

MATLAB:

Noções Básicas

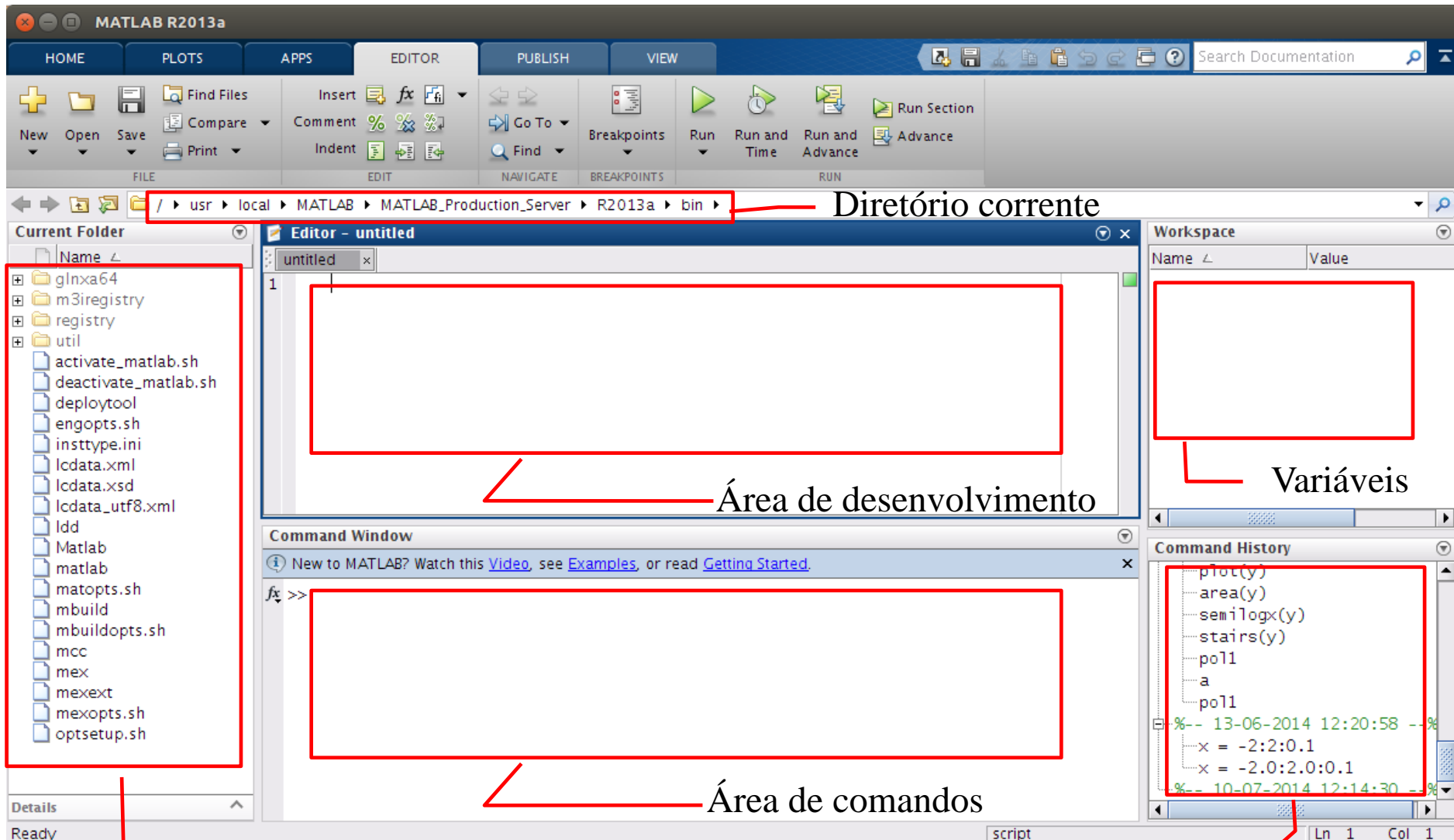
Prof. Dr. Rogério Galante Negri



- **MA**trix **LAB**oratory é um software otimizado para cálculos
 - Criado no fim dos anos 1970 por Cleve Moler (Univ. Novo México)
 - Jack Little (Univ. Stanford), reconhecendo o seu potencial comercial do MATLAB, juntou-se a Moler e Steve Bangert
 - Em 1984, Moler, Bangert e Litte reescreveram o MATLAB em C
 - As bibliotecas reescritas ficaram conhecidas como LAPACK
- MATLAB implementa a linguagem de programação MATLAB

Vantagens	Desvantagens
Facilidade de uso	Linguagem interpretada: execução mais lenta
Independência de plataforma	
Funções predefinidas	
Desenhos independente do dispositivo	Custo de aquisição é de 5 a 10 vezes maior que um compilador convencional C ou Fortran (mas o custo é compensado pelo tempo de programação)
Interface gráfica de usuário	
Compilação independente de dispositivo	

Ambiente



Conteúdo do diretório corrente

Histórico

Ambiente + Exemplo

The image displays the MATLAB R2013a software interface. The top menu bar includes HOME, PLOTS, APPS, EDITOR, PUBLISH, and VIEW. Below the menu bar is a toolbar with icons for file operations (New, Open, Save, Print), editing (Insert, Comment, Indent), navigation (Go To, Find), breakpoints, and running (Run, Run and Time, Run and Advance, Run Section). The main workspace is divided into several panes:

- Current Folder:** Shows the files 'areaCirc.m' and 'pol1.m'.
- Editor:** Displays the script 'areaCirc.m' with the following code:

