

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

Centro Tecnológico

Departamento de Engenharia Elétrica

Princípios de Comunicações I

Introdução à Linguagem de Computação Matlab

Prof.: Jair A. Lima Silva

UFES, 2012/2

Índice

1. **Definição e Ambiente de Trabalho da Linguagem**
2. **Operações Aritméticas, Números Complexos, etc**
3. **Definição, Operação e Manipulação de Matrizes**
4. **Scripts e Funções no Matlab**
5. **Estruturas de Controle de Fluxo**
6. **Gráficos**
7. **Exemplo de Aplicação**



1. Definição da Linguagem

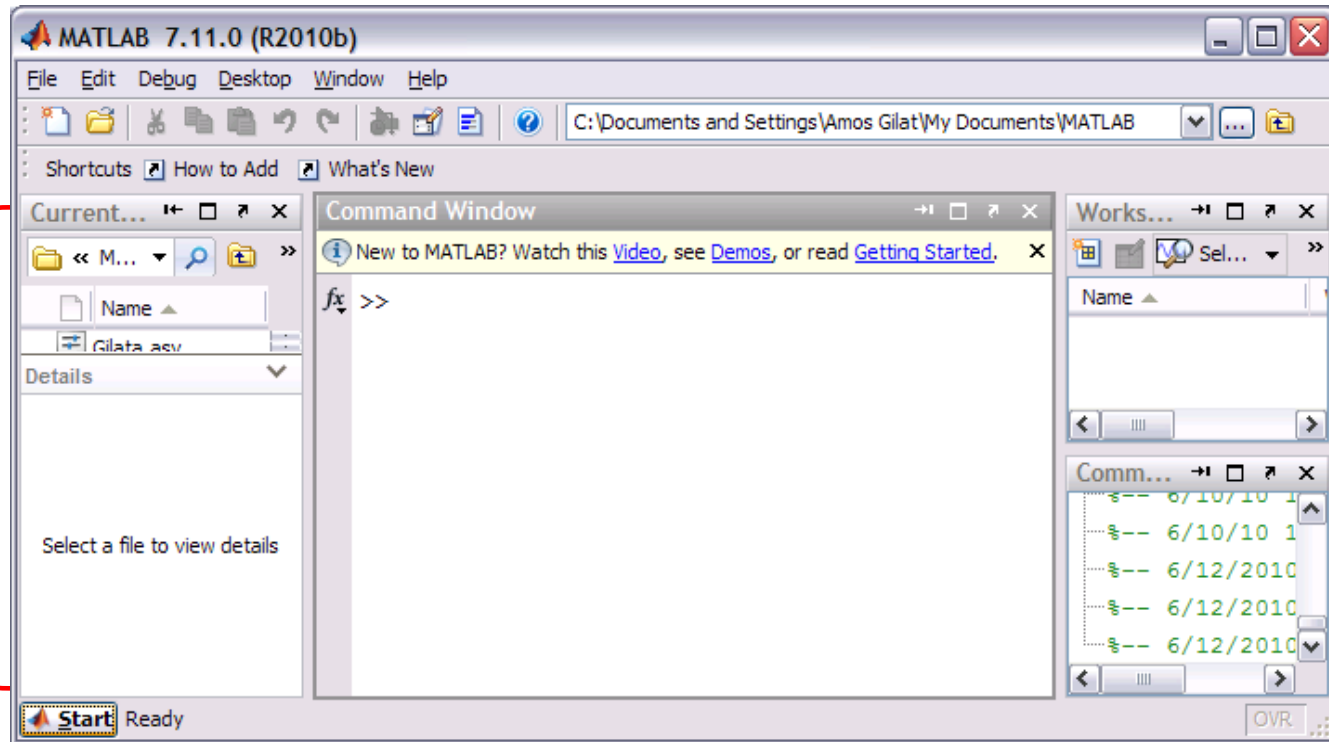
- Vem sendo utilizado em Universidades, (em cursos de Matemática, Ciências e Engenharias) e em Indústrias como ferramenta de pesquisa, projeto e desenvolvimento.

