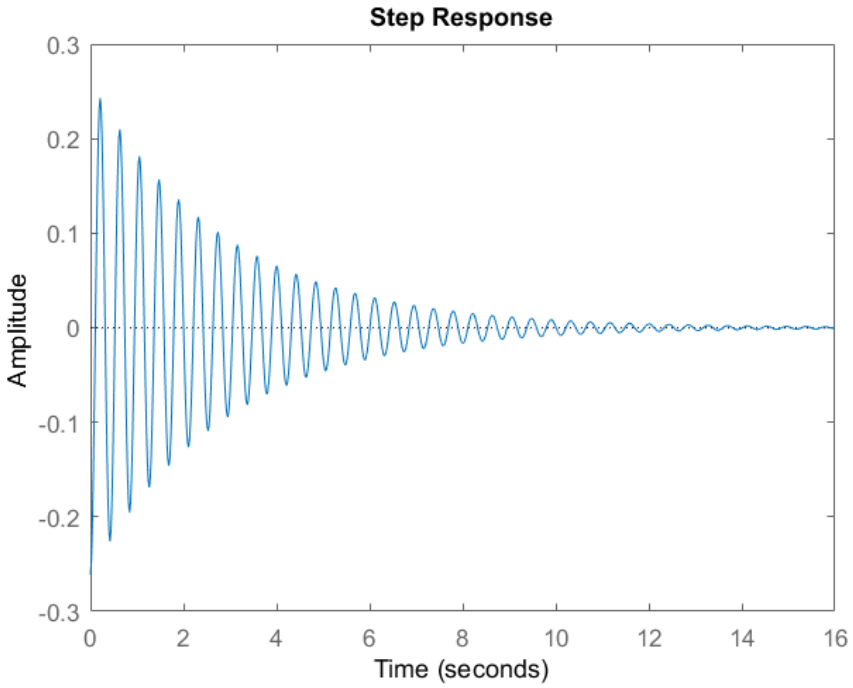

```
i =

     5


FT15 =

    0.8261 s - 0.04237
  ----------------------
  s^2 + 0.6541 s + 112.1

Continuous-time transfer function.
```



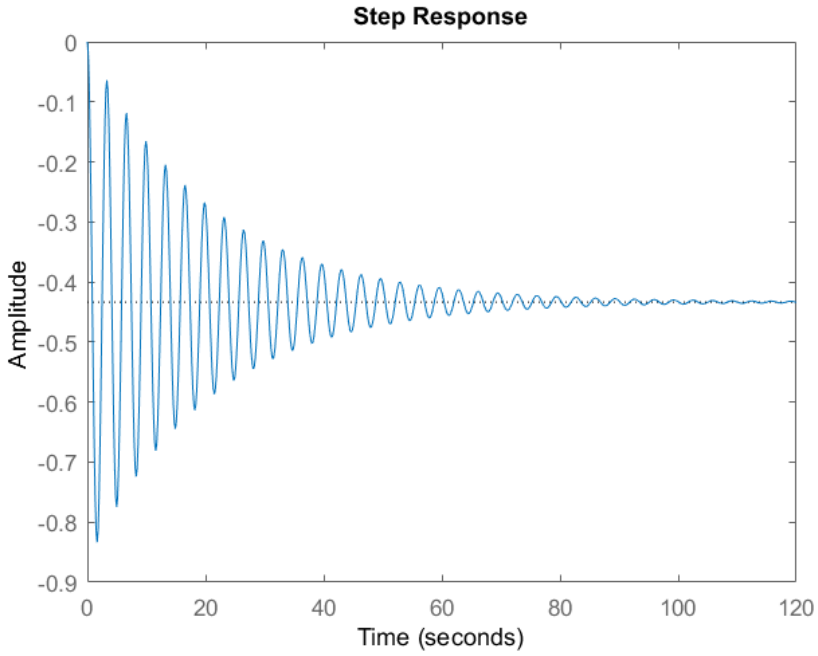

```
i =

     6


FT15 =

    1.682 s + 0.8949
  ---------------------
  s^2 + 2.819 s + 1.916

Continuous-time transfer function.
```