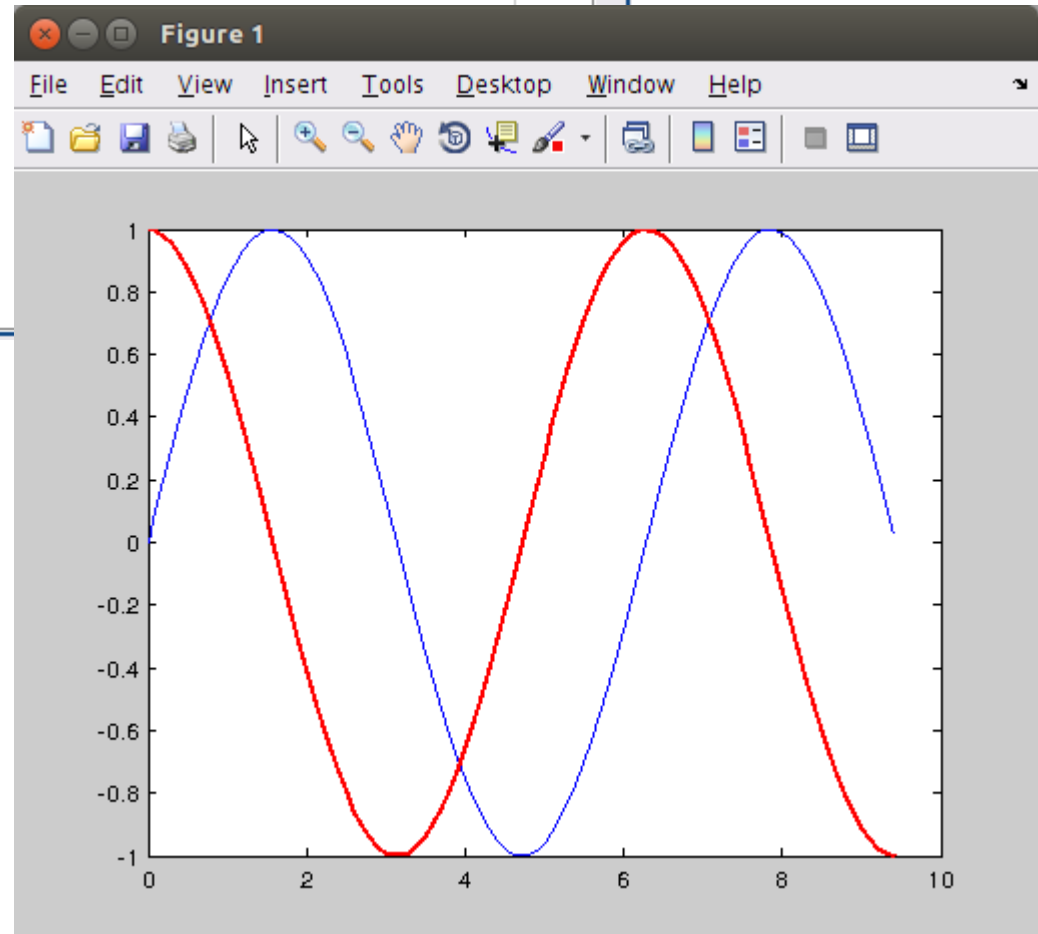
```
1 %Calcula da área de um círculo de raio 3.5
2 r = 3.5;
3 area = pi*r^2;
4 str = ['Area do círculo de raio ', num2str(r), ' é de ', num2str(area)];
5 disp(str);
6
```
- Workspace:** Shows the variables 'area' (38.4845), 'r' (3.5000), and 'str' ('Area do círculo d..').
- Command Window:** Shows the execution of the script:

```
>> areaCirc
Area do círculo de raio 3.5 é de 38.4845
fx >>
```
- Command History:** Lists the commands executed, including 'area(y)', 'semilogx(y)', 'stairs(y)', 'pol1', 'a', 'pol1', and 'areaCirc'.

The status bar at the bottom indicates 'areaCirc.m (MATLAB Script)'.

Outro exemplo

```
Editor - /home/rogerio/Matlab.Workspace/grafSenoCosseno.m
areaCirc.m x grafSenoCosseno.m x
1 %Gráfico de seno e cosseno de 0 até 3pi
2 x = 0:0.1:3*pi;
3 gseno = sin(x);
4 gcosseno = cos(x);
5
6 graf1 = plot(x,gseno);|
7
8 hold on
9
10 graf2 = plot(x,gcosseno);
11 set(graf2,'Color','red','LineWidth',2)
```

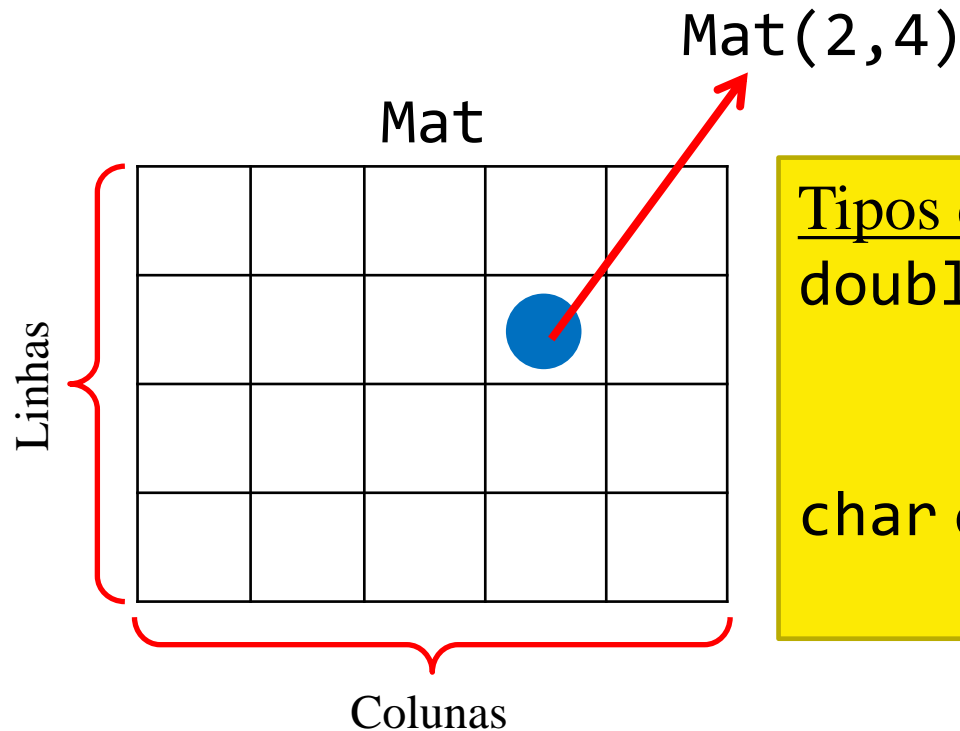


Conceitos

- Variáveis (matrizes), declarações e inicialização
 - Operadores Relacionais e Lógicos
 - Estruturas condicionais (`if`)
 - Estruturas de repetição (`while` e `for`)
 - Funções de entrada/saída
-

