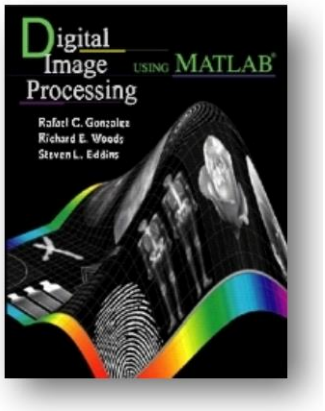


Introdução ao Matlab

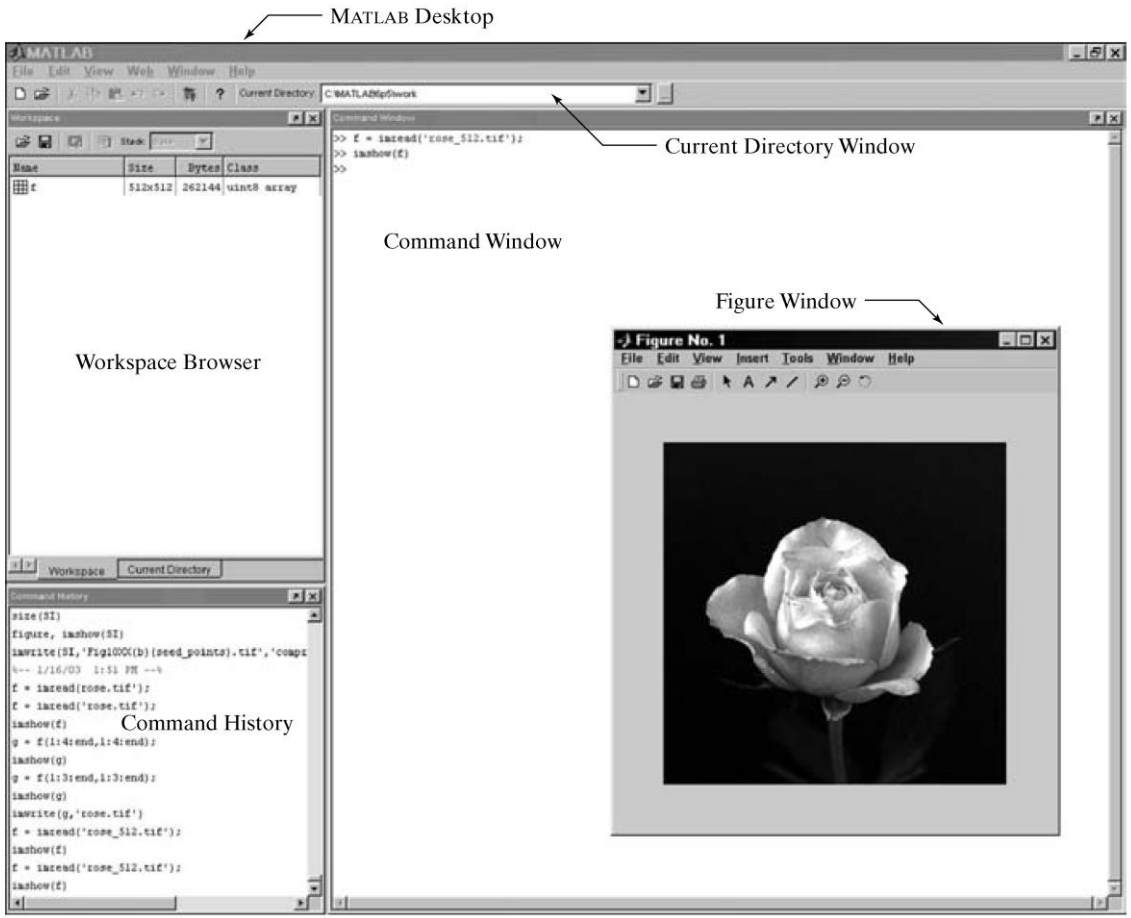
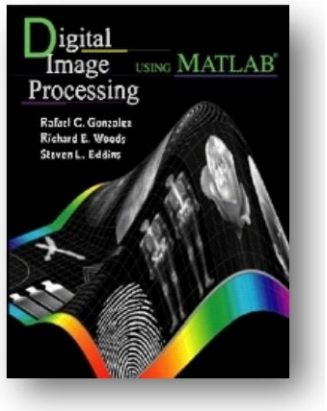


FIGURE 1.1 The MATLAB desktop and its principal components.



