

# Aula 2

## Operações com Matrizes e Vetores

### Gráficos 2D e 3D.

# Criando matrizes (2D)

Matrizes: podem ser criadas das mesmas formas, separando as linhas com “,”

No caso, temos uma matriz 2x4

Podemos também criar vetores a partir de matrizes!!

The screenshot displays the MATLAB Command Window and Workspace. The Command Window shows the following commands and outputs:

```
>> %% Criamos matrizes (2D) da mesma forma
>> %% Agora, separando as linhas com ";"
>> Mat1=[0 0.5 1 1.5; 2 3 4 5]

Mat1 =

    0    0.5000    1.0000    1.5000
    2    3.0000    4.0000    5.0000

>> %% ou, usando o operador ":"
>> Mat2=[0:0.5:1.5;2:5]

Mat2 =

    0    0.5000    1.0000    1.5000
    2    3.0000    4.0000    5.0000

>> %% Criando um vetor a partir de uma matriz
>> Vec1=Mat1(1,:)

Vec1 =

    0    0.5000    1.0000    1.5000

>> % Vec1 é a 1a linha de Mat1
>> Vec2=Mat1(:,2)

Vec2 =

    0.5000
    3.0000

>> % Vec2 é a 2a coluna de Mat1
>> Vec1(2)

ans =

    0.5000
```

The Workspace window shows the following variables and their values:

Name	Value
Mat1	[0,0.5000,1,1.5000;2,3,4,5]
Mat2	[0,0.5000,1,1.5000;2,3,4,5]
Vec1	[0,0.5000,1,1.5000]
Vec2	[0.5000;3]
ans	3

The Command History window shows the following commands:

```
Mat2=[0:0.5:1.5;2:5]
%% Criando um vetor a partir de uma matriz
Vec1=Mat1(1,:)
% Vec1 é a 1a linha de Mat1
% Vec2 é a 2a coluna de Mat1
clear
%% Criamos matrizes (2D) da mesma forma
%% Agora, separando as linhas com ";"
Mat1=[0 0.5 1 1.5; 2 3 4 5]
%% ou, usando o operador ":"
Mat2=[0:0.5:1.5;2:5]
%% Criando um vetor a partir de uma matriz
Vec1=Mat1(1,:)
% Vec1 é a 1a linha de Mat1
Vec2=Mat1(:,2)
```

# Criando matrizes: concatenação.

```
Editor - C:\Users\Gregorio\Documents\Gregorio\Matlab_files\CursoFisicaComputacional\Introducao\MatLab\Aula1_Extra_Concat\matrizes.m
EDITOR PUBLISH VIEW
1 %% Concatenando vetores: linhas
2 clear;
3 N=3;
4 Vec1=1:N;
5 Mat1=Vec1;
6 for ii=2:N
7     Mat1=[Mat1;Vec1];
8 end
9 %% Mat1 será N repeticoes de Vec1
10 %%%%%%%%%
11 %% Concatenando vetores: colunas
12 Vec2=(1:N)'; % Vetor coluna
13 Mat2=Vec2; % Inicializa
14 for ii =2:N
15     Mat2=[Mat2 Vec2];
16 end
```

Neste for loop, a matriz foi criada recursivamente “empilhando” vetores

No caso, temos uma matriz  $N \times N = 3 \times 3$

Podemos também criar a matriz concatenando colunas.

Aula1\_Extra\_ConcatMatrizes.m x + script Ln 16 Col 4

# Tarefas da aula 2 (início)

Lembrando: toda aula haverá tarefas!! (20% da média final!!!)

- Tarefa 1: *Gere uma matriz 20x20* (elementos à sua escolha)

Dica: use o Help e veja comandos como `rand()` e `ones()`.

# Vetores: adição e multiplicação.

Command Window

```
>> vec1=[0 2];  
>> vec2=[1 3];  
>> vec1 + vec2
```

```
ans =
```

```
1 5
```

```
>> (vec1)' % Transposto!
```

```
ans =
```

```
0  
2
```

Para o vetor/variável não ser impresso na tela, use ";" no final da linha.

Operador ' indica transposição de vetores ou matrizes.




fx


# Vetores: adição e multiplicação.

```
Command Window
>> vec1*(vec2) '
ans =
    6
>> (vec2)'*vec1
ans =
    0    2
    0    6
fx >>
```

Multiplicação segue a regra de multiplicação de matrizes:


$$(0 \ 2) \cdot \begin{pmatrix} 1 \\ 3 \end{pmatrix} = 6$$

Podemos criar matrizes via multiplicação de vetores!


$$\begin{pmatrix} 1 \\ 3 \end{pmatrix} \cdot (0 \ 2) = \begin{pmatrix} 0 & 2 \\ 0 & 6 \end{pmatrix}$$

O que ocorre se fizermos  $vec1*vec2$ ??