Variáveis

- Em MATLAB, a unidade fundamental é a **matriz** (tudo é matriz)
- Os elementos da matriz são acessados a partir do nome da matriz e de seus índices (como em qualquer linguagem)



Tipos comuns em MATLAB:

double escalares/matrizes de 64-*bits*
valores \pm entre 10^{-308} a 10^{308}
precisão de 15 a 16 dígitos
char escalares/matrizes de 16-*bits*
armazenam caracteres

A linguagem C é “fortemente tipada” (o tipo das var. deve ser definida)
MATLAB é “fracamente tipada”

Variáveis

- Exemplos

```
>> v = 10;  
>> x = 5*v;  
>> info = 'Exemplo de cadeia de caracteres';
```

; ao final da linha é para evitar eco no console e identificar final comando

Comentário

- Inicializações:

```
>> vec = [1 2 3 4]; %define o vetor (1,2,3,4)  
>> mat = [1, 2; 4, 5]; %define a matriz  $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$   
>> matNu(2,3) = 0; %define uma matriz  $2 \times 3$  nula
```


Atalhos de inicialização

- Operador “dois-pontos”

`primeiro:incremento:ultimo`

```
>> x = 1 : 2 : 10; %gera x = [1, 3, 5, 7, 9]
```

```
>> ang = (0.1 : 0.1 : 1.0) * pi; %gera valores de 0.1 a 1.0,  
incrementados em 0.1,multiplicados por  $\pi$ 
```

- Transposição

```
>> m = [1, 2];
```

```
>> n = m'; %gera  $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$ 
```

- Combinação

```
>> p = [m' m']; %gera  $\begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}$ 
```



Funções úteis de inicialização

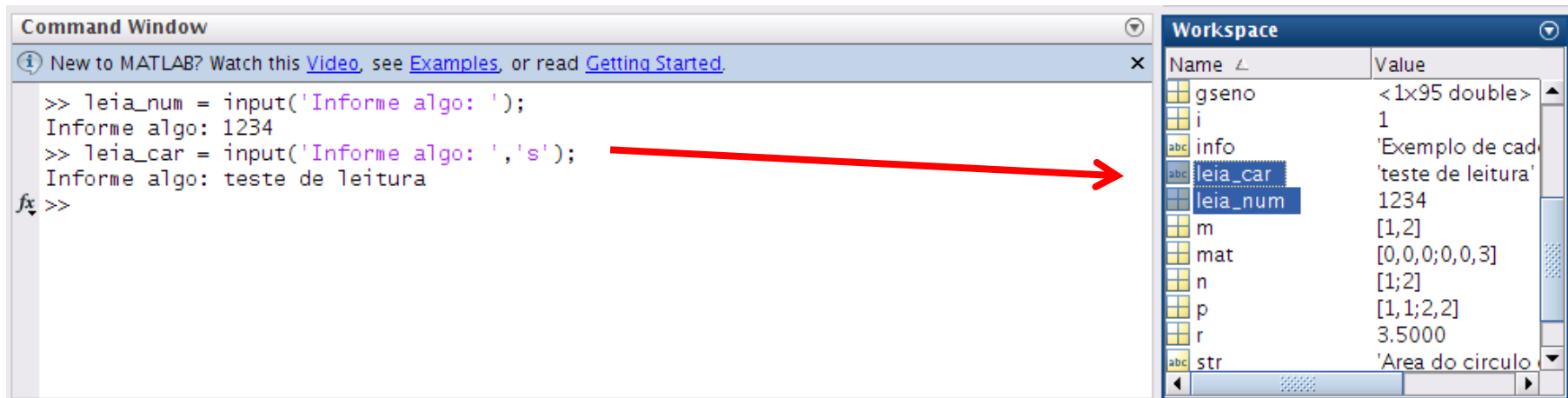
Função	Ação
<code>zeros(n);</code>	Gera uma matriz $n \times n$ de 0
<code>zeros(n,m);</code>	Gera uma matriz $n \times m$ de 0
<code>zeros(size(var));</code>	Gera uma matriz de 0 do mesmo tamanho de <code>var</code>
<code>ones(n);</code>	Gera uma matriz $n \times n$ de 1
<code>ones(n,m);</code>	Gera uma matriz $n \times m$ de 1
<code>ones(size(var));</code>	Gera uma matriz de 1 do mesmo tamanho de <code>var</code>
<code>eye(n)</code>	Gera uma matriz identidade $n \times n$
<code>eye(n,m)</code>	Gera uma “matriz identidade” $n \times m$
<code>length(var)</code>	Retorna o comprimento de um vetor <code>var</code> ou a dimensão maior de uma matriz <code>var</code>
<code>size(var)</code>	Retorna o numero de linhas e de colunas de <code>var</code>

Entrada pelo teclado – input

- O comando `input` permite que uma informação seja inserida via teclado

```
>> leia_num = input('Informe algo: ');
```

```
>> leia_car = input('Informe algo: ','s');
```