Datasheet:

<http://www.mathworks.com/products/datasheets/pdf/matlab.pdf>

Contacto: <http://www.mathworks.com/products/matlab/>

1. Ambiente de Trabalho



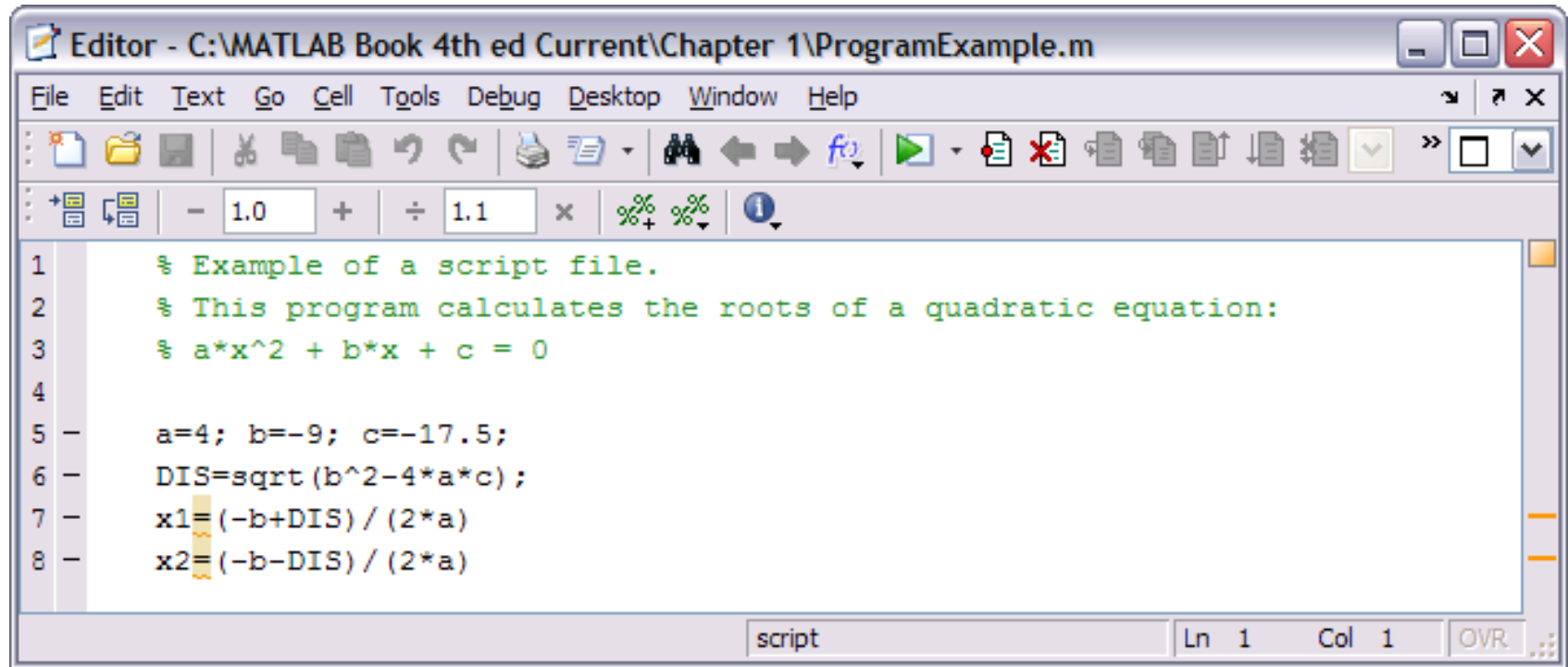
Current Folder Window

Workspace Window

Command History Window

Command Window

1. Ambiente de Trabalho



Editor/Debugger Window