# Operações elemento a elemento.

```
Command Window
>> vecN=1:10

vecN =

     1     2     3     4     5     6     7     8     9    10

>> Nquadrado=vecN.*vecN
Nquadrado =

     1     4     9    16    25    36    49    64    81   100

fx >> |
```

Para operações como multiplicação elemento a elemento, usamos o operador “.”

Tente fazer “vecN^2”. Dá certo? Por quê??

# Operações elemento a elemento.

Command Window

```
>> vecN=1:10
```

```
vecN =
```

```
     1     2     3     4     5     6     7     8     9    10
```

```
>> vecN.^2
```

```
ans =
```

```
     1     4     9    16    25    36    49    64    81   100
```

```
fx >> |
```

Colocando “.” antes da operação, o MatLab interpreta como uma operação elemento a elemento.



# Usando Vetores: funções.

Command Window

```
>> vecx=0:pi/2:2*pi
```

Crio o vetor “vecx”: inicia em 0 e termina em 2pi com espaçamento de pi/2.

```
vecx =
```

```
0    1.5708    3.1416    4.7124    6.2832
```

```
>> func=sin(vecx)
```

```
func =
```

```
0    1.0000    0.0000   -1.0000   -0.0000
```

*fx* >> |

“func” armazena um vetor em que cada elemento  $\text{func}(i)$  corresponde a seno de  $\text{vecx}(i)$ .

# Tarefas Aula 2:

Lembrando: toda aula haverá tarefas!! (20% da média final!!!)

- Tarefa 2: *Dados dois vetores cartesianos* 
$$\begin{cases} \vec{v}_1 = 2\mathbf{i} + 4\mathbf{j} \\ \vec{v}_2 = 3\mathbf{i} + 5\mathbf{j} \end{cases}$$

*calcule:*

- *Os módulos  $|v_1+v_2|$  e  $|v_1-v_2|$*
- *O ângulo de  $v_3=v_1+v_2$  em relação ao eixo x.*
- *O vetor  $v_4=A.v_1$  onde é uma matriz 2x2 (\*)  $A=(v_1+v_2)^T.(v_2)$*

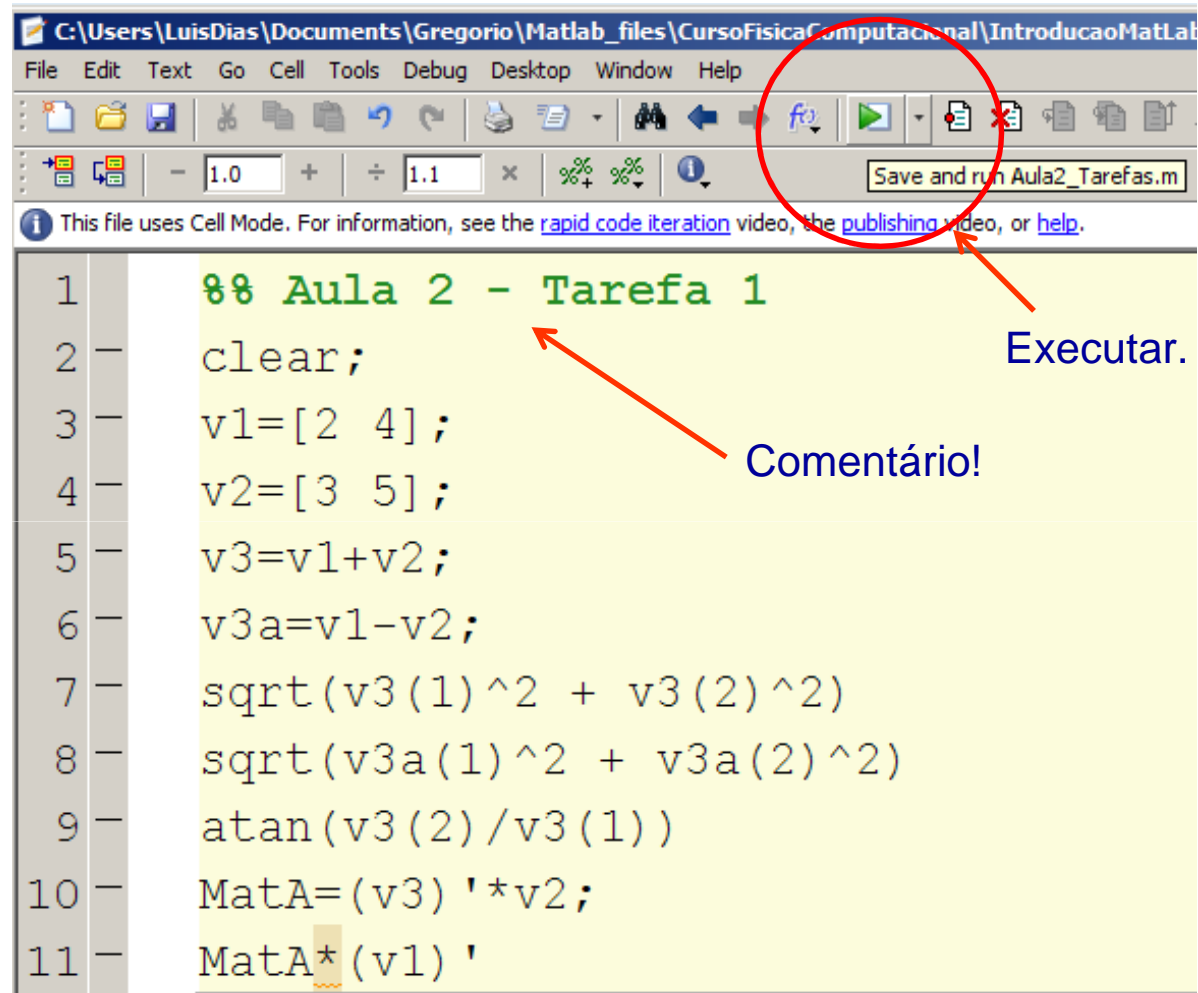
(\*) Aqui representando os vetores como **linhas**, como no MatLab.

