INSTITUTO POLITÉCNICO DE TOMAR

Engenharia Electrotécnica

Matlab Tutorials

Principais Funções do Matlab

Funções Básicas do Matlab

O Matlab consiste num programa interactivo de computação numérica e visualização de dados. Este programa é muito utilizado na área de controlo, quer na análise de dados quer a nível de projecto, incluindo diversas *toolboxes* que extendem as funções básicas do Matlab, as quais irão ser revistas de seguida, em diversificadas áreas de aplicação.

Vectores

Colocar cada elemento do vector entre parêntesis rectos (cada elemento deve estar separado por um espaço). O vector deve ser atribuído a uma variável, tal como é apresentado de seguida:

```
a = [1 2 3 4 5 6 9 8 7]

b = a + 2

     b =
          3   4   5   6   7   8   11   10   9

a =
   1   2   3   4   5   6   9   8   7
```

Caso se pretenda criar um vector com elementos entre 0 e 20 separados igualmente por elementos de 2 (este método é frequentemente utilizado quando se pretende criar vectores de tempo), deve-se escrever o seguinte comando:

```
t = 0:2:20
```

O Matlab devolve:

```
t =
   0   2  4  6  8  10  12  14  16  18  20
```

Manipulação de Vectores

A manipulação de vectores é igualmente fácil. Suponha que pretende adicionar o valor 2 a cada um dos elementos do vector a, deve então fazer o seguinte:

```
b = a + 2
```

O Matlab devolve:

```
b =
   3   4   5   6   7   8   11   10   9
```

Supor agora que se pretende somar o vector a com o vector b, desde que estes tenham a mesma dimensão (tenham o mesmo número de elementos) a operação é bastante simples, bastando fazer:

```
c = a + b
```

O Matlab devolve:

```
c =
   4   6  8  10  12  14  20  18  16
```

Funções

O Matlab contém várias funções matemáticas, tais como: *sin*, *cos*, *log*, *exp*, *sqrt* e muitas outras. Algumas constantes, tais como *pi*, *i* ou *j* (utilizadas para a raiz quadrada de –1) são também incorporadas no Matlab.

```
sin(pi/4)
```

O Matlab devolve:

```
ans =
     0.7071
```

Quando não se atribui nenhuma variável a uma determinada função ou valor, o Matlab atribui-lhe uma variável por defeito – *ans*.

Como criar as suas próprias funções no Matlab

Quando se introduz um comando, tais como, `roots`, `plot`, ou `step`, no Matlab, o que se está realmente a fazer é a correr um m-file com determinadas entradas e saídas as quais são introduzidas com o intuito de realizarem uma tarefa em particular. Este tipo de ficheiros são semelhantes a uma sub-rotina, a qual tem variáveis de entrada (parâmetros que são passados para o m-file), saídas (valores que retornam do m-file) e um corpo de comandos o qual pode conter variáveis locais. É possível escrever as nossas próprias funções através da utilização do comando `function`.

A nova função tem de ter um nome com uma extensão '.m'. Esta função deve ser guardada na mesma directoria do restante software. A primeira linha deste ficheiro deve conter a sintaxe desta função, do seguinte modo:

```
function [output1,output2] = filename(input1,input2,input3)
```

A função pode ter o número de entradas e saídas que se achar necessário. Sugere-se que a seguir ao comando `function, se inclua o seguinte comando:`

```
error(nargchk(x,y,nargin));
```

O x e o y consistem respectivamente no menor e no maior número de entradas que podem ser aceites pela função; caso se introduzam mais ou menos entradas que as definidas um erro é lançado.

As funções podem ser complexas e é requerida muita prática com o Matlab para que estas possam ser definidas de um modo fiável e eficaz. De seguida apresenta-se um exemplo simples de uma função denominada de `add.m`:

```
function [var3] = add(var1,var2)
%add is a function that adds two numbers
var3 = var1+var2;
```

Se guardar estas três linhas num ficheiro denominado "add.m" na directoria do Matlab, pode depois utilizá-la através do comando:

```
y = add(3,8)
```

Obviamente que a maioria das funções são bastante mais complexas do que a que foi apresentada. Este exemplo pretende apenas mostrar a forma básica da função.