Introdução ao Matlab

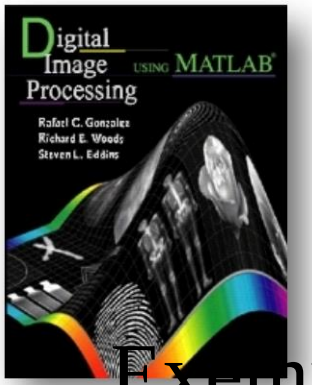
Lendo imagens:

f = imread ('filename'), lê a imagem do arquivo para a matriz f.

Format Name	Description	Recognized Extensions
TIFF	Tagged Image File Format	`.tif`, `.tiff`
JPEG	Joint Photographic Experts Group	`.jpg`, `.jpeg`
GIF	Graphics Interchange Format†	`.gif`
BMP	Windows Bitmap	`.bmp`
PNG	Portable Network Graphics	`.png`
XWD	X Window Dump	`.xwd`

† `GIF` is supported by `imread`, but not by `imwrite`.

TABLE 2.1 Some of the image/graphics formats supported by `imread` and `imwrite`, starting with MATLAB 6.5. Earlier versions support a subset of these formats. See online help for a complete list of supported formats.



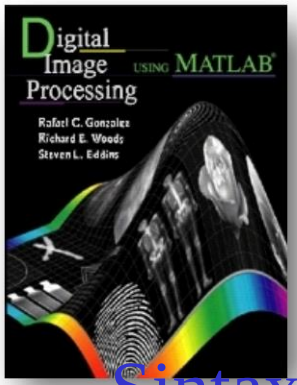
Introdução ao Matlab

Exemplos de comandos em Matlab para ler uma imagem e obter informações:

```
>> f = imread('Fig0222(a)(face).tif');
>> size(f)
>> [M N] = size(f)
>> whos f
>> figure, imshow(f, 'InitialMagnification', 'fit');
>> imwrite(f, 'teste.tif');
```

M- function : function [output] = function_name(input)

Exemplo : cria_imagem(x,y,filename)



Introdução ao Matlab

Sintaxe geral para imwrite():

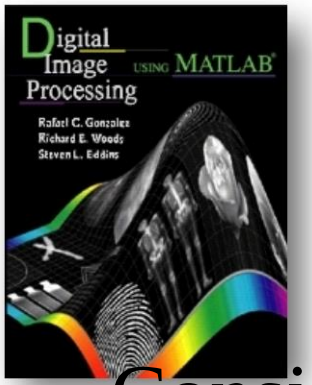
imwrite(f, filename, ‘compression’, ‘*parameter*’,
‘resolution’, *[colres, rowres]*);

parameter: none – sem compressão

packbits – default para imagens não binárias

ccitt – default para imagens binárias

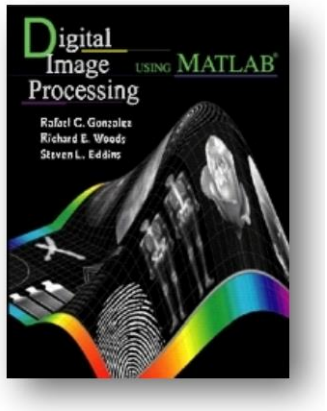
[colres, rowres]: resolução na coluna e resolução na linha



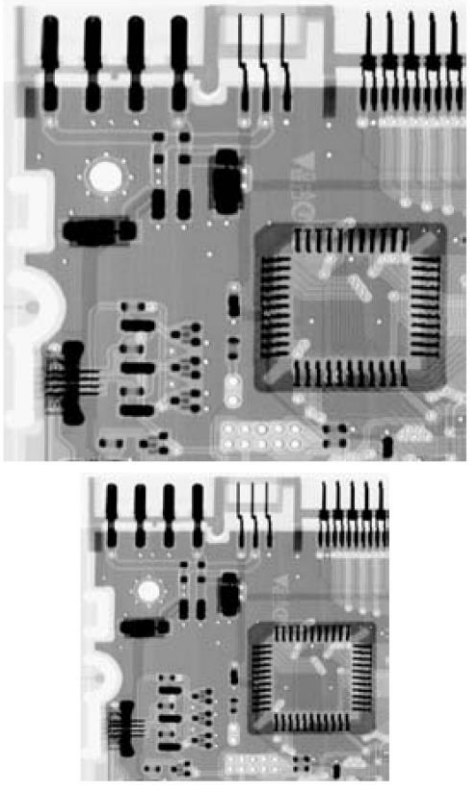
Introdução ao Matlab

Considere uma imagem de 450x450 pixels a 200dpi ➡ 2.25x2.25 inches. Queremos esta imagem em formato tif, sem compressão, com o mesmo numero de pixels e tamanho 1.5x1.5 inches:

```
res = round(200*2.25/1.5) = 300;
imwrite(f, 'sf.tif', 'compression', 'none', 'resolution', res);
```