O editor de *scripts* pode estar ou não acoplado à janela principal

1. Ambiente de Trabalho

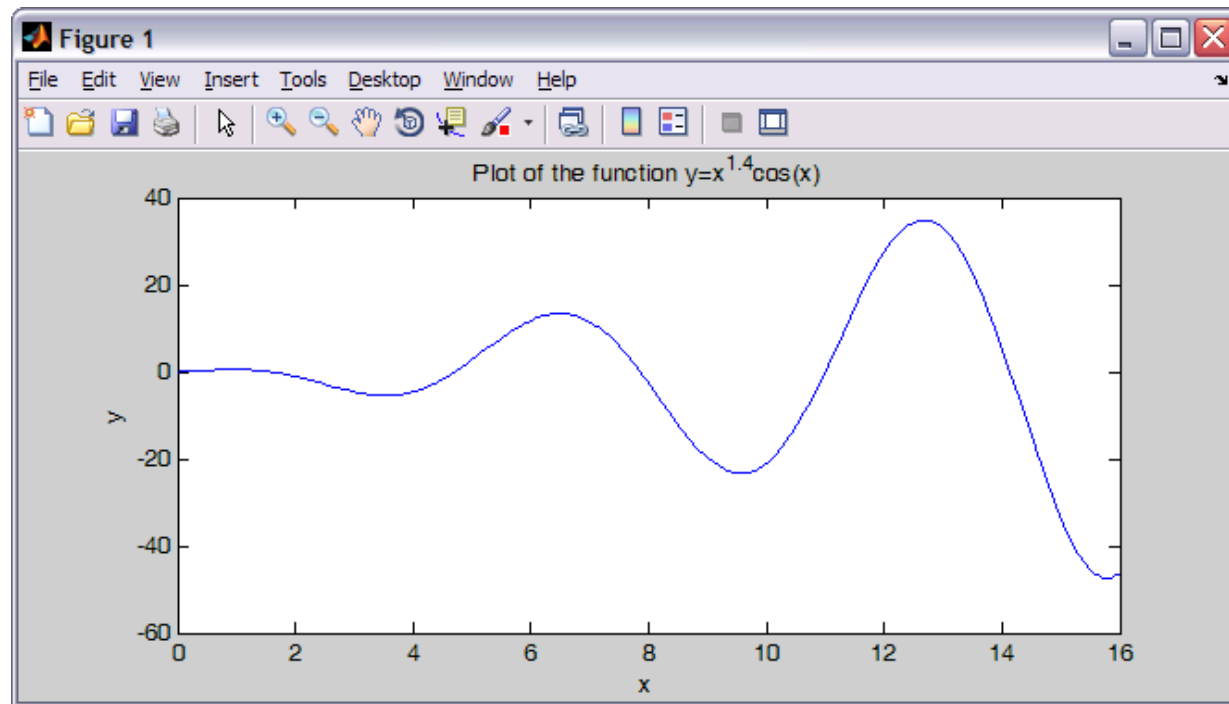
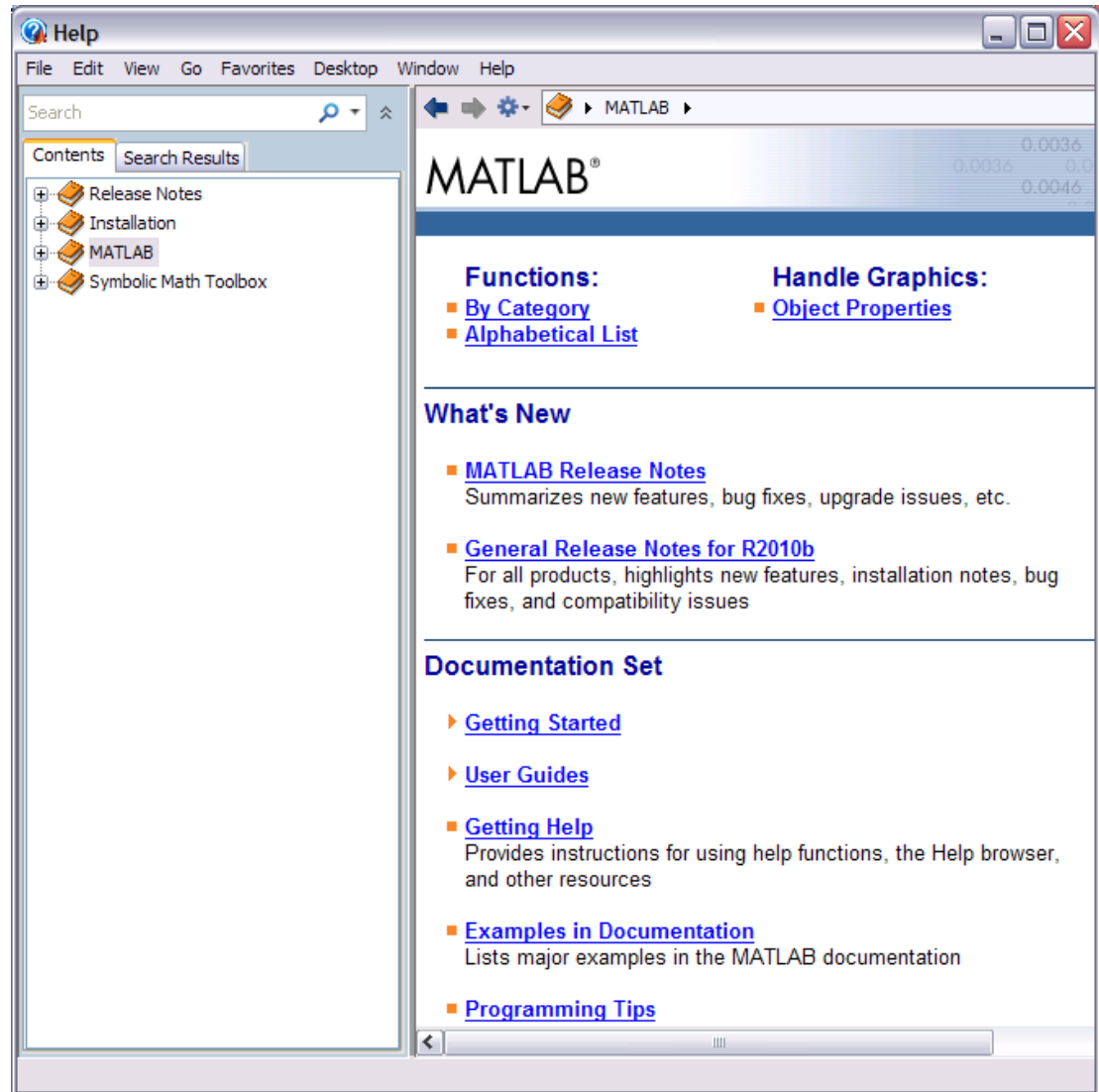