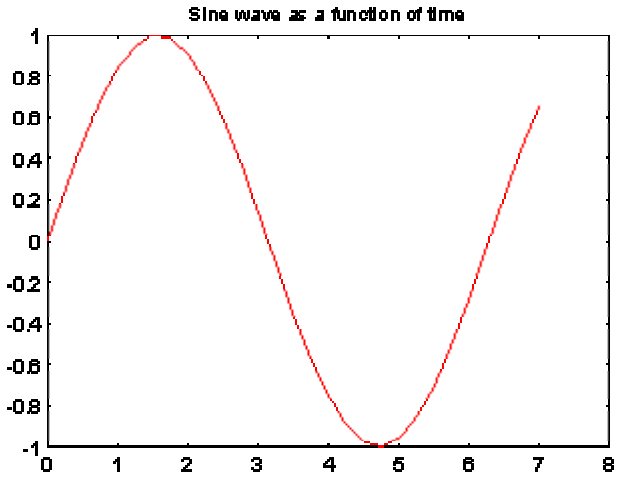
Gráficos

É muito simples criar gráficos com o Matlab. Suponha que pretende criar um gráfico de uma função seno (função do tempo). O procedimento é o seguinte: Criar um vector de tempo - `t`; Definir um vector – `y`, em que cada elemento consiste num valor da função seno para cada valor de `t`. Note-se que tanto o vector `y` como o vector `t` têm a mesma dimensão.

```
t=0:0.25:7;
y = sin(t);
plot(t,y)
```

Nota: Deve-se colocar sempre um ponto e vírgula a seguir a cada comando para evitar que os valores de cada variável estejam sempre a aparecer, cada vez que estas sejam calculadas.

O Matlab devolve:

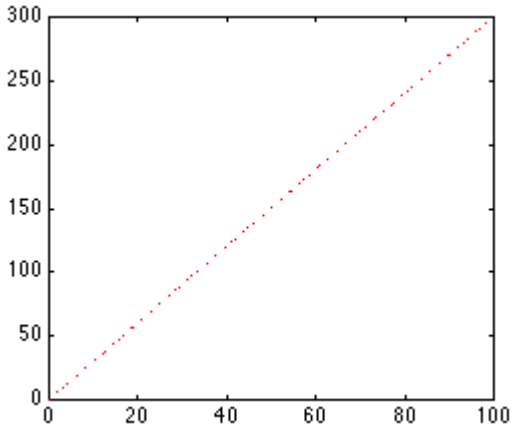


Estética do Gráfico

Adicionando um terceiro parâmetro na função `plot` é possível alterar quer a cor quer o padrão da linha do gráfico. Observe o seguinte exemplo:

```
x = 0:0.1:100;
y = 3*x;
plot(x,y,'r:')
```

O novo gráfico tem o seguinte aspecto:



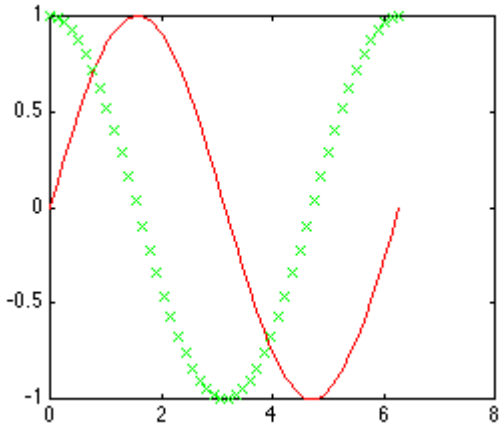
O terceiro parâmetro consiste num dos três caracteres que definem a cor e o padrão. A lista de caracteres é apresentada de seguida:

```
y     amarelo       .     ponto
m     magenta       o     círculo
c     oceano        x     x-mark
r     vermelho      +     mais
g     verde         -     sólido
b     azul          *     asterisco
w     branco        :     ponteado
k     preto         -.    tracejado com ponto
                    --    tracejado
```

É possível colocar mais do que uma função no mesmo gráfico. Suponha que pretende colocar uma função seno e uma função co-seno no mesmo gráfico, tendo estas cor e padrão diferentes:

```
x = linspace(0,2*pi,50);
y = sin(x);
z = cos(x);
plot(x,y,'r', x,z,'gx')
```

O gráfico teria o seguinte aspecto:



O mesmo efeito pode ser obtido utilizando as funções `hold on` e `hold off` tal como é apresentado de seguida:

```
x = linspace(0,2*pi,50);
y = sin(x);
plot(x,y,'r')
z = cos(x);
hold on
plot(x,z,'gx')
hold off
```

Nota: o comando **`x = linspace(0,2*pi,50)`**, faz o mesmo que **`x = 0: 2*pi: 50`**.