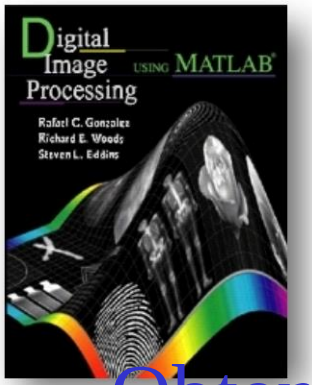


Introdução ao Matlab



a
b

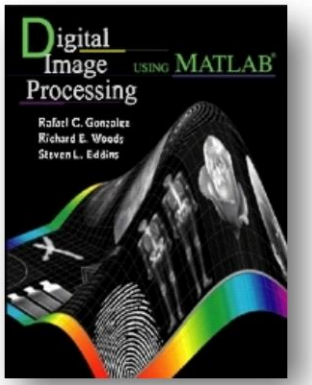
FIGURE 2.5 Effects of changing the dpi resolution while keeping the number of pixels constant. (a) A 450 × 450 image at 200 dpi (size = 2.25 × 2.25 inches). (b) The same 450 × 450 image, but at 300 dpi (size = 1.5 × 1.5 inches). (Original image courtesy of Lixi, Inc.)



Introdução ao Matlab

Obtendo informações sobre a imagem:

```
>> imfinfo filename
>> k = imfinfo('filename');
>> Xres = k.Xresolution
```

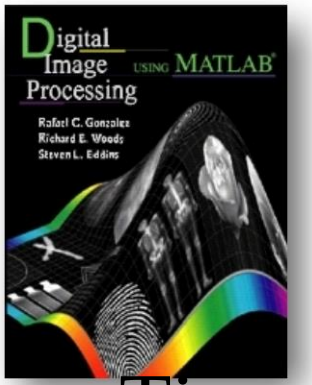


Introdução ao Matlab

Classes de dados:

TABLE 2.2
Data classes. The first eight entries are referred to as *numeric* classes; the ninth entry is the *character* class, and the last entry is of class *logical*.

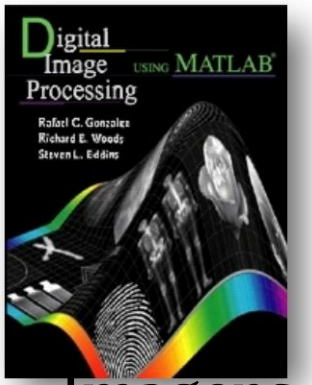
Name	Description
`double`	Double-precision, floating-point numbers in the approximate range -10^{308} to 10^{308} (8 bytes per element).
`uint8`	Unsigned 8-bit integers in the range [0, 255] (1 byte per element).
`uint16`	Unsigned 16-bit integers in the range [0, 65535] (2 bytes per element).
`uint32`	Unsigned 32-bit integers in the range [0, 4294967295] (4 bytes per element).
`int8`	Signed 8-bit integers in the range [−128, 127] (1 byte per element).
`int16`	Signed 16-bit integers in the range [−32768, 32767] (2 bytes per element).
`int32`	Signed 32-bit integers in the range [−2147483648, 2147483647] (4 bytes per element).
`single`	Single-precision floating-point numbers with values in the approximate range -10^{38} to 10^{38} (4 bytes per element).
`char`	Characters (2 bytes per element).
`logical`	Values are 0 or 1 (1 byte per element).