- Tarefa 3: *Calcule os valores de  $p(x)=x^4-2x^2+x+1$  em 100 pontos no intervalo entre -1 e +1.*

# Usando “m-files”.

Digitamos os comandos (um por linha) e depois executamos a sequência “de uma vez”.

Caso ocorra algum erro, podemos corrigir diretamente no arquivo e executar novamente.



```
C:\Users\LuisDias\Documents\Gregorio\Matlab_files\CursoFisicaComputacional\IntroducaoMatLab
File Edit Text Go Cell Tools Debug Desktop Window Help
Save and run Aula_2_Tarefas.m
This file uses Cell Mode. For information, see the rapid code iteration video, the publishing video, or help.
1 %% Aula 2 - Tarefa 1
2 clear;
3 v1=[2 4];
4 v2=[3 5];
5 v3=v1+v2;
6 v3a=v1-v2;
7 sqrt(v3(1)^2 + v3(2)^2)
8 sqrt(v3a(1)^2 + v3a(2)^2)
9 atan(v3(2)/v3(1))
10 MatA=(v3)'*v2;
11 MatA*(v1)'
```

# Usando “m-files”.

Variáveis armazenadas

The image displays the MATLAB 7.8.0 (R2009a) interface. The Command Window shows the results of several commands, with colored arrows pointing to specific values. The Workspace window shows variables stored in memory, with a red circle highlighting the 'ans' variable. The Editor window shows the code for 'Aula2\_Tarefas.m', with colored arrows pointing to specific lines of code.

Command Window Output:

```
ans =  
10.2956  
  
ans =  
1.4142  
  
ans =  
1.0637  
  
ans =  
130  
234
```

Workspace Variables:

Name	Value
MatA	[15,25;27,45]
ans	[130;234]
v1	[2,4]
v2	[3,5]
v3	[5,9]
v3a	[-1,-1]

Editor Code (Aula2\_Tarefas.m):

```
1 %% Aula 2 - Tarefa 1  
2 clear;  
3 v1=[2 4];  
4 v2=[3 5];  
5 v3=v1+v2;  
6 v3a=v1-v2;  
7 sqrt(v3(1)^2 + v3(2)^2)  
8 sqrt(v3a(1)^2 + v3a(2)^2)  
9 atan(v3(2)/v3(1))  
10 MatA=(v3)'*v2;  
11 MatA*(v1)'
```

Os resultados dos comandos (aqueles sem “;” no final da linha) aparecem na janela de comando.

# Graficos: o comando “plot”

Command Window

```
>> %% Graficos: usamos vetores!!
```

```
>> vecx=0:pi/10:2*pi;
```

```
>> senox=sin(vecx);
```

```
>> plot(vecx, senox);
```