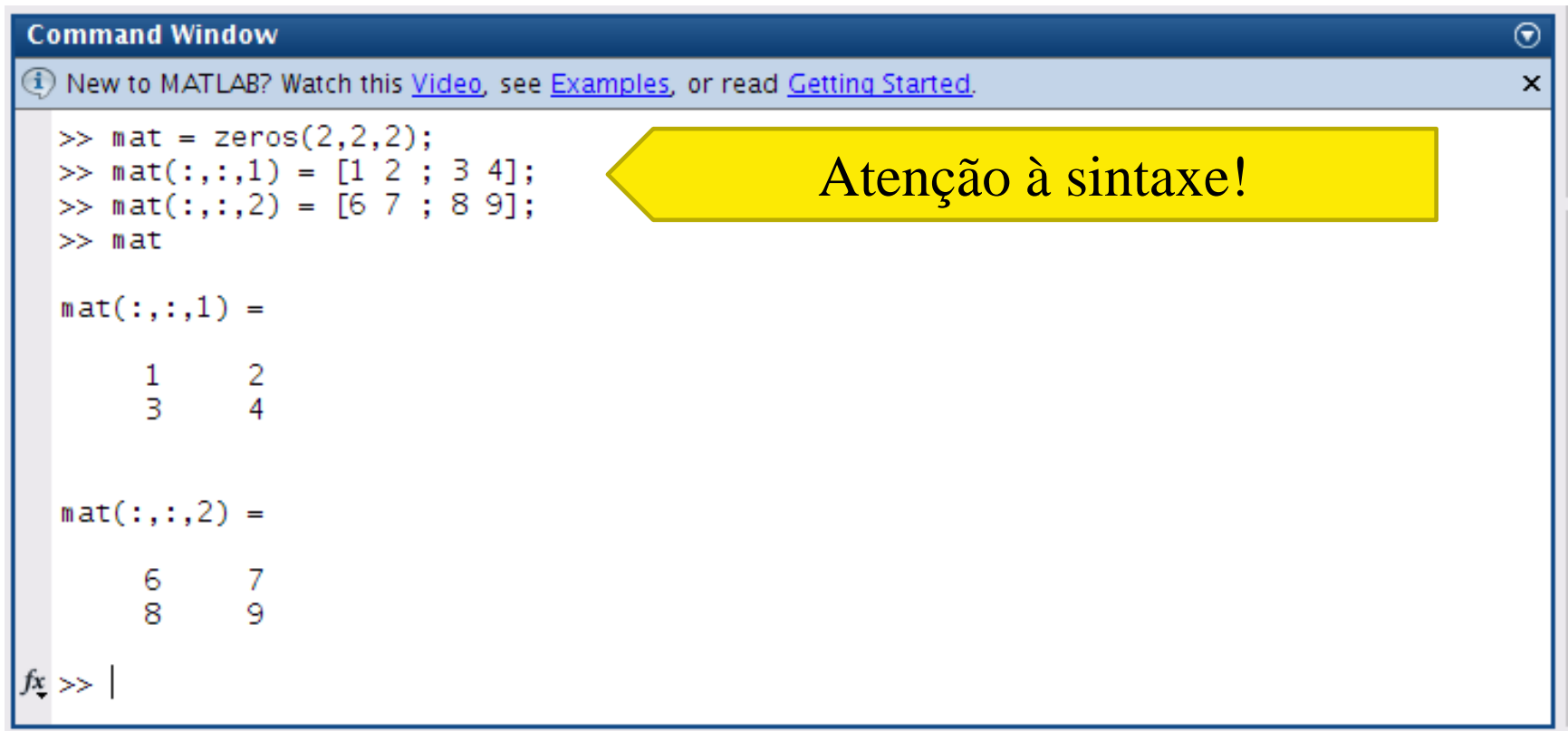


The screenshot displays the MATLAB Command Window and Workspace. The Command Window shows the execution of two `input` commands. The first command prompts for 'Informe algo: ' and the user enters '1234'. The second command prompts for 'Informe algo: ' with a string format ('s') and the user enters 'teste de leitura'. The Workspace window shows the resulting variables: `leia_num` is a double scalar with value 1234, and `leia_car` is a string with value 'teste de leitura'. A red arrow points from the second command in the Command Window to the `leia_car` variable in the Workspace.

Name	Value
gseno	<1x95 double>
i	1
info	'Exemplo de cad'
leia_car	'teste de leitura'
leia_num	1234
m	[1,2]
mat	[0,0,0;0,0,3]
n	[1;2]
p	[1,1;2,2]
r	3.5000
str	'Area do circulo'

Matrizes multidimensionais

- Ao contrário do acostumado, podemos definir matrizes com mais de duas dimensões



The screenshot shows the MATLAB Command Window with the following code and output:

```
>> mat = zeros(2,2,2);  
>> mat(:,:,1) = [1 2 ; 3 4];  
>> mat(:,:,2) = [6 7 ; 8 9];  
>> mat
```

The output displays the contents of the 3D matrix `mat` for each of its two slices:

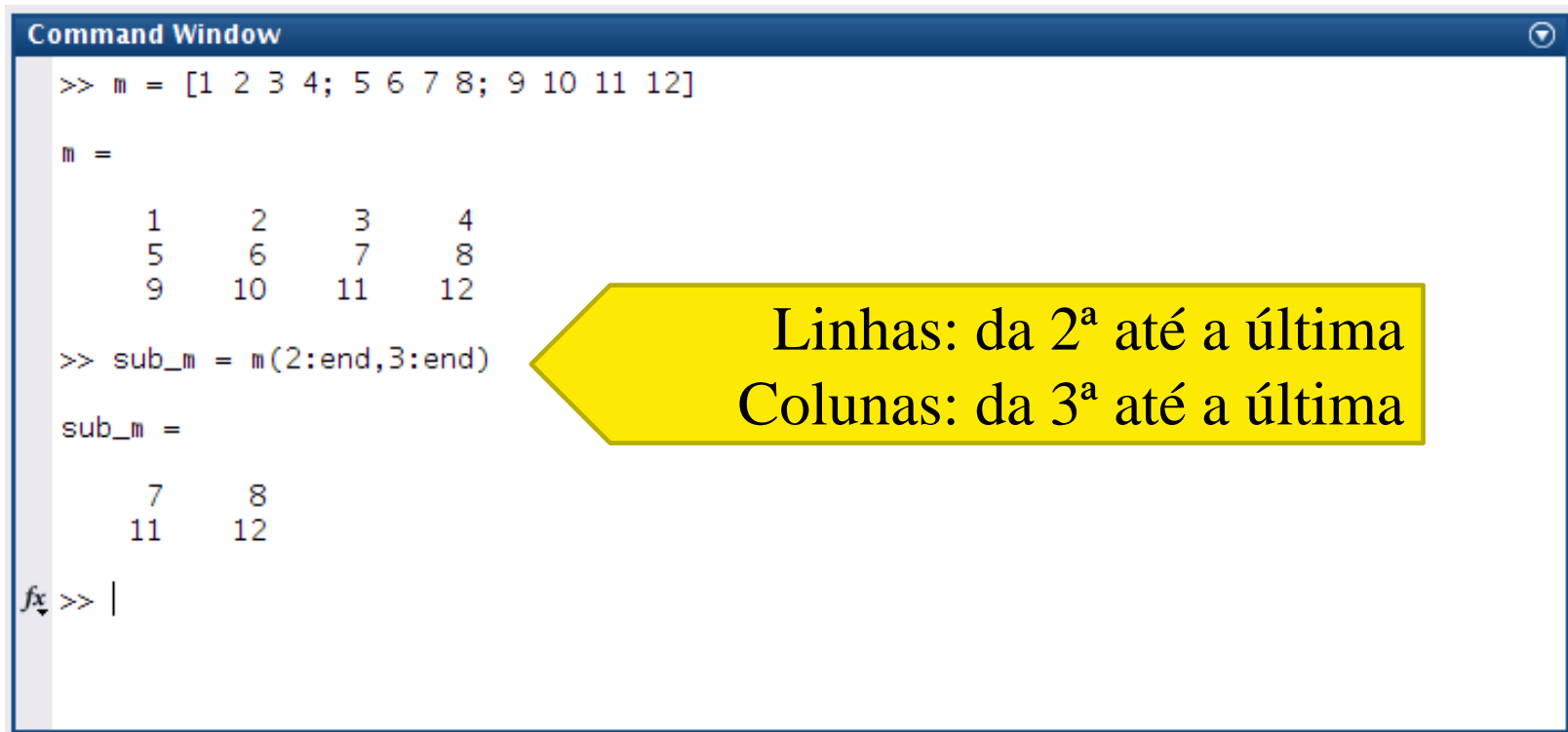
```
mat(:,:,1) =  
  
     1     2  
     3     4  
  
mat(:,:,2) =  
  
     6     7  
     8     9
```