Subplotting

A mesma figura pode ter mais do que um gráfico. Para que tal seja possível é necessário utilizar a função `subplot`, tal como é apresentado de seguida:

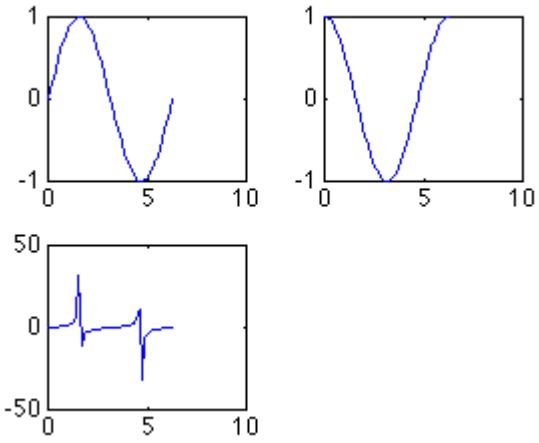
```
subplot(m,n,p)
```

Este comando divide a figura numa matriz de m linhas e n colunas, podendo deste modo criar m*n gráficos em cada figura. Veja-se o seguinte exemplo:

```
x = linspace(0,2*pi,50);
y = sin(x);
z = cos(x);
w = tan(x);

subplot(2,2,1)
plot(x,y)
subplot(2,2,2)
plot(x,z)
subplot(2,2,3)
plot(x,w)
```

O Matlab devolve:



Embora tenha sido criada uma matriz 2*2 para quatro gráficos, apenas três foram utilizados. Era extremamente simples adicionar um quarto gráfico através do comando `subplot(2,2,4)`. Deste modo é fácil perceber que o parâmetro `p` indica a ordem em que o gráfico irá ser apresentado.

Alterar os Eixos

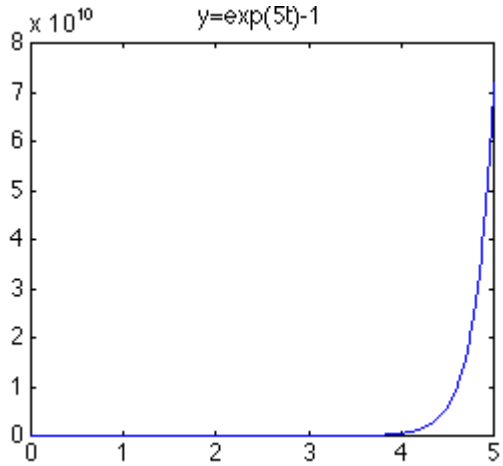
O comando `axis` é responsável por alterar as dimensões dos eixos. Este comando deve ser colocado antes da função `plot`.

```
axis([xmin, xmax, ymin, ymax])
```

Por exemplo, suponha que pretende realizar um gráfico da função `y = exp(5t)-1`:

```
t=0:0.01:5;
y=exp(5*t)-1;
plot(t,y)
```

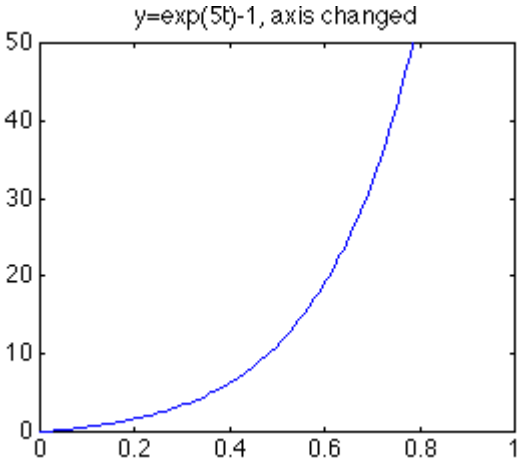
O Matlab devolve:



Se antes do código que é apresentado anteriormente colocar o segionte comando:

```
axis([0, 1, 0, 50])
```

Terá como resultado:

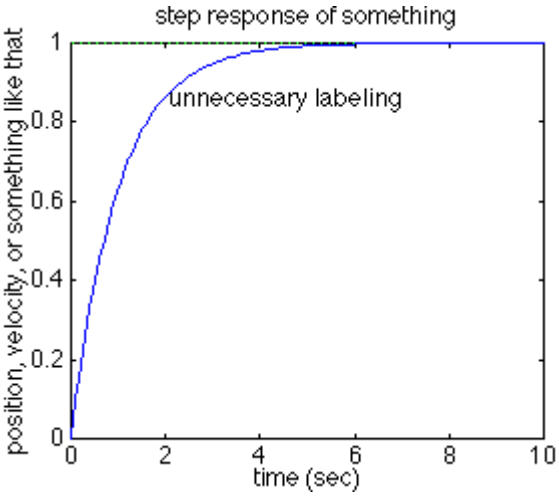


Adicionar Texto aos Gráficos

É possível colocar texto nos gráficos através da utilização de três comandos: `title, x-label e y-label`. Observe o seguinte exemplo:

```
title('step response of something')
xlabel('time (sec)')
ylabel('position, velocity, or something like that')
gtext('unnecessary labeling')
```

O resultado é o seguinte:



O comando `gtext`, é utilizado para colocar outras legendas com o rato.

Outras Funções Úteis

- `clf` (Limpa o gráfico actual);
- `figure` (Abre uma nova figura, de modo a que a figura prévia seja guardada);
- `close` (Fecha a janela da figura actual);
- `loglog` (Faz o mesmo que a função plot, no entanto os eixos são à escala logarítmica);
- `semilogx` (Faz o mesmo que a função plot, no entanto o eixo dos xx é à escala logarítmica);
- `semilogy` (Faz o mesmo que a função plot, no entanto o eixo dos yy é à escala logarítmica);
- `grid` (Adiciona uma grelha ao gráfico).

Polinómios

Em Matlab um polinómio é representado por um vector. Para criar um polinómio basta introduzir os coeficientes do polinómio num vector pela ordem descendente. Por exemplo, suponha que pretende introduzir o seguinte polinómio:

$$s^4+3s^3-15s^2-2s+9$$

Para colocar este polinómio no Matlab, basta fazer:

```
x = [1 3 -15 -2 9]
```

```
x =
   1   3  -15  -2   9
```

Se alguns dos coeficientes dos polinómios forem zeros, estes também devem ser introduzidos na ordem correcta: Veja-se o seguinte exemplo:

$$s^4+1$$

Para colocar este polinómio no Matlab, basta fazer:

```
y = [1 0 0 0 1]
```

Para calcular o valor do polinómio para determinado valor da variável s, deve-se usar a função `polyval`.

```
z = polyval([1 0 0 0 1],2)
```

Terá como resultado:

```
z =
   17
```

O comando `roots` serve para calcular as raízes do polinómio. Observe os seguinte exemplo:

$$s^4+3s^3-15s^2-2s+9$$

Para calcular as raízes do polinómio, basta fazer:

```
roots([1 3 -15 -2 9])

ans =
   -5.5745
    2.5836
   -0.7951
    0.7860
```

Suponha agora que pretende multiplicar dois polinómios. Para tal terá de realizar a função convolução (`conv`) do seguinte modo:

```
x = [1 2];
y = [1 4 8];
z = conv(x,y)

z =
   1  6  16  16
```

Para dividir dois polinómios utiliza-se a função `deconv`. A função devolve dois valores, nomeadamente o resultado e o resto da divisão. Voltando ao exemplo anterior, caso pretenda dividir o polinómio z por y, devo preceder do seguinte modo:

```
[xx, R] = deconv(z,y)
```

Terá como resultado:

```
xx =
    1  2

R =
   0  0  0  0
```

No caso de pretender somar ou subtrair polinómios podem ocorrer duas situações distintas. Se os polinómios tiverem a mesma dimensão não há problema bastando fazer o mesmo que foi feito no caso dos vectores, caso tenham dimensões distintas terá de realizar algumas operações para colocar primeiro os polinómios com a mesma dimensão.

Matrizes

Colocar uma matriz no Matlab é praticamente igual a colocar um vector. A única diferença é que as linhas da matriz vêm separadas por ponto e vírgula.

```
B = [1 2 3 4;5 6 7 8;9 10 11 12]
```

Terá como resultado:

```
B =
   1    2    3    4
   5    6    7    8
   9   10   11   12
```

Outro modo de definir a matriz:

```
B = [ 1  2  3  4
      5  6  7  8
      9 10 11 12]
```

Terá como resultado:

```
B =
   1    2    3    4
   5    6    7    8
   9   10   11   12
```

Caso pretenda encontrar a transposta de uma matriz, basta fazer:

```
C = B'
```

Terá como resultado:

```
C =
   1   5    9
   2   6   10
   3   7   11
   4   8   12
```

Nota: se a matriz C fosse complexa caso se utilizasse o comando anterior obter-se-ía a complexa conjugada da transposta. Neste caso, se quiser obter somente a transposta deve utilizar o comando .'B. Caso a matriz não seja complexa os comandos anteriores fazem exactamente o mesmo.