Figure Window

As figuras aparecem automaticamente após a execução de algum comando de visualização.

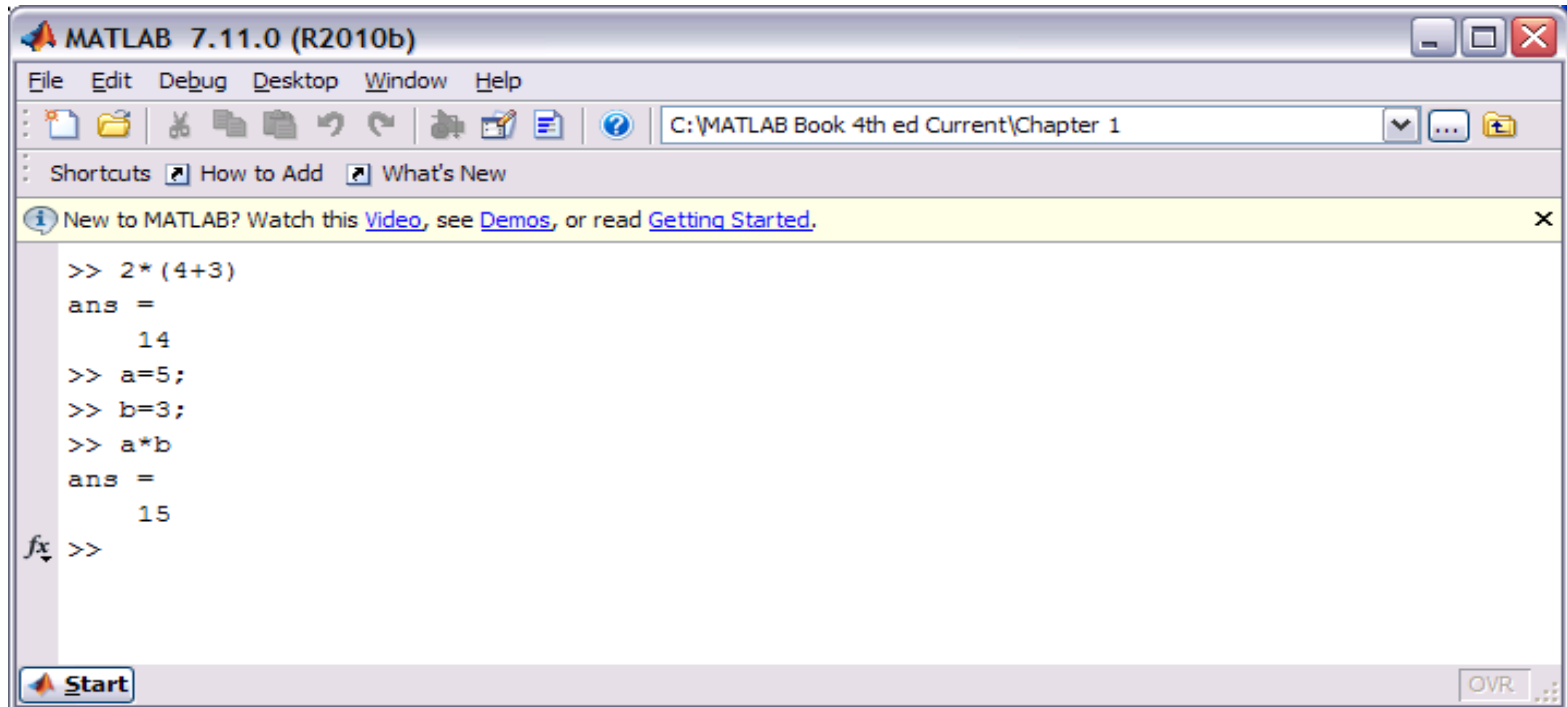
1. Ambiente de Trabalho

Existe ainda uma janela de ajuda que pode ser acessada em menus ou no *prompt*.



2. Operações Aritméticas

- O MATLAB possui todas as operações básicas de uma calculadora. Assim, no *prompt* da janela de comando podes:



A screenshot of the MATLAB 7.11.0 (R2010b) Command Window. The window title is "MATLAB 7.11.0 (R2010b)". The menu bar includes File, Edit, Debug, Desktop, Window, and Help. The toolbar shows icons for file operations and help. The address bar displays "C:\MATLAB Book 4th ed Current\Chapter 1". A yellow banner at the top of the command window area says "New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#)." The command window shows the following input and output:

```
>> 2*(4+3)
ans =
    14
>> a=5;
>> b=3;
>> a*b
ans =
    15
fx >>
```

The status bar at the bottom shows a "Start" button and an "OVR" indicator.

2. Operações Aritméticas

Alguns símbolos para a Aritmética

Operation	Symbol	Example
Addition	+	$5 + 3$
Subtraction	—	$5 - 3$
Multiplication	*	$5 * 3$
Right division	/	$5 / 3$
Left division	\	$5 \setminus 3 = 3 / 5$
Exponentiation	^	$5 \wedge 3$ (means $5^3 = 125$)

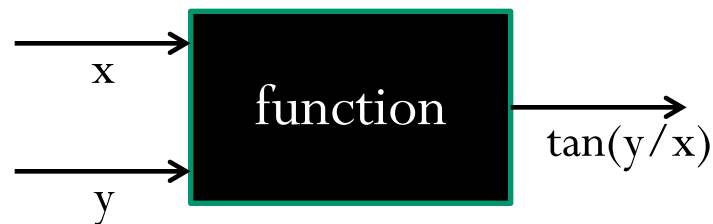
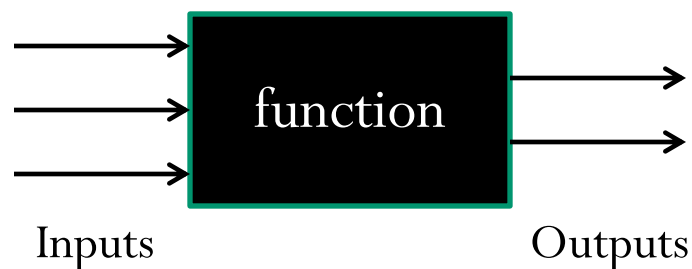
Exercício: Qual a ordem de execução dos cálculos?