A yellow callout box with a left-pointing arrow contains the text: **Atenção à sintaxe!**

- As funções `zeros`, `ones`, `eye` podem ser generalizadas para quantas dimensões desejadas

Função end

- A função end é muito útil na manipulação de matizes
- Retorna o maior índice de uma matriz, segundo a dimensão observada
- Desempenha ainda utilidade na definição de submatrizes



```
Command Window
>> m = [1 2 3 4; 5 6 7 8; 9 10 11 12]

m =

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

>> sub_m = m(2:end,3:end)

sub_m =

     7     8
    11    12

fx >> |
```

Linhas: da 2ª até a última
Colunas: da 3ª até a última

Valores especiais

Função	Ação
<code>pi</code>	Armazena π com 15 dígitos significativos
<code>i</code> , <code>j</code>	Contém o valor $\sqrt{-1}$
<code>Inf</code>	Infinito de máquina (e.g., resultado de divisão por 0)
<code>NaN</code>	“Not a Number” (e.g., resultado de op. indefindas)
<code>clock</code>	Vetor [ano, mês, dia, hora, minuto, segundo]
<code>date</code>	Cadeia de caracteres da data corrente em Dia-Mês-Ano
<code>eps</code>	Menor diferença possível entre dois números no comp.
<code>ans</code>	Variável especial que armazena o resultado de uma expressão, caso não seja atribuída a outra

Função disp

- Uma forma de exibir dados é a função `disp`
 - Aceita como argumento um vetor/matriz
 - Como resultado a exibição da informação na janela de comandos
- O argumento deve ser do tipo `char`
- Usualmente está associado a `num2str` e `int2str`
 - `num2str` converte numero em cadeia de caracteres
 - `int2str` converte inteiro em cadeia de caracteres

```
>>str = [' Pi é igual a ' ,num2str(pi)];  
>>disp(str);
```

Função fprintf

- Exibe um ou mais valores junto com textos relacionados
- Usa códigos de formatação da linguagem C
- Sua sintaxe é:

`fprintf(formato, dados);`

- Exemplo: `>>fprintf('O valor de Pi é %5.4f \n', pi);`

Código	Ação
%d	Exibe como valor inteiro
%e	Exibe na forma exponencial
%f	Exibe em ponto flutuante
%g	Exibe em exponencial ou ponto flutuante (o mais curto)
\n	Muda de linha

Arquivos – save

- O comando `save` grava dados da área de variáveis em um arquivo em disco

- A sintaxe é:

```
>>save nome_arquivo var1 var2 ... varN
```

- O arquivo salvo será escrito no formato MAT
 - Este arquivo contém as variáveis salvas
 - Pode ser lido apenas pelo MATLAB
 - Para acessar os dados por outros programas, é usado `-ascii`

```
>>save nome_arquivo var1 var2 ... varN -ascii
```
-

Arquivos – load

- Em oposição ao `save`, existe o comando `load`
- A sintaxe é:

```
>>load nome_arquivo
```

- Como resultado, as variáveis armazenadas em `nome_arquivo` são importadas para a área de variáveis do MATLAB
-

Operações com escalares

Operação	Rep. algébrica	Rep. MATLAB
Soma	$a + b$	<code>a + b</code>
Subtração	$a - b$	<code>a - b</code>
Multiplicação	$a \times b$	<code>a * b</code>
Divisão	$\frac{a}{b}$	<code>a / b</code>
Exponenciação	a^b	<code>a ^ b</code>

- A atribuição possui a seguinte forma geral:
`>>nome_variavel = expressao`
 - Ordem de prioridade convencional
 - Alteração da ordem de prioridade via uso de parênteses
-

Operações com matrizes

- MATLAB suporte:
 - Operações estruturais – “elemento a elemento”
 - Operações matriciais – segue a Álgebra Linear

Operação	MATLAB	Descrição
Soma estrutural	$a+b$	Soma duas matrizes de mesma dimensão
Subtração estrutural	$a-b$	Subtrai duas matrizes de mesma dimensão
Multiplicação estrutural	$a.*b$	Multiplica elementos de a e b dois a dois
Multiplicação matricial	$a*b$	Multiplica as matrizes a por b
Divisão à direita estrutural	$a./b$	Divide elementos de a por b dois a dois
Divisão à esquerda estrutural	$a.\backslash b$	Divide elementos de b por a dois a dois
Divisão matricial à direita	a/b	Divisão matricial equiv. $a*\text{inv}(b)$
Divisão matricial à esquerda	$a\backslash b$	Divisão matricial equiv. $\text{inv}(a)*b$
Exponenciação estrutural	$a.^b$	Exponenciação de a e b dois a dois

Funções	MATLAB	Descrição
Matemáticas	abs(x)	Módulo de x
	cos(x)	Cosseno de x, com x em radianos
	sin(x)	Seno de x, com x em radianos
	tan(x)	Tangente de x, com x em radianos
	log(x)	Logaritmo natural de x
	exp(x)	Exponencial natural de x
	[val,ind]=max(x)	Valor máximo e posição (vetorial) de x
	[val,ind]=min(x)	Valor mínimo e posição (vetorial) de x
	mod(x,y)	Resto da divisão de x por y
	sqrt(x)	Raiz quadrada de x
Aprox.	ceil(x)	Teto de x
	floor(x)	Piso de x
	round(x)	Arredondamento de x
Conversão	char(x)	Converte o número x em cadeia de caractere ($x \leq 127$)
	double(x)	Converte a cadeia de caractere x em uma matriz
	int2str(x)	Converte o número inteiro x em cadeia de caractere
	num2str(x)	Converte o número decimal x em cadeia de caractere
	str2num(x)	Converte a cadeia de caractere x em um número