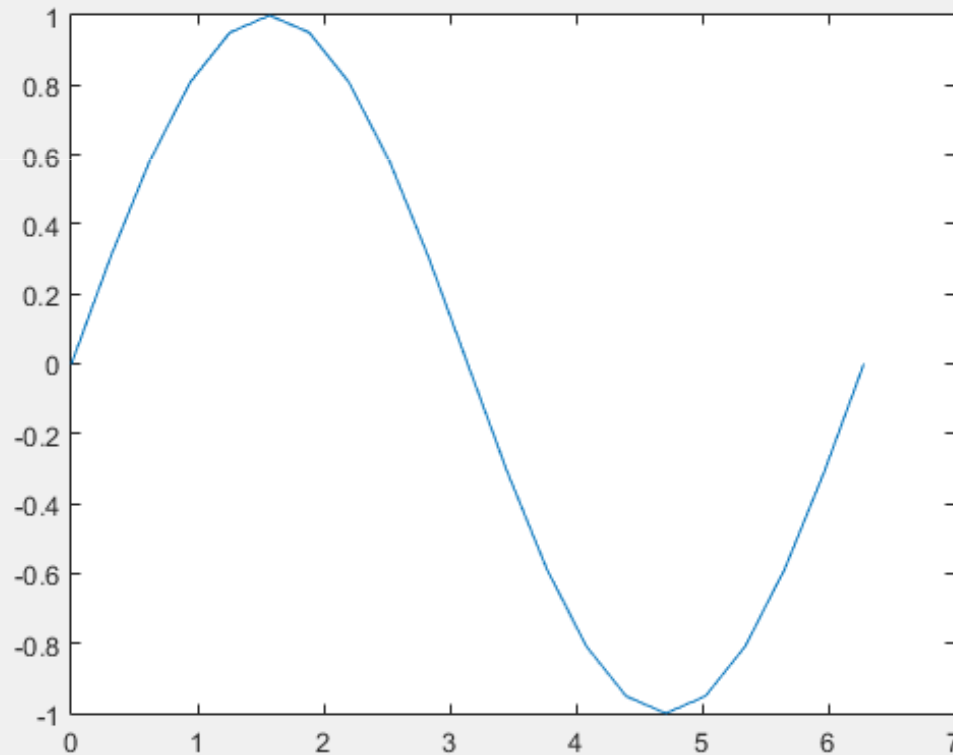
*fx* >>

- Crio dois vetores e usamos `plot(vx,vy)`

- Gráfico aparece em outra janela.

Figure 1

File Edit View Insert Tools Desktop Window Help



# Graficos: opções do comando “plot”

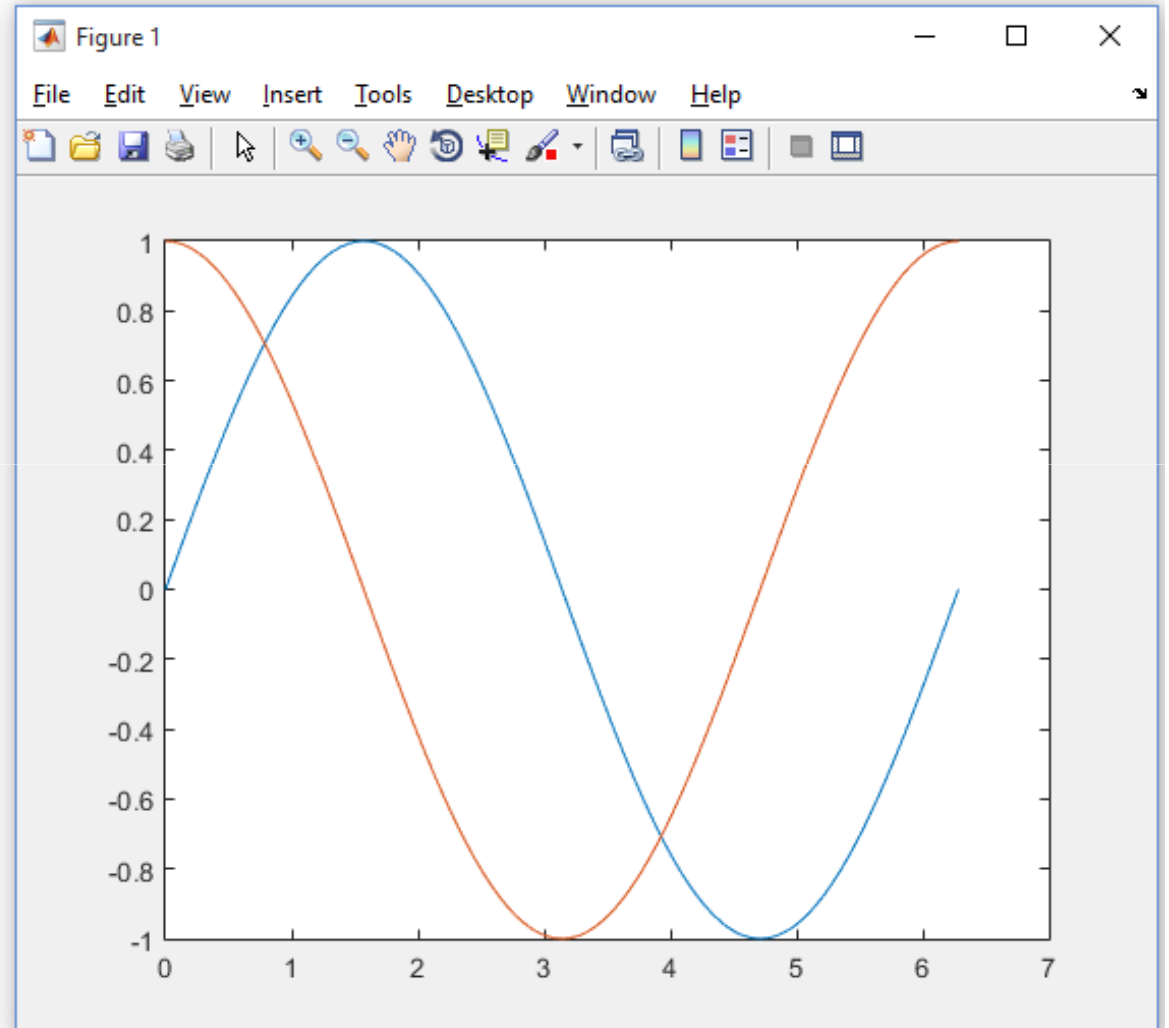
Command Window

```
>> vecx=0:pi/50:2*pi;  
>> senox=sin(vecx);  
>> cosx=cos(vecx);  
>> plot(vecx, senox, vecx, cosx)
```