Introdução ao Matlab

Tipos de imagens:

- imagens em tons de cinza;
- imagens binarias
- imagens indexadas
- imagens RGB



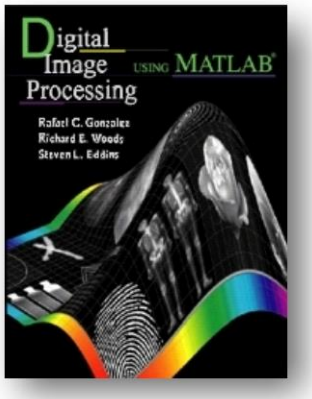
Introdução ao Matlab

Imagens em tons de cinza

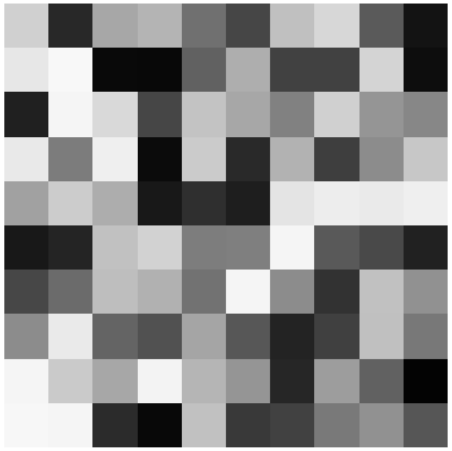
-é uma matriz de dados cujos valores representam níveis de cinza;

- class unit8 >> [0,255]
- class unit16 >>[0,65535]
- class double >> [0,1]

```
f  = [0 0 0 0; 0 0 0 0; 0.5 0.5 0.5 0.5; 0.5 0.5 0.5 0.5];
fg = im2uint8(f);
imshow(fg,‘InitialMagnification’, ‘fit’);
```

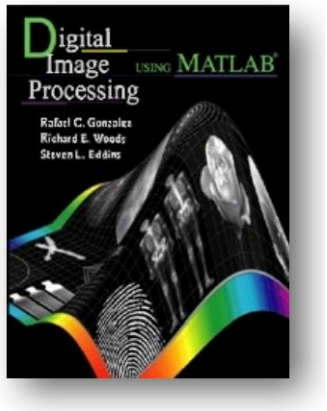


Introdução ao Matlab



```
ff = rand(10,10)
```

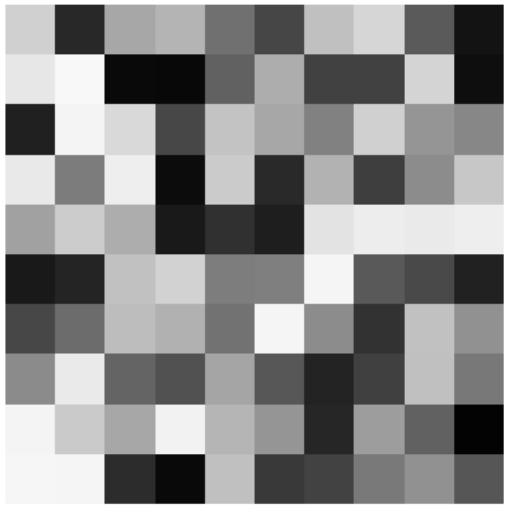
```
ff =
    0.8147    0.1576    0.6557    0.7060    0.4387    0.2760    0.7513    0.8407    0.3517    0.0759
    0.9058    0.9706    0.0357    0.0318    0.3816    0.6797    0.2551    0.2543    0.8308    0.0540
    0.1270    0.9572    0.8491    0.2769    0.7655    0.6551    0.5060    0.8143    0.5853    0.5308
    0.9134    0.4854    0.9340    0.0462    0.7952    0.1626    0.6991    0.2435    0.5497    0.7792
    0.6324    0.8003    0.6787    0.0971    0.1869    0.1190    0.8909    0.9293    0.9172    0.9340
    0.0975    0.1419    0.7577    0.8235    0.4898    0.4984    0.9593    0.3500    0.2858    0.1299
    0.2785    0.4218    0.7431    0.6948    0.4456    0.9597    0.5472    0.1966    0.7572    0.5688
    0.5469    0.9157    0.3922    0.3171    0.6463    0.3404    0.1386    0.2511    0.7537    0.4694
    0.9575    0.7922    0.6555    0.9502    0.7094    0.5853    0.1493    0.6160    0.3804    0.0119
    0.9649    0.9595    0.1712    0.0344    0.7547    0.2238    0.2575    0.4733    0.5678    0.3371
```