Gráficos planos – Exemplo

```
x = 0:0.1:10;  
y = x.^2 + 2.*x;  
z = x.^3 + 1;
```

Atenção aos comandos:
grid on/off - xlabel
title - subplot - legend



```
subplot(2,2,1);  
plot(x,y,'r--',x,z,'b-'); xlabel('X'); ylabel('Y'); title('Linear');  
grid on; legend('f(x)=x^2+2x','g(x)= x^3+1',2);
```

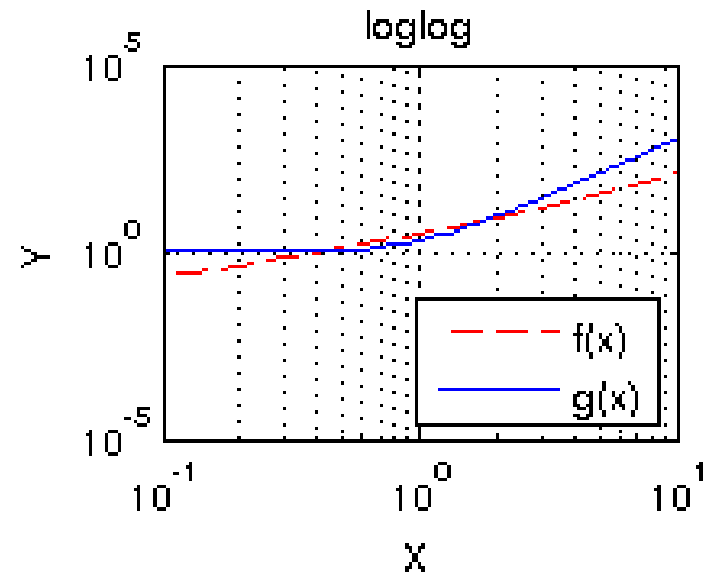
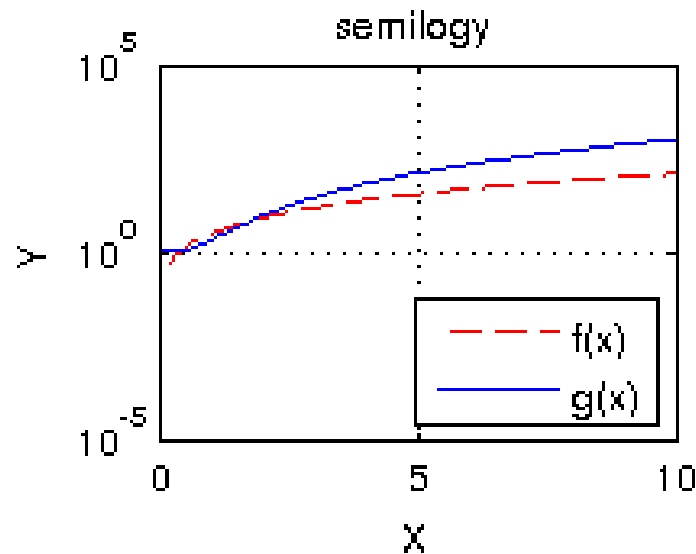
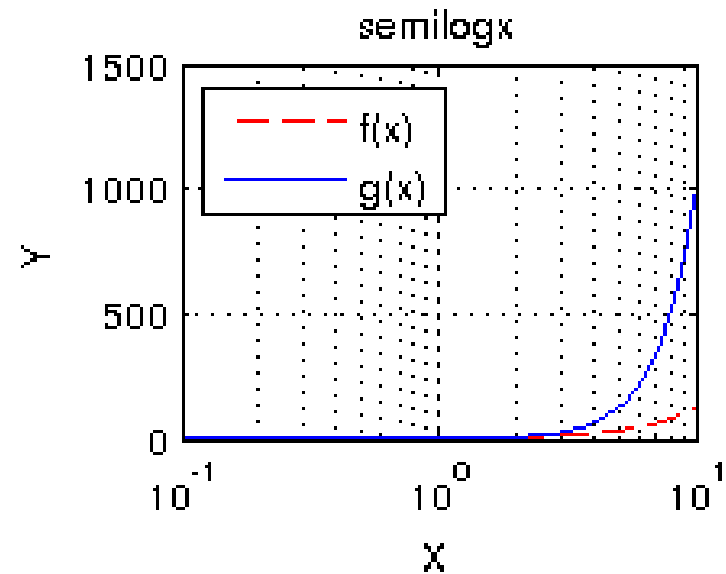
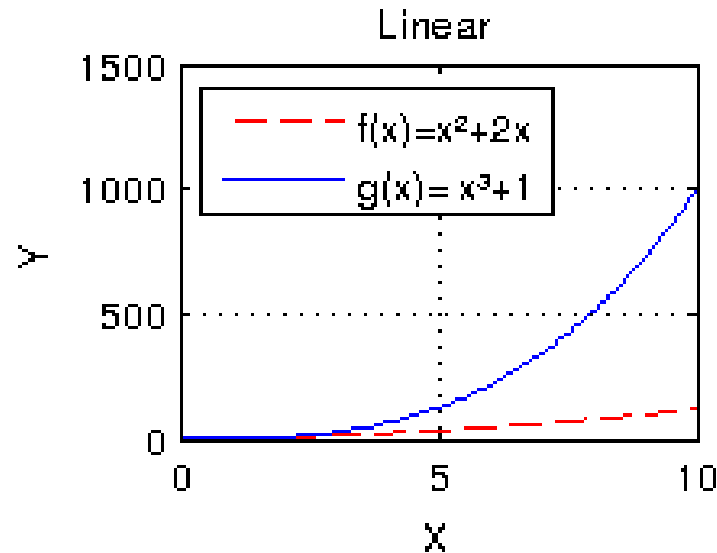
← Posição da legenda

```
subplot(2,2,2);  
semilogx(x,y,'r--',x,z,'b-'); xlabel('X'); ylabel('Y'); title('semilogx');  
grid on; legend('f(x)','g(x)',2);
```

```
subplot(2,2,3);  
semilogy(x,y,'r--',x,z,'b-'); xlabel('X'); ylabel('Y'); title('semilogy');  
grid on; legend('f(x)','g(x)',4);
```

```
subplot(2,2,4);  
loglog(x,y,'r--',x,z,'b-'); xlabel('X'); ylabel('Y'); title('loglog');  
grid on; legend('f(x)','g(x)',4);
```

Gráficos planos – Resultado



Styling!