*fx* >>

- Para plotar duas curvas no mesmo gráfico:

```
plot(x,f1,x,f2)
```



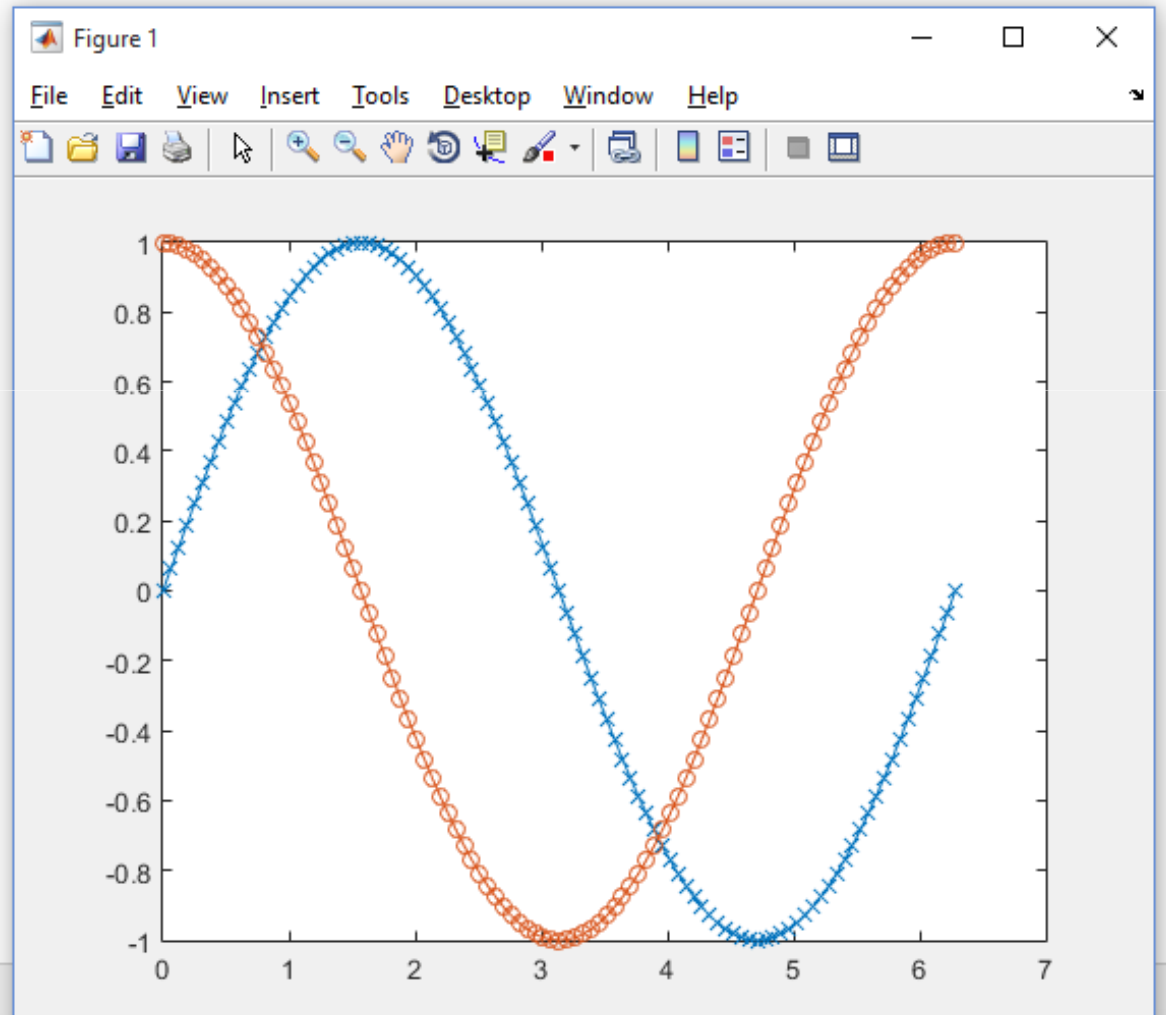
# Graficos: opções do comando “plot”

Command Window

```
>> plot(vecx, senox, '-x', vecx, cosx, '-o')
```

```
fx >>
```

Duas curvas no mesmo gráfico com símbolos diferentes.