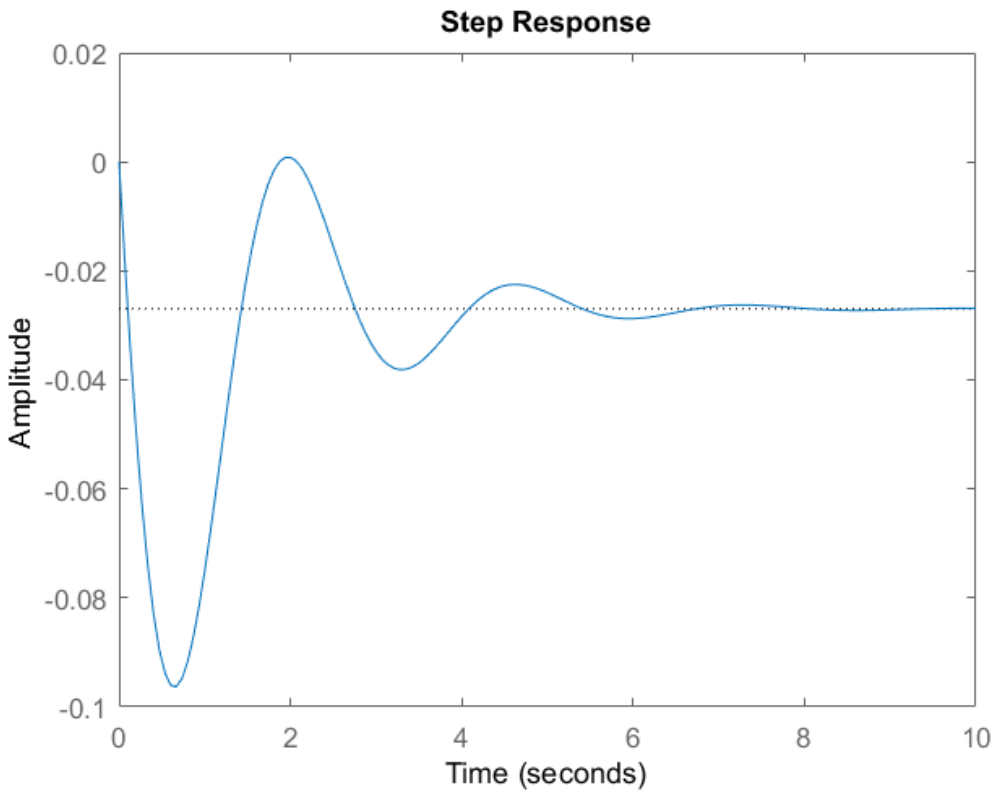

```
i =

     7


FT15 =

  -0.2612 s^2 - 0.08145 s - 0.02626
  ---------------------------------
        s^2 + 0.69 s + 222.4

Continuous-time transfer function.
```



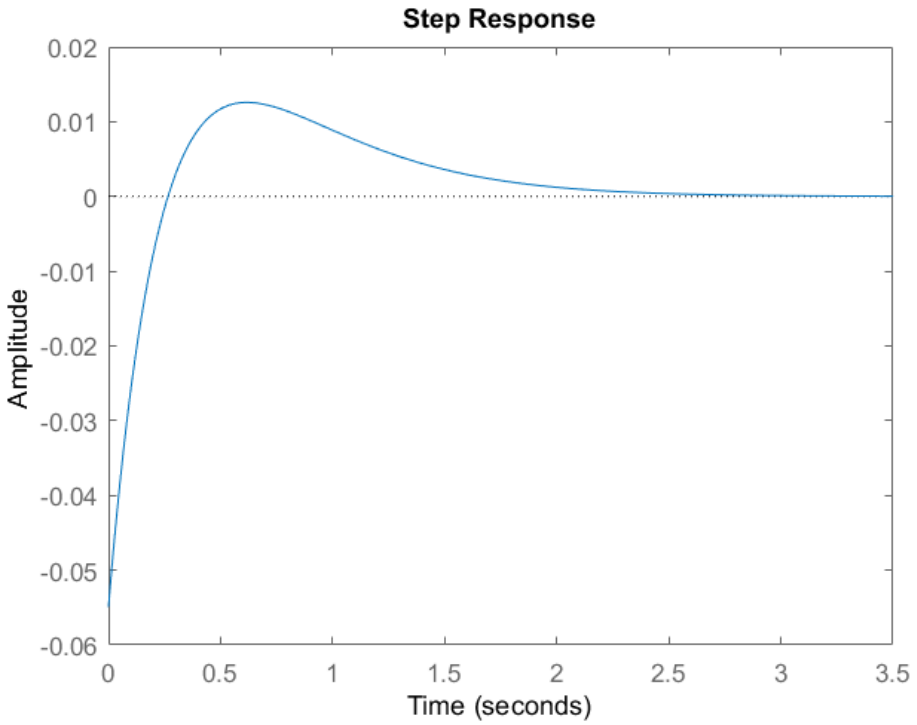

```
i =

     8


FT15 =

          -1.565
  ----------------------
  s^2 + 0.09671 s + 3.61

Continuous-time transfer function.
```



```
i =

     9


FT15 =

   -0.2781 s - 0.1641
  ---------------------
  s^2 + 1.378 s + 6.083

Continuous-time transfer function.
```




```
i =

    10


FT15 =

  -0.05492 s^2 + 0.05079 s - 2.532e-06
  ------------------------------------
         s^2 + 5.687 s + 8.074

Continuous-time transfer function.
```




```
%pause

end
```