Cor		Marcador				Linha	
y	Amarelo	.	Ponto	<	Δ p/ esq.	-	Sólido
m	Magenta	o	Circulo	>	Δ p/ dir	:	Pontilhado
c	Ciano	x	X	v	Δ p/ baixo	-.	Ponto-traço
r	Vermelho	+	Mais	^	Δ p/ cima	--	Tracejado
g	Verde	*	Asterisco	p	Pentágono		
b	Azul	s	Quadrado	h	Hexágono		
w	Branco	D	Losango				
k	Preto						

Códigos de posição da legenda

Valor	Significado	Valor	Significado
0	Automático	3	Inf. esquerdo
1	Sup. direito	4	Inf. direito
2	Sup. esquerdo	-1	À direita do desenho

Operadores relacionais

- Valores lógicos em MATLAB (assim como em C):
1 – Verdadeiro 2 – Falso
- Operadores relacionais atuam sobre dois operandos (numéricos ou caracteres) e produzem um valor lógico

Operador	Operação
==	Igual a
~=	Diferente de
>	Maior que
>=	Maior ou igual a
<	Menor que
<=	Menor ou igual a

- Operadores relacionais podem ser usados para compara um escalar com uma matriz
- O resultado é uma matriz de mesma dimensão com o resultado lógicos das comparações
- É possível compara ainda duas matrizes de mesma dimensão (“elemento a elemento”)

Operadores lógicos

- Operam sobre valores/expressões lógicas e retornam valores lógicos

Operador	Operação
&&	“E” lógico (conjunção)
	“OU” lógico (disjunção)
xor	“OU Ex.” (disjunção exclusiva)
~	“NÃO” (negação)

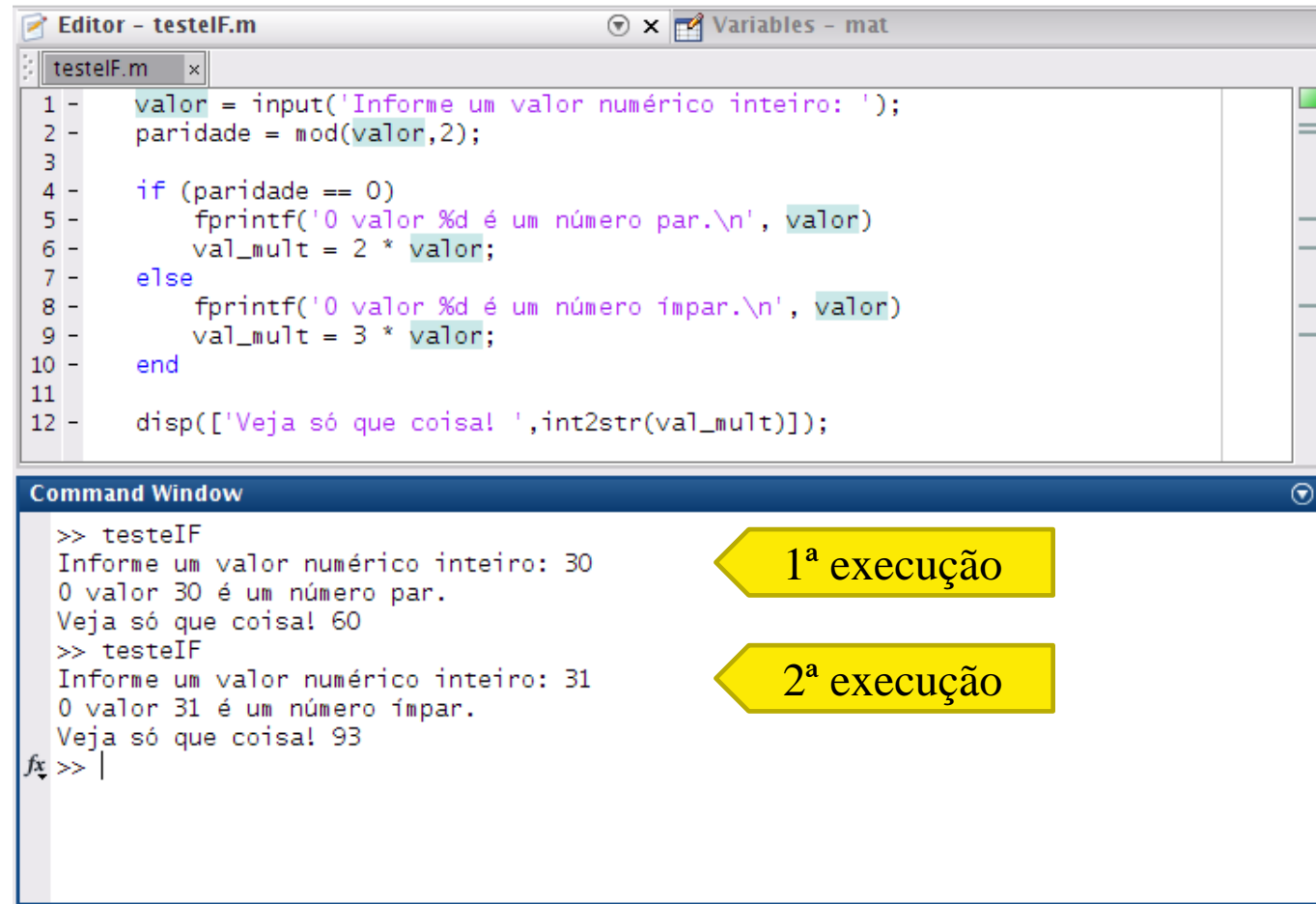
Entradas		“E”	“OU”	“OU Ex.”	“NÃO”
a	b	a&&b	a b	xor(a,b)	~a
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Estrutura condicional – if