Se pretender multiplicar a matriz B pela matriz C, deve fazer do seguinte modo:

```
D = B * C
```

Terá como resultado:

```
D =
    30    70   110
    70   174   278
   110   278   446
```

Se antes fizer:

```
D  = C * B
```

Terá como resultado:

```
D =
   107   122   137   152
   122   140   158   176
   137   158   179   200
   152   176   200   224
```

Se pretender multiplicar os elementos correspondentes de duas matrizes pode utilizxar o seguinte comando:

```
E = [1 2;3 4]
F = [2 3;4 5]
G = E .* F
```

Terá como resultado:

```
E =
   1   2
   3   4

F =
   2   3
   4   5

G =
   2   6
  12  20
```

Pode também levantar uma matriz a um determinado expoente (multiplicá-la por ela própria um determinado número de vezes), se utilizar o seguinte comando (considere a matriz E do exemplo anterior):

```
E^3
```

Terá como resultado:

```
ans =
    37    54
    81   118
```

Se quiser que cada elemento da matriz seja elevado, isoladamente, a um determinado expoente, deve fazer do seguinte modo:

```
E.^3
```

Terá como resultado:

```
ans =
     1     8
    27    64
```

Para calcular a inversa da matriz `E` deve proceder do seguinte modo:

```
X = inv(E)
```

Terá como resultado:

```
X =
   -2.0000    1.0000
    1.5000   -0.5000
```

Para calcular os valores próprios da matriz `E`:

```
eig(E)
```

Terá como resultado:

```
ans =
     -0.3723
      5.3723
```

Para determinar os coeficientes do polinómio característico da matriz `E`:

```
p = poly(E)
```

Terá como resultado:

```
p =
   1.0000   -5.0000   -2.0000
```

Não esquecer que os valores próprios da matriz `E`, serão iguais às raízes do polinómio `p`.

```
roots(p)
```

Terá como resultado:

```
ans =
    5.3723
   -0.3723
```

INSTITUTO POLITÉCNICO DE TOMAR

Engenharia Electrotécnica

Lista de alguns dos comandos mais utilizados no Matlab

Comando	Descrição
abs	Valor absoluto
acker	Cálcula a matriz K de modo a colocar os pólos de A-BK, ver também a função place
axis	Estabelece a escala do gráfico, ver também as funções plot e figure
bode	Desenha o diagrama de Bode, ver também as funções logspace, margin, nyquist1
c2dm	Converte o sistema contínuo para um sistema discreto
clf	Limpar a figura - Clear figure (usar a função clg no Matlab 3.5)
conv	Convolução (utilizada na multiplicação de polinómios), ver também a a função deconv
ctrb	Determina a matriz de controlabilidade, ver também a função obsv
deconv	Deconvolução e divisão polinomial, ver também a função conv
det	Calcular o determinante da matriz
dimpulse	Resposta a impulso de um sistema discreto linear, ver também a função dstep
dlqr	Projecto de um regulador linear quadrático para sistemas discretos, ver também, a função lqr
dlsim	Simulação de sistemas lineares discretos, ver também a função lsim
dstep	Resposta a degrau de um sistema linear discreto, ver também a função stairs
eig	Calcula os valores próprios de uma matriz
eps	Tolerância numérica do Matlab
feedback	Realiza a realimentação de dois sistemas
figure	Cria uma nova figura ou redefine a figura actual, ver também as funções subplot, axis
for	Cíclo For
format	Formatação numérica (dígitos mais significativos, expoentes)
function	Cria uma função
grid	Desenha uma grelha no gráfico actual
gtext	Adiciona um pedaço de texto ao gráfico actual, ver também a função text