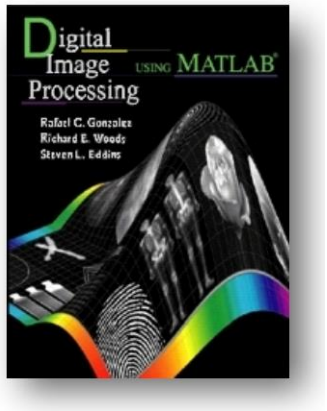
Introdução ao Matlab

```
g = im2unit8(ff)
g =

  208   40  167  180  112   70  192  214   90   19
  231  248    9    8   97  173   65   65  212   14
   32  244  217   71  195  167  129  208  149  135
  233  124  238   12  203   41  178   62  140  199
  161  204  173   25   48   30  227  237  234  238
   25   36  193  210  125  127  245   89   73   33
   71  108  189  177  114  245  140   50  193  145
  139  234  100   81  165   87   35   64  192  120
  244  202  167  242  181  149   38  157   97    3
  246  245   44    9  192   57   66  121  145   86
```



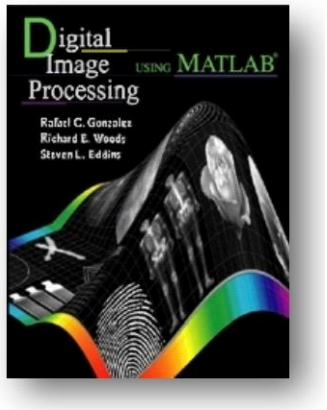
```
Imshow(g, 'InitialMagnification', 'fit');
```



Introdução ao Matlab

Name	Converts Input to:	Valid Input Image Data Classes
`im2uint8`	`uint8`	`logical`, `uint8`, `uint16`, and `double`
`im2uint16`	`uint16`	`logical`, `uint8`, `uint16`, and `double`
`mat2gray`	`double` (in range $[0, 1]$)	`double`
`im2double`	`double`	`logical`, `uint8`, `uint16`, and `double`
`im2bw`	`logical`	`uint8`, `uint16`, and `double`

TABLE 2.3
Functions in IPT for converting between image classes and types. See Table 6.3 for conversions that apply specifically to color images.

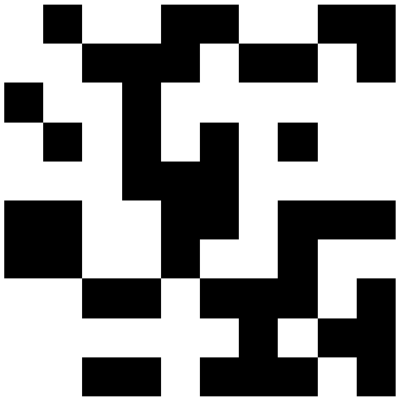


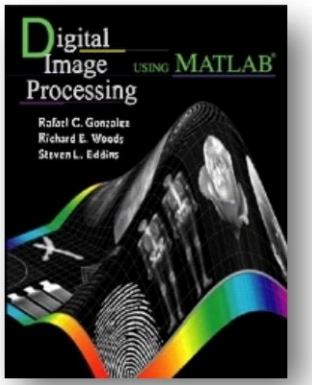
Introdução ao Matlab

bw = im2bw(g, 0.5); T = threshold no intervalo [0,1]

bw =

1	0	1	1	0	0	1	1	0	0
1	1	0	0	0	1	0	0	1	0
0	1	1	0	1	1	1	1	1	1
1	0	1	0	1	0	1	0	1	1
1	1	1	0	0	0	1	1	1	1
0	0	1	1	0	0	1	0	0	0
0	0	1	1	0	1	1	0	1	1
1	1	0	0	1	0	0	0	1	0
1	1	1	1	1	1	0	1	0	0
1	1	0	0	1	0	0	0	1	0

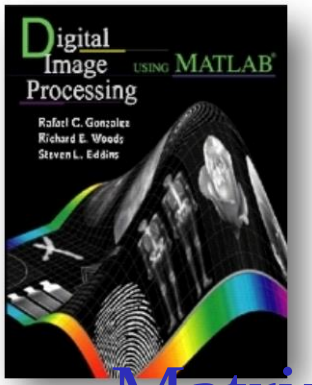




Introdução ao Matlab

Vetores (primeiro elemento é v(1))

definiçao de um vetor → v = [1 3 5 7];
acesso ao 2º. elemelnto → v(2) = 3;
transposta de v → w = v.'
acesso a blocos → v(1:3) = 1 3 5;
→ v(3:end) = 5 7;
→ v(1:2:end) = 1 5
→ v(end:-2:1) = 7 3
gera vetor linearmente espaçado → x = linspace(1,10,5) =
1.0000 3.2500 5.5000 7.7500 10.0000