# Graficos: opções do comando “plot”

Command Window

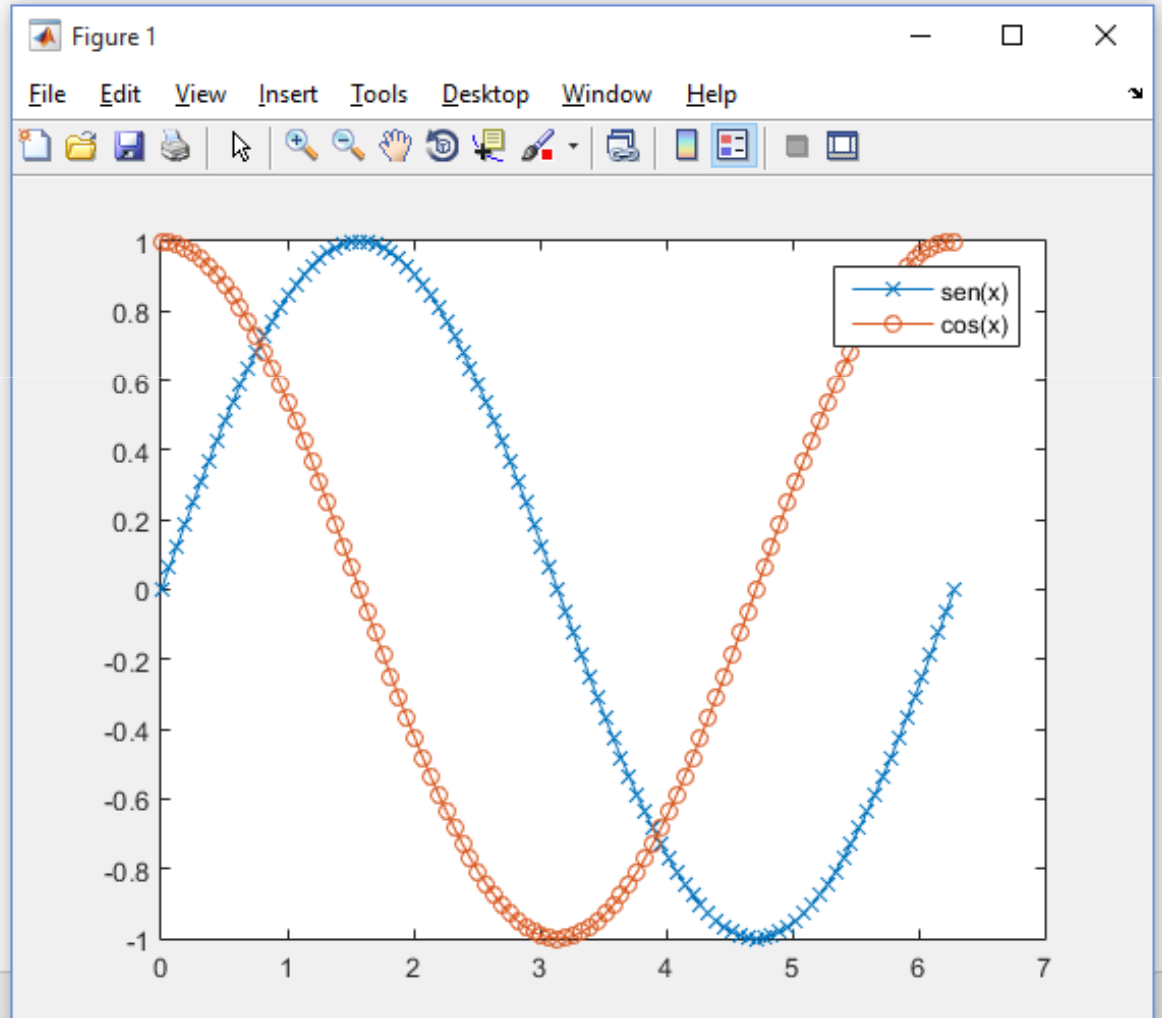
```
>> plot(vecx, senox, '-x', vecx, cosx, '-o')
```

```
>> legend('sen(x)', 'cos(x)')
```

*fx*

```
>>
```

- Adicionando legendas.





# Tarefas Aula 2 (cont):

Lembrando: toda aula haverá tarefas!! (20% da média final!!!)

- Tarefa 3: Dado  $p(x)=x^4-2x^2+x+1$ 
  - *Faça um gráfico de  $p(x)$  vs  $x$  no intervalo  $-2 \leq x \leq 2$ .*
  - Use o “Data Cursor” da janela do gráfico para estimar os valores  $x_i$  para os quais  $p(x_i)=0$ .