help	HELP!
hold	Assegura o gráfico actual, ver também a função figure
if	Execução condicional
imag	Retorna a parte imaginária de um número complexo, ver também a função real
impulse	Resposta a impulso de um sistema linear contínuo, ver também as funções step, lsim, dlsim
input	Permite que o utilizador introduza uma entrada
inv	Calcula o inverso de uma matriz
jgrid	Gera as linhas do factor de amortecimento (zeta) e do tempo de estabelecimento (sigma), ver também as funções sgrid, sigrid, zgrid
legend	Legenda do gráfico
length	Dimensão de um vector, ver também a função size
linspace	Retorna um vector linearmente espaçado
lnyquist1	Produz um diagrama de Nyquist numa escala logarítmica, ver também a função nyquist1
log	Logaritmo natural, ver também log10: logaritmo comum
loglog	Gráfico com escala logarítmica, ver também semilogx/semilogy
logspace	Retorna um vector espaçado a uma escala logarítmica
lqr	Regulador linear quadrático para um sistema linear contínuo, ver também a função dlqr
lsim	Simula um sistema linear contínuo, ver também as funções step, impulse, dlsim.
margin	Retorna as margens de ganho e de fase e frquências de canto, ver também as funções bode
norm	Calcula a norma de um vector
nyquist1	Desenha o diagrama de Nyquist, ver também lnyquist1.
obsv	Calcula a matriz de observabilidade, ver também ctrb
ones	Retorna um vector ou uma matriz de uns, ver também a função zeros
place	Calcula a matriz K de modo a colocar os pólos de A-BK, ver também a função acker
plot	Desenha um gráfico, ver também figure, axis, subplot.

poly	Retorna o polinómio característico
polyadd	Adiciona dois polinómios
polyval	Calcula o valor do polinómio, dando um determinado valor à variável deste
print	Imprime o gráfico actual (para uma impressora ou para um ficheiro ps)
pzmap	Mapeamento de pólos e zeros de um sistema linear
rank	Calcula a característica de uma matriz
real	Retorna a parte real de um número complexo, ver também a função imag
rlocfind	Determina os valores de k e dos pólos num determinado ponto
rlocus	Desenha o lugar das raízes
roots	Determina as raízes de um polinómio
rscale	Determina o factor de escala de um sistema realimentado
set	Set(gca,'Xtick',xticks,'Ytick',yticks) para controlar o número e espaçamento de marcas dos eixos
series	Conexão em série de sistemas independentes lineares e temporais
sgrid	Gera as linhas do factor de amortecimento (onde ele é constante) (zeta) e da frequência natural não amortecida (Wn), ver também jgrid, sigrid, zgrid
sigrid	Gera as linhas onde o tempo de estabelecimento é constante (sigma), ver também jgrid, sgrid, zgrid
size	Retorna a dimensão de uma matriz ou de um vector, ver também length
sqrt	Calcula a raíz quadrada
ss	Cria um modelo em espaço de estados ou converte um sistema linear invariante no tempo para espaço de estados, ver também a função tf
ss2tf	Converte um modelo em espaço de estados para função de transferência, ver também a função tf2ss
ss2zp	Converte um modelo em espaço de estados para uma representação em pólos e zeros, ver também a função zp2ss
stairs	Define um gráfico em escada de uma resposta discreta, ver também a função dstep
step	Desenha a resposta a degrau, ver também impulse, lsim, dlsim.
subplot	Divide a janela dos gráficos, possibilitando a apresentação de vários gráficos, ver também plot, figure

text	Adiciona um pedaço de texto no gráfico actual, ver também title, xlabel, ylabel, gtext
tf	Cria uma função de transferância ou converte para função de transferância, ver também ss
tf2ss	Converte uma função de transferência para uma representação em espaço de estados, ver também a função ss2tf
tf2zp	Converte um função de transferência para uma representação de pólos e zeros, ver também a função zp2tf
title	Adiciona um título ao gráfico actual
wbw	Retorna a largura de banda, sendo necessário introduzir o factor de amortecimento e tempos de subida e de amortecimento
xlabel/ylabel	Adiciona uma legenda aos eixos dos xx e dos yy respectivamente, ver também title, text, gtext
zeros	Retorna uma matriz ou um vector de zeros
zgrid	Gera as linhas onde o factor de amortecimento é constante (zeta) e frequência natural não amortecida (Wn), ver também as funções sgrid, jgrid, sigrid
zp2ss	Converte uma representação em pólos e zeros para um modelo em espaço de estados, ver também a função ss2zp
zp2tf	Converte uma representação em pólos e zeros para uma função de transferência, ver também tf2zp