Introdução ao Matlab

Matrizes (primeiro elemento é mat(1,1))

Cria matriz → A = [1 2 3;4 5 6;7 8 9];

Acesso a um elemento → A(2,3) = 6;

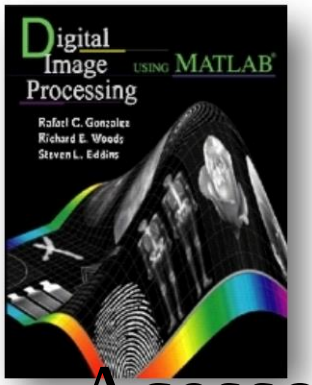
A(end,end) = 9

Acesso a uma coluna → C = A(:,3);

Acesso a uma linha → L = A(2,:) = 4 5 6

Acesso a um bloco → B = A(1:2, 1:3) = 1 2 3

4 5 6

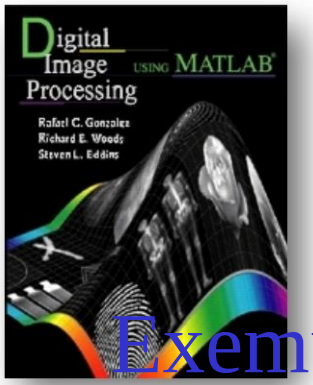


Introdução ao Matlab

Acesso a blocos → A(2:end, end:-2:1) = 6 4
9 7

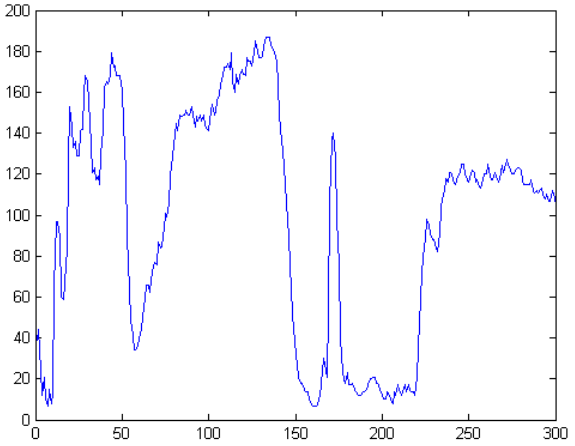
arranja uma matriz na forma de vetor , coluna por coluna → A(:) = [1 4 7 2 5 8 3 6 9]t

rebate matriz → A(end:-1:1, :)