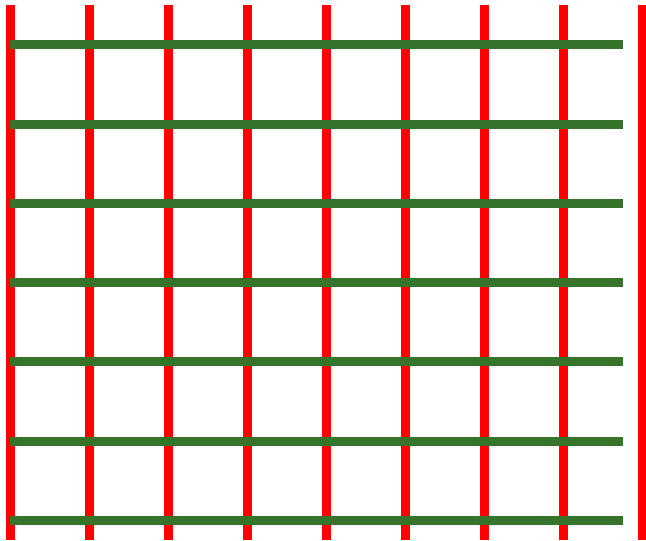
# Gráficos em 3D: comando “meshgrid”

Command Window

```
>> [gridx,gridy]=meshgrid(-2:1/100:2);  
fx >>
```

Primeiro passo: criar matrizes com

- colunas iguais (x) e com
- linhas iguais (y).



Workspace

Name	Value
gridx	<401x40...
gridy	<401x40...

Command History

# Gráficos em 3D: comando “meshgrid”

The screenshot displays the MATLAB interface with three main windows:

- Variable Editor - gridx**: Shows a 12x7 grid of values. The first column contains the value -2, and the subsequent columns contain values from -1.9900 to -1.9000 in increments of 0.0100.
- Workspace**: Lists two variables: `gridx` and `gridy`, both with values of type `<401x401 double>`.
- Command History**: Shows the following commands:

```
z1/z2  
sin(pi/4)  
cos(pi/4)  
...
```

	1	2	3	4	5	6	7
1	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
2	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
3	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
4	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
5	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
6	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
7	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
8	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
9	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
10	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
11	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400
12	-2	-1.9900	-1.9800	-1.9700	-1.9600	-1.9500	-1.9400

`gridx`: colunas iguais (x)

# Gráficos em 3D: comando “meshgrid”

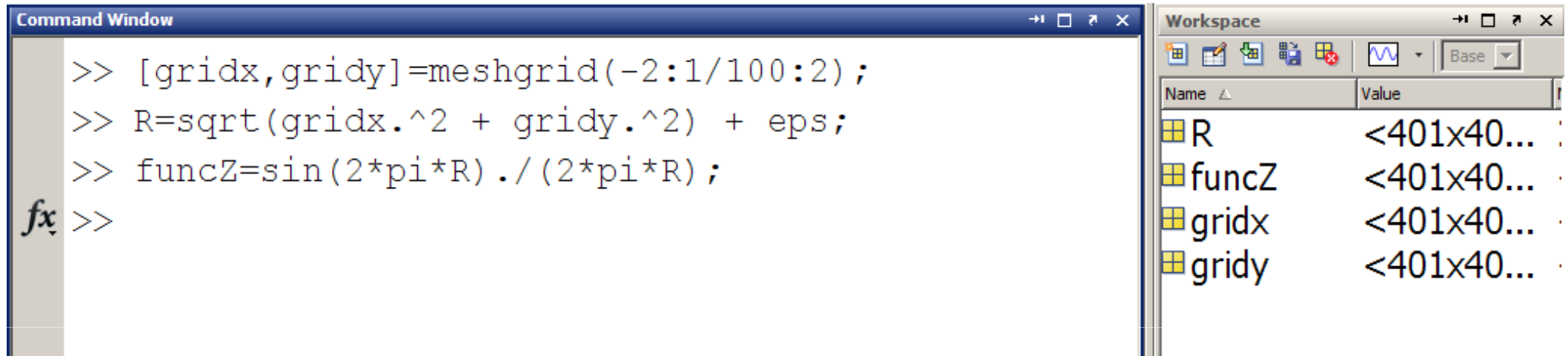
The screenshot displays the MATLAB interface with three windows open:

- Variable Editor - gridy**: Shows a 401x401 double array. The first row contains the value -2 for all columns. Subsequent rows show values increasing from -1.9900 to -1.8900 in increments of 0.0100.
- Workspace**: Lists variables `gridx` and `gridy`, both with dimensions `<401x40...`.
- Command History**: Shows the following commands:

```
z1/z2  
sin(pi/4)  
cos(pi/4)
```

`gridy`: linhas iguais (y).

