



## EE-210/2017

### Lista de Exercícios 2 (opcional)

#### Familiarização com MATLAB

#### Operações Elementares, Matrizes, Gráficos e Programação Básica

Abrir o programa ®MATLAB e praticar a sua utilização através de alguns exemplos simples.

Em princípio, basta digitar os caracteres que aparecem seguindo o prompt ">" , mas é claro que as experimentações (do tipo *what if*) são fundamentais.

Para sair do MATLAB digite: > quit

Para obter ajuda digite: > help

Em caso de pânico digite: <ctrl>-c

#### 1. Expressões Aritméticas Básicas

```
> a = 5
> b = 2;           %Sera que este ";" faz alguma diferenca?
> c = a + b;
> c
> a - b
> d = a - b
> f = a/b
> g = a\b
> h = a ^ b
> k = sqrt(b)
> l = log10(b)
> m = exp(b)
> e = log(m)
> pi              %Cuidado para não fazer pi=2, por exemplo
> sin(pi/2)
> cos(2\pi)       %Para converter (2\pi) em radianos:(180/pi)*(2\pi)
> tan(-0.250*pi)
```

#### 2. Operações com Matrizes

```
> A = [ 1 2 3 ; 4 5 6 ]

> B = [ 9 8 7 ;
        6 5 4 ] ;

> B

% Tente tambem E1 = [ 1,2,3 ] e
% E2 = [ 1 <Enter> 2 <Enter> 3 ]
% Sera que B e b representam a mesma entidade?
```

```

> b %Quando que b foi definido?
> C = A + B
> D = B' %Transposicao de matrizes
> F = A * D
> G = A' * B %Algo de errado se tentar G = A * B ?
> X = A(1:2,2) %Extração de partes de matrizes
> Y = B(2,2:3)
> Z = B(:,3)

```

### 3. Operações com Arrays (ou seja, um caso particular de Matriz)

```

> t = 0:0.2:2 %Geralmente se faz t=0:0.2:2;
> u = 11:-1:1 %Incrementos negativos permitidos!
> v = t .* u
> z = t ./ u
> w = u .^ 2
> y = 2 .^ u %Que acontece se y = 2.^ u
> AA = A .* A
> AA = A ./ A
> AEX = exp(A)

```

### 4. Traçado de Gráficos

```

> x = 0:pi/25:2*pi;
> y1 = sin(x);
> plot(x,y1)
> clf %Que faz isto?
> y2 = exp(x);
> plot(x,y2,'*r')
> semilogy(x,y2,'+')
> title('EE-210/2017')
> xlabel('log de x')
> ylabel('y')
> figure(2) %Que faz isto?
> plot(x,y1)
> subplot(2,1,1) % 2,1 = 2 linhas 1 coluna
> plot(x,y1,'--')
> subplot(2,1,2)
> plot(x,cos(x),'--r')
> figure(3)
> x=0:0.2:6.28;
> [xx,yy]=meshgrid(x,y);
> z=sin(xx) .* cos(yy);
> mesh(x,y,z)
> surf(xx,yy,z)
> surfc(xx,yy,z)
> contour(x,y,z,6)
> contour3(x,y,z,6)

```

## 5. Números Complexos

```
> z1 = 1+1i
> z2 = 4+5i
> z3 = z2 - z1
> abs(z3)
> angle(z3)
> z3r = real(z3)
> z3i = imag(z3)
> atan(z3i/z3r)
```

## 6. Controle de Fluxo

```
> f=1;
> for i=1:5           %Uso do "for"
    f=f*i
end

> f=1;
> i=1;
> while i < 6         %Uso do "while"
    f=f*i
    i=i+1;
end

> angulo = input('Digite numero menor que pi/2 -> ')
> if abs(angulo) > pi/2
> disp('Entrada Invalida')
> else
> tangente=tan(angulo)
> end
```

## 7. Inline functions

```
func = inline('x^2 + x^2')           %Não esquecer os apóstrofes
modz = func(z3r,z3i)
```

## 8. Definição de Funções tipo m-file

Na barra de ferramentas: file → New → Function

No editor, definir as saídas, a função e as entradas:

```
function [soma,produto,soma_dos_quadrados] = aritmetica(x,y)
soma = x + y;
produto = x .* y;
soma_dos_quadrados = x^2 + y^2;
end
```

Na barra de ferramentas: file → save as (nome que quiser). Usualmente é boa prática usar o mesmo nome que aparece na "function", no caso, "*aritmética*". Tomar cuidado com o diretório em que está salvando o m-file".

```
> [a,b,c] = aritmética(2,3)      %Deve resultar a=5, b = 6 e c = 13
```

## 9. Ferramentas básicas de controle

```
> servo = tf([1 1],[2 3 4 0])
> bode(servo)
> s=zpk('s')                % ou s = tf('s')
> g=(s+1)/(s^2+1*s+4)
> rlocus(g)
> rltool(g)
> nyquist(g)
> grid on                    % que são os círculos que foram traçados?
> nichols(g)
> grid on
> % mais ferramentas use "> help control"
```

## 10. Equações diferenciais ordinárias

Selecionar pelos "tabs": File -> New -> Function

Criar a função f:

```
function [ xdot ] = f(t,x )
xdot=[x(1)-x(1)*x(2);-x(2)+x(1)*x(2)];
end
```

Salvar com algum nome interessante, por exemplo xd.m

Na tela principal, supondo  $t_0 = 0$  e  $t_f = 100$ , a partir da condição inicial  $x_0 = [2 \ 2]'$ , chamar uma rotina de solução de ODEs:

```
> [t,x]=ode45(@xd,[0 100],[2 2])
```

Plotar:

```
> plot(x(:,1),x(:,2))
> plot(t,x)
```