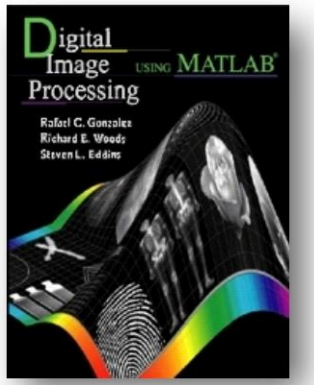


Introdução ao Matlab

Exemplo: modifica_lena.m

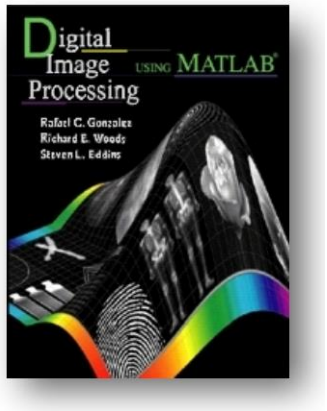


149, 290 = 202



Operator	Name	MATLAB Function	Comments and Examples
+	Array and matrix addition	`plus(A, B)`	`a + b`, `A + B`, or `a + A`.
–	Array and matrix subtraction	`minus(A, B)`	`a – b`, `A – B`, `A – a`, or `a – A`.
`.*`	Array multiplication	`times(A, B)`	`C = A.*B`, `C(I, J) = A(I, J)*B(I, J)`.
*	Matrix multiplication	`mtimes(A, B)`	`A*B`, standard matrix multiplication, or `a*A`, multiplication of a scalar times all elements of `A`.
`./`	Array right division	`rdivide(A, B)`	`C = A./B`, `C(I, J) = A(I, J)/B(I, J)`.
`.\`	Array left division	`ldivide(A, B)`	`C = A.\B`, `C(I, J) = B(I, J)/A(I, J)`.
/	Matrix right division	`mrdivide(A, B)`	`A/B` is roughly the same as `A*inv(B)`, depending on computational accuracy.
\	Matrix left division	`mldivide(A, B)`	`A\B` is roughly the same as `inv(A)*B`, depending on computational accuracy.
`.^`	Array power	`power(A, B)`	If `C = A.^B`, then `C(I, J) = A(I, J)^B(I, J)`.
^	Matrix power	`mpower(A, B)`	See online help for a discussion of this operator.
`.'`	Vector and matrix transpose	`transpose(A)`	`A.'`. Standard vector and matrix transpose.
'	Vector and matrix complex conjugate transpose	`ctranspose(A)`	`A'`. Standard vector and matrix conjugate transpose. When `A` is real `A.' = A'`.
+	Unary plus	`uplus (A)`	`+A` is the same as `0 + A`.
–	Unary minus	`uminus (A)`	`–A` is the same as `0 – A` or `–1*A`.
:	Colon		Discussed in Section 2.8.

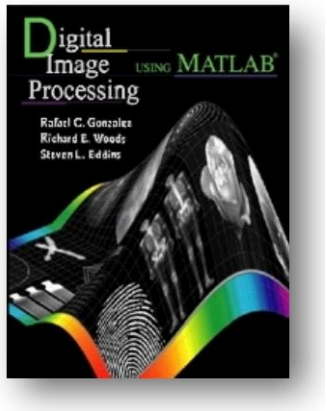
TABLE 2.4
Array and matrix arithmetic operators. Computations involving these operators can be implemented using the operators themselves, as in `A + B`, or using the MATLAB functions shown, as in `plus (A, B)`. The examples shown for arrays use matrices to simplify the notation, but they are easily extendable to higher dimensions.



Introdução ao Matlab

Function	Description
`imadd`	Adds two images; or adds a constant to an image.
`imsubtract`	Subtracts two images; or subtracts a constant from an image.
`immultiply`	Multiplies two images, where the multiplication is carried out between pairs of corresponding image elements; or multiplies a constant times an image.
`imdivide`	Divides two images, where the division is carried out between pairs of corresponding image elements; or divides an image by a constant.
`imabsdiff`	Computes the absolute difference between two images.
`imcomplement`	Complements an image. See Section 3.2.1.
`imlincomb`	Computes a linear combination of two or more images. See Section 5.3.1 for an example.

TABLE 2.5
The image arithmetic functions supported by IPT.



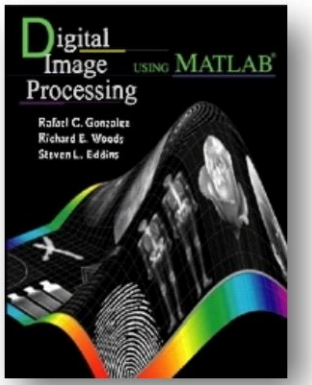
Introdução ao Matlab

Operator	Name
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
~=	Not equal to

TABLE 2.6
Relational operators.

Operator	Name
&	AND
|	OR
~	NOT

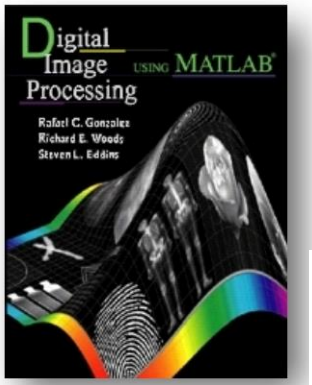
TABLE 2.7
Logical operators.



Introdução ao Matlab

Function	Comments
`xor` (exclusive OR)	The `xor` function returns a 1 only if both operands are logically different; otherwise `xor` returns a 0.
`all`	The `all` function returns a 1 if all the elements in a vector are nonzero; otherwise `all` returns a 0. This function operates columnwise on matrices.
`any`	The `any` function returns a 1 if any of the elements in a vector is nonzero; otherwise `any` returns a 0. This function operates columnwise on matrices.