# Gráficos em 3D: definindo $z=f(x,y)$



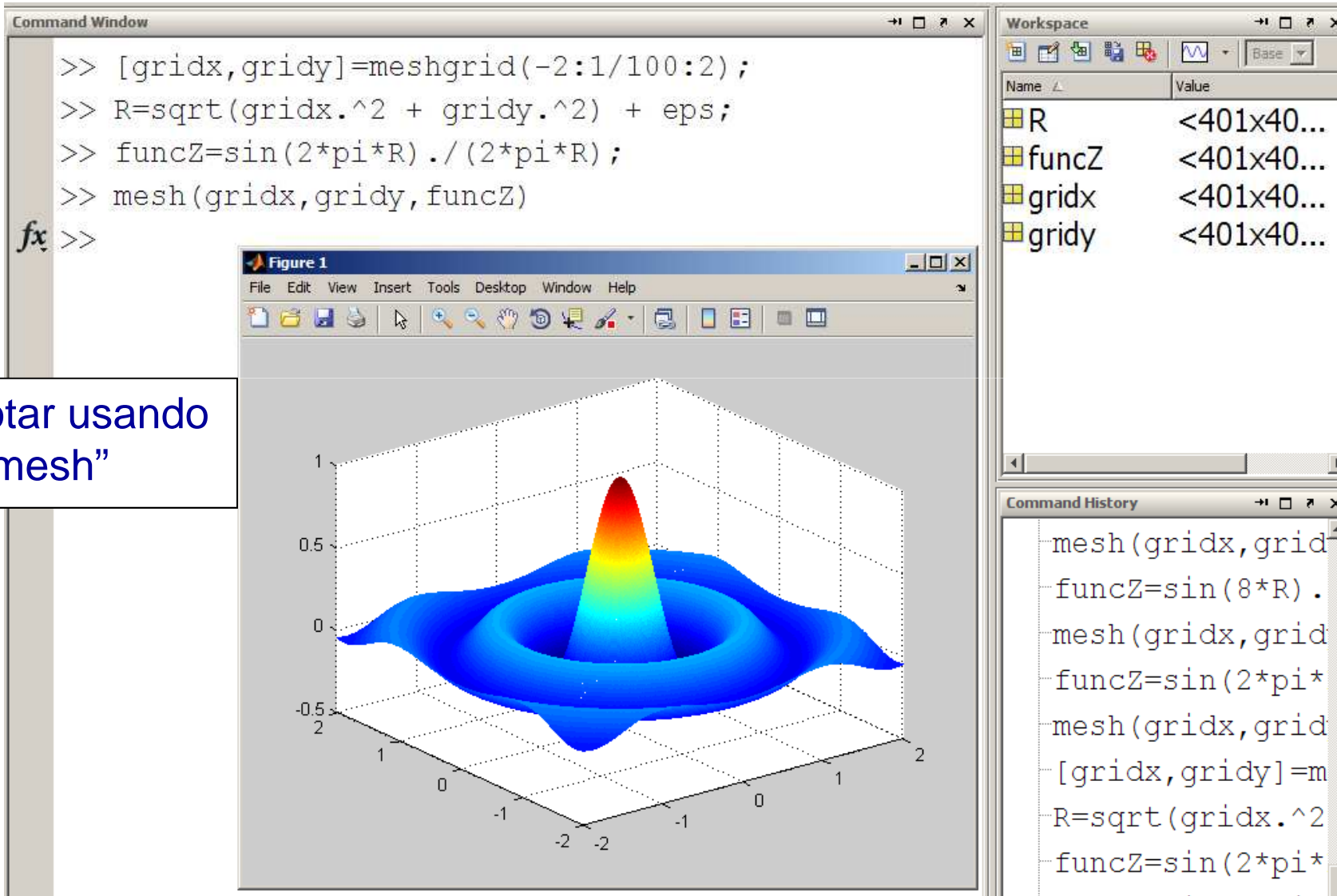
```
>> [gridx,gridy]=meshgrid(-2:1/100:2);  
>> R=sqrt(gridx.^2 + gridy.^2) + eps;  
>> funcZ=sin(2*pi*R) ./ (2*pi*R);  
>>
```

Name	Value
R	<401x401>
funcZ	<401x401>
gridx	<401x401>
gridy	<401x401>

Segundo passo: definir a função  $f(x,y)$ .

- Será uma função que atua nas matrizes *elemento a elemento*:
- Usar o “.” antes dos operadores!  
Exemplo:  $.^*$  ,  $./$  ,  $.^2$  , etc.

# Gráficos em 3D: comando “mesh”



# Tarefas Aula 2 (cont):

Lembrando: toda aula haverá tarefas!! (20% da média final!!!)

- Tarefa 4: *Plote a função*:

$$f(x, y) = e^{-R(x,y)} \text{sen}(\pi x) \text{sen}(\pi y)$$

*no intervalo  $-2 \leq x \leq 2$  ;  $-2 \leq y \leq 2$  usando o comando “mesh”.*