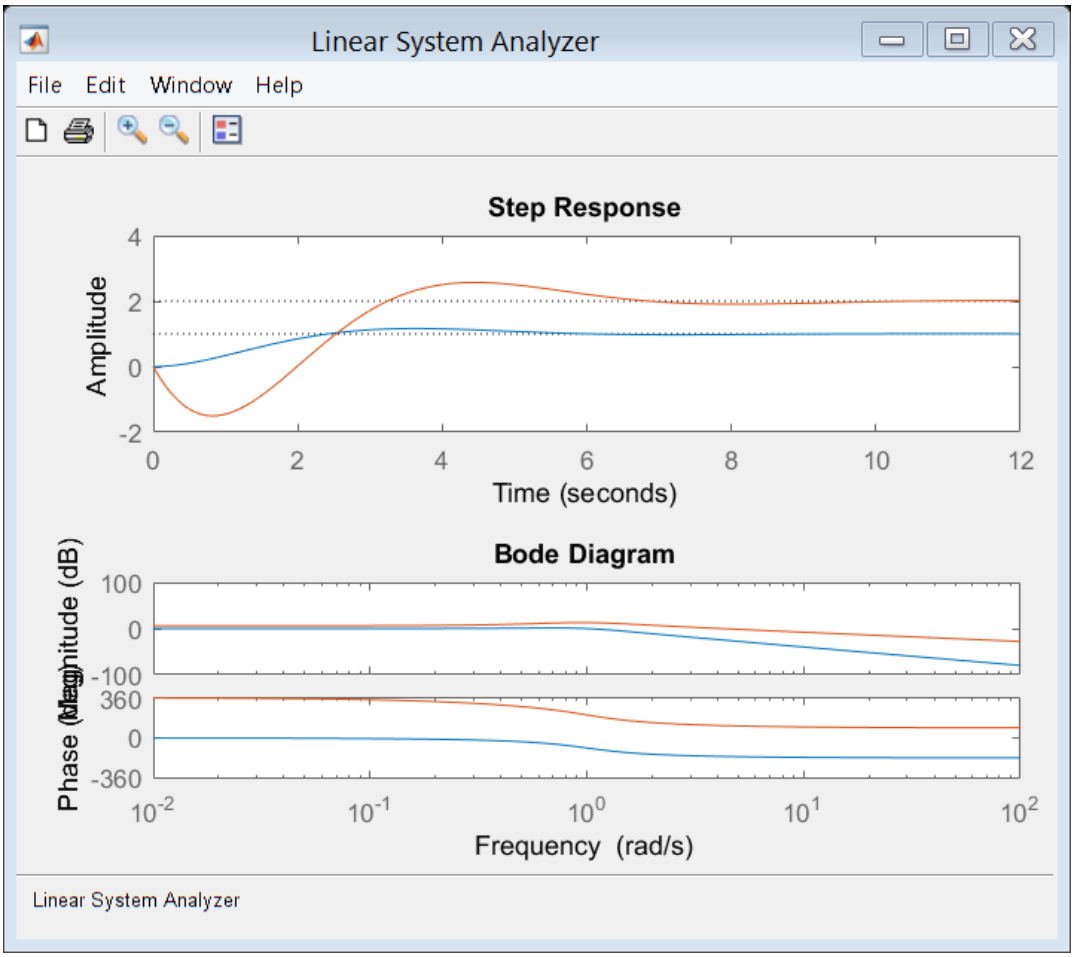
Comando RMODEL para gerar modelos de 2ª ordem aleatório

```
%comando DAMP [Wn,Z] ou  DAMP(SYS)
% Permite obter o coeficiente de amortecimento e Wo e depois calcular Taus
FT16=tf(1,[1 6 8])
Polos_FT16=eig(FT16) % raizes da equação caracteristica FT16
[Wn,Z] = damp(FT16) % Indica qual o Wn e o Z de um sistema linear
Tau1=1/(Wn(1,1)*Z(1,1));  Tau2=1/(Wn(2,1)*Z(2,1));
fprintf('O valor da constante de tempo 1, Tau1= %.3f\n', Tau1);  disp(' ')
fprintf('O valor da constante de tempo 2, Tau2= %.3f\n', Tau2);  disp(' ')
```

```
FT16 =

        1
  -------------
  s^2 + 6 s + 8

Continuous-time transfer function.

Polos_FT16 =

    -4
    -2

Wn =
     2
     4
```

```
Z =

     1
     1

O valor da constante de tempo 1, Tau1= 0.500

O valor da constante de tempo 2, Tau2= 0.250
```

```
%%%%%%%%%%%%%%%%%% 10ª Parte - ltview

FT_17=tf([1],[1 1 1]);
FT_18=tf([-4 2],[1 1 1]);
ltiview({'step';'bode'},FT_17,FT_18)

%É possível inserir legenda diretamente da figura gerada (ao lado direito da
% lupa -, e para retirar a borda fazer edge a branco (nas opções da caixa)

%
% disp(' '); disp(' ');
% %disp('Prima qualquer tecla para fechar todas as figuras.....'); %pause;
%
% close all %(fecha todas as figuras abertas)
```



Published with MATLAB® R2016b