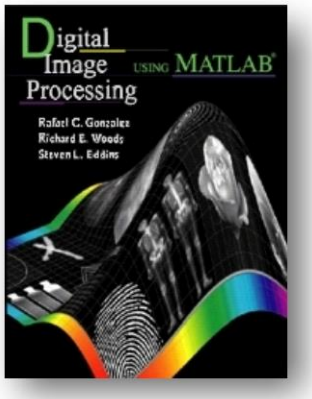
TABLE 2.8
Logical functions.



Introdução ao Matlab

Function	Description
`iscell(C)`	True if `C` is a cell array.
`iscellstr(s)`	True if `s` is a cell array of strings.
`ischar(s)`	True if `s` is a character string.
`isempty(A)`	True if `A` is the empty array, [ ].
`isequal(A, B)`	True if `A` and `B` have identical elements and dimensions.
`isfield(S, 'name')`	True if `'name'` is a field of structure `S`.
`isfinite(A)`	True in the locations of array `A` that are finite.
`isinf(A)`	True in the locations of array `A` that are infinite.
`isletter(A)`	True in the locations of `A` that are letters of the alphabet.
`islogical(A)`	True if `A` is a logical array.
`ismember(A, B)`	True in locations where elements of `A` are also in `B`.
`isnan(A)`	True in the locations of `A` that are `NaN`s (see Table 2.10 for a definition of `NaN`).
`isnumeric(A)`	True if `A` is a numeric array.
`isprime(A)`	True in locations of `A` that are prime numbers.
`isreal(A)`	True if the elements of `A` have no imaginary parts.
`isspace(A)`	True at locations where the elements of `A` are whitespace characters.
`issparse(A)`	True if `A` is a sparse matrix.
`isstruct(S)`	True if `S` is a structure.

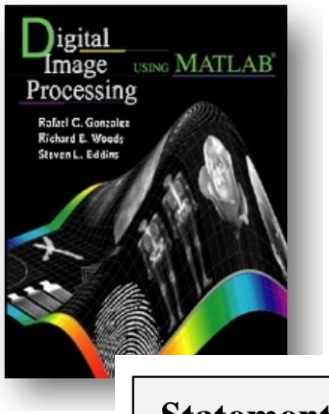
TABLE 2.9 Some functions that return a logical 1 or a logical 0 depending on whether the value or condition in their arguments are `true` or `false`. See online help for a complete list.



Introdução ao Matlab

Function	Value Returned
`ans`	Most recent answer (variable). If no output variable is assigned to an expression, MATLAB automatically stores the result in `ans`.
`eps`	Floating-point relative accuracy. This is the distance between 1.0 and the next largest number representable using double-precision floating point.
`i` (or `j`)	Imaginary unit, as in `1 + 2i`.
`NaN` or `nan`	Stands for Not-a-Number (e.g., `0/0`).
`pi`	3.14159265358979
`realmax`	The largest floating-point number that your computer can represent.
`realmin`	The smallest floating-point number that your computer can represent.
`computer`	Your computer type.
`version`	MATLAB version string.

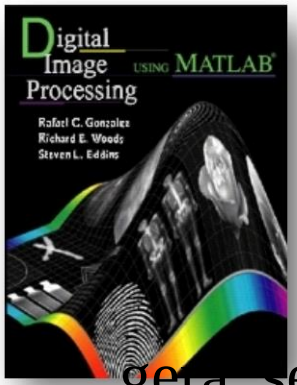
TABLE 2.10
Some important variables and constants.



Introdução ao Matlab

Statement	Description
`if`	`if`, together with `else` and `elseif`, executes a group of statements based on a specified logical condition.
`for`	Executes a group of statements a fixed (specified) number of times.
`while`	Executes a group of statements an indefinite number of times, based on a specified logical condition.
`break`	Terminates execution of a `for` or `while` loop.
`continue`	Passes control to the next iteration of a `for` or `while` loop, skipping any remaining statements in the body of the loop.
`switch`	`switch`, together with `case` and `otherwise`, executes different groups of statements, depending on a specified value or string.
`return`	Causes execution to return to the invoking function.
`try...catch`	Changes flow control if an error is detected during execution.

TABLE 2.11 Flow control statements.



Introdução ao Matlab

gera_seno_vetorv1(Ampli,tudo M, frequencia)
gera_seno_vetorv1(1,512,1/(4*pi));

f = gera_seno_2d(1,1/(4*pi), 1/(4*pi),512,512);