- Permite o desvio de fluxo de execução

- Sintaxe:

```
if expr1
    Bloco1
else
    Bloco2
end
```



The screenshot displays the MATLAB environment. The Editor window shows a script named `testeIF.m` with the following code:

```
1 - valor = input('Informe um valor numérico inteiro: ');
2 - paridade = mod(valor,2);
3
4 - if (paridade == 0)
5 -     fprintf('0 valor %d é um número par.\n', valor)
6 -     val_mult = 2 * valor;
7 - else
8 -     fprintf('0 valor %d é um número ímpar.\n', valor)
9 -     val_mult = 3 * valor;
10 - end
11
12 - disp(['Veja só que coisa! ',int2str(val_mult)]);
```

The Command Window shows the execution of the script twice:

```
>> testeIF
Informe um valor numérico inteiro: 30
0 valor 30 é um número par.
Veja só que coisa! 60
>> testeIF
Informe um valor numérico inteiro: 31
0 valor 31 é um número ímpar.
Veja só que coisa! 93
fx >> |
```

Yellow callout boxes highlight the execution results:

- 1ª execução (for input 30, output 60)
- 2ª execução (for input 31, output 93)

Estrutura condicional – switch

- Permite escolher um fluxo de execução
- Sintaxe:

```
switch (exp_switch)
case (exp_case1)
    Bloco1
case (exp_case2)
    Bloco2
:
otherwise
    BlocoContrario
end
```

```
valor = input('Informe um valor de
1 a 10: ');
switch valor
case {1, 3, 5, 7, 9}
    disp('É ímpar')
case {2, 4, 6, 8, 10}
    disp('É par')
otherwise
    disp('Inválido')
end
```

```
>> testeSwitch
Informe um valor de 1 a 10: 4
É par
>> testeSwitch
Informe um valor de 1 a 10: 19
Inválido
```

Estrutura de repetição – while

- Repete indefinidamente um bloco de códigos, enquanto a condição é satisfeita

%Série de Fibonacci até número definido

```
num = input('Informe um limite superior (>1): ');  
n1 = 1; n2 = 1;
```

```
fprintf('%d - %d',n1,n2);  
while (n1+n2) <= num  
    aux = n1+n2;  
    n1 = n2;    n2 = aux;  
    fprintf(' - %d', aux);  
end
```

- Sintaxe:

```
while expr  
    Bloco  
end
```

```
>>Informe um limite superior (>1): 56  
1 - 1 - 2 - 3 - 5 - 8 - 13 - 21 - 34 - 55>>
```

Estrutura de repetição – for

- Repete um bloco de códigos durante um número específico de vezes

```
disp('Soma dos X primeiros multiplos menores que Y');
Y = input('Informe o multiplo: ');
X = input('Informe o valor superior: ');
soma = 0;
for ind = 0:Y:X
    soma = soma + ind;
    fprintf('+ %d ',ind);
end
fprintf('= %d\n',soma);
```

- Sintaxe:

```
for indice = expr
    Bloco
end
```

```
Soma dos X primeiros multiplos menores que Y
Informe o multiplo: 3
Informe o valor superior: 17
+ 0 + 3 + 6 + 9 + 12 + 15 = 45
```

Funções

- O uso de funções permite:
 - Fragmentar o problema em sub-tarefas
 - Podem ser testadas independentemente
 - Código reutilizável
 - Uma mesma sub-tarefa pode ser usada em diferentes partes
 - Isolamento de efeitos colaterais indesejados
 - Erros dentro das funções devem afetar somente as mesmas
 - No MATLAB, cada função é salva em um arquivo separado, cujo nome é igual ao da função.
-

Funções - Exemplo

The image shows a MATLAB environment with two windows: the Editor and the Command Window.

Editor Window: The title bar shows the file path `/home/rogerio/Matlab.Workspace/dist2.m`. The code defines a function `dist2` that calculates the 2D distance between two vectors. The code is as follows:

```
function distance = dist2(v1,v2)
%DIST2 Calcula a distância entre dois vetores 2d
%Pois é, calcula mesmo...
%
%Calling sequence:
% resultado = dist2(Vec1,Vec2)
% onde VecN = [xN , yN]
%
%Destes comentários em diante não entra na documentação...
%Perecebeu?

distance = sqrt( (v1(1) - v2(1)).^2 + (v1(2) - v2(2)).^2 );
end
```

Command Window: The window shows the execution of the function and a help query.

```
>> val = dist2([0,0],[1,2])

val =

    2.2361

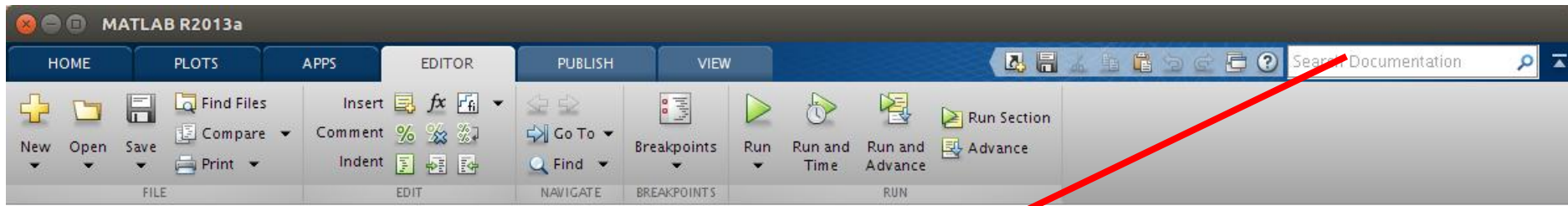
>> help dist2
dist2 Calcula a distância entre dois vetores 2d
Pois é, calcula mesmo...

Calling sequence:
resultado = dist2(Vec1,Vec2)
onde VecN = [xN , yN]
```

Annotations:

- A yellow label **Documentação** with a red bracket points to the comment lines in the function definition.
- A yellow label **Nome da função** points to the function name `dist2` in the function signature.
- A yellow label **Variável de retorno** points to the variable `distance` in the function signature.
- A red arrow points from the text **Consulta sobre a documentação da função** to the `help dist2` command in the Command Window.

Sugestão interessante!



- Dúvida?
- Antes de qualquer coisa, procure na DOCUMENTAÇÃO

Bibliografia da aula

- CHAPMAN, S. J. **Programação em MATLAB para engenheiros**. 2ª Ed. Cengage, 2010.