2. Comandos de formatação de números

Command	Description	Example
format short	Fixed-point with 4 decimal digits for: $0.001 \leq \textit{number} \leq 1000$ Otherwise display format short e	>> 290/7 ans = 41.4286
format long	Fixed-point with 15 decimal digits for: $0.001 \leq \textit{number} \leq 1000$ Otherwise display format long e	>> 290/7 ans = 41.428571428571431
format short e	Scientific notation with 4 decimal digits	>> 290/7 ans = 4.1429e+001
format long e	Scientific notation with 15 decimal digits	>> 290/7 ans = 4.142857142857143e+001
format compact	Eliminates empty lines to allow more lines with information displayed on the screen	
format loose	Adds empty lines (opposite of compact)	

2. Funções Reservadas

- As expressões podem incluir funções (caixas pretas) para facilitar a computação de dados.



$$y = \text{sqrt}(x)$$

output name argument

$\text{sqrt}(64)$ Argument is a number

$\text{sqrt}(a)$ Argument is the variable "a"

$\text{atan}(y/\text{sqrt}(3^2+y^2))$

Argument to arctan function is an expression

2. Funções Reservadas

Algumas funções elementares

- **sqrt(x)** – square root
- **nthroot(x,n)** – nth real root
- **exp(x)** – e^x
- **abs(x)** – absolute value
- **log(x)** – natural log (base e)
- **log10(x)** – log base 10
- **factorial(x)** – $x!$

2. Funções Reservadas

Algumas funções trigonométricas

- **$\sin(\mathbf{x})$** – sine (x in radians)
- **$\text{sin}d(\mathbf{x})$** – sine (x in degrees)
- **$\cos(\mathbf{x})$** – cosine (x in radians)
- **$\text{cos}d(\mathbf{x})$** – cosine (x in degrees)
- **$\tan(\mathbf{x})$** – tangent (x in radians)
- **$\text{tan}d(\mathbf{x})$** – tangent (x in degrees)
- **$\cot(\mathbf{x})$** – cotangent (x in radians)
- **$\text{cot}d(\mathbf{x})$** – cotangent (x in degrees)

2. Funções Reservadas

Exercício: Qual a funcionalidade das funções abaixo?

- `round(x)`
- `fix(x)`
- `ceil(x)`
- `floor(x)`
- `rem(x,y)`
- `sign(x)`

2. Funções Reservadas

Outras funções reservadas e muito utilizadas

- clear** - Removes all variables from memory
- clear x y z** - Removes only variables x, y, and z from memory
- who** - Displays a list of the variables currently in memory
- whos** - Displays a list of the variables currently in memory and their size, together with information about their bytes and class

Veja também no *Help* as funções *clc*, *close*, *hold*, etc

2. Números Complexos

- Os números complexos podem ser facilmente definidos pelos três métodos descritos a seguir:

```
>> a = 1 - 2i;
```

```
>> a = 1 - 2j;
```

```
>> a = 1 - 2*sqrt(-1);
```

COMPLEXAS	
abs	Valor Absoluto
angle	Argumento (em radianos)
conj	Complexo conjugado
imag	Parte imaginaria
real	Parte real

3. Definição, Operação e Manipulação de Matrizes

- Os **arrays** ou arranjos são estruturas básicas que sustentam a base operacional do software MATLAB.
- Estes podem conter várias dimensões e podem ser formadas de números e/ou caracteres:
 - **Vetor**: Arranjo de dimensão 1 (1 linha ou 1 coluna)
 - **Matriz**: 2 ou mais dimensões.

3. Criação de Matrizes

Vetor Linha:

Syntax: `variable_name = [n1, n2, n3]`


Commas optional

```
>> yr = [1984 1986 1988]
```

```
yr =
```

```
1984 1986 1988
```

MATLAB displays row vector horizontally

3. Criação de Matrizes

Vetor Coluna:

Syntax: `variable_name = [n1; n2; n3]`

```
>> yr = [1984; 1986; 1988]
```

```
yr =
```

```
    1984
```

```
    1986
```

```
    1988
```

MATLAB displays column vector vertically

3. Criação de Matrizes

Outras formas:

Syntax: `variable_name = m:q:n`

- `m` is first number
- `n` is last number
- `q` is difference between consecutive numbers

`v = m:q:n` means

`v = [m m+q m+2q m+3q ... n]`

3. Criação de Matrizes

Outras formas:

Testes:

- $(m:q:n)$ — Gere um vetor sem o parâmetro q
- $(m:q:n)$ — Gere um vetor com um q negativo
- $(m:q:n)$ — Gere um vetor com um $q = 0.1$

3. Criação de Matrizes

Outras formas:

Syntax: $v = \text{linspace}(xi, xf, n)$

- xi is first number
- xf is last number
- n is number of terms

Exemplos:

```
>> va = linspace(0,8,6)
```

```
va = 0 1.6000 3.2000 4.8000 6.4000 8.0000
```

```
>> va = linspace(30,10,11)
```

```
va = 30 28 26 24 22 20 18 16 14 12 10
```

3. Criação de Matrizes

Exemplo de Aplicação:

- a) Gere um vetor tempo com $0.4 \mu\text{s}$ de duração, contendo 100 posições.

```
>> t = linspace(0,0.4e-6,100);
```

- b) Qual a taxa de amostragem deste vetor tempo.

```
>> ta = t(2) - t(1);
```

- c) Gere um vetor frequência a partir dos dados gerados acima.

```
>> deltaf = 1/ta;
```

```
>> fpos = 0:deltaf:(100/2-1)*deltaf;
```

```
>> fneg = -(100/2-1)*deltaf:deltaf:-deltaf;
```

```
>> f = [fpos fneg];
```

3. Criação de Matrizes

Matrizes

Syntax:

m = [row 1 numbers; row 2 numbers;...;last row numbers]

- Each row separated by semicolon
- All rows have same number of columns

```
>> a=[ 5 35 43; 4 76 81; 21 32 40]
```

```
a =
```

```
    5    35    43  
    4    76    81  
   21    32    40
```


3. Criação de Matrizes

Matrizes

Exemplos:

```
>> cd=6; e=3; h=4;
```

```
>> Mat=[e,cd*h,cos(pi/3);h^2 sqrt(h*h/cd) 14]
```

```
Mat =
```

```
    3.0000    24.0000    0.5000
```

```
   16.0000    1.6330   14.0000
```

3. Criação de Matrizes

Matrizes

Exercício: Quais as funcionalidades das funções?

- `zeros(m,n)`
- `ones(m,n)`
- `eye(n)`
- `rand(m,n)`
- `randn(m,n)`

>> `z=100*ones(3,4)` ??????

3. Operações com Matrizes

Transposta

```
>> aa=[3 8 1]
```

```
aa =    3    8    1
```

```
>> bb=aa'
```

```
bb =  3
```

```
      8
```

```
      1
```

3. Operações com Matrizes

Transposta

```
>> C=[2 55 14 8; 21 5 32 11; 41 64 9 1]
```

```
C =  2   55   14    8
      21    5   32   11
      41   64    9    1
```

```
>> D=C'
```

```
D =  2   21   41
      55    5   64
      14   32    9
      8   11    1
```

**Faça a transposta de uma
matriz de números
complexos!!**

As operações aritméticas são realizadas elemento a elemento!

Operações com Arrays

Sabendo que : $A = [a_1 \ a_2 \ \dots \ a_n]$; $B = [b_1 \ b_2 \ \dots \ b_n]$; c – Escalar

Adição com um escalar

$$A+c = [a_1+c \ a_2+c \ \dots \ a_n+c]$$

Multiplicação com um escalar

$$A*c = [a_1*c \ a_2*c \ \dots \ a_n*c]$$

Adição

$$A+B = [a_1+b_1 \ a_2+b_2 \ \dots \ a_n+b_n]$$

Multiplicação

$$A.*B = [a_1.*b_1 \ a_2.*b_2 \ \dots \ a_n.*b_n]$$

Divisão à esquerda

$$A.\backslash B = [a_1.\backslash b_1 \ a_2.\backslash b_2 \ \dots \ a_n.\backslash b_n]$$

Divisão à direita

$$A./B = [a_1./b_1 \ a_2./b_2 \ \dots \ a_n./b_n]$$

Potenciação

$$A.^c = [a_1.^c \ a_2.^c \ \dots \ a_n.^c]$$

$$c.^A = [c.^{a_1} \ c.^{a_2} \ \dots \ c.^{a_n}]$$

$$A.^B = [a_1.^{b_1} \ a_2.^{b_2} \ \dots \ a_n.^{b_n}]$$

Symbol

Description

Symbol

Description

.*

Multiplication

/

Right division

.^

Exponentiation

.\

Left Division

As regras da Algebra devem ser respeitadas

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \\ A_{41} & A_{42} & A_{43} \end{bmatrix} \text{ and } B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix}$$

then, the matrix that is obtained by the operation $A*B$ has the dimension of 4×2 with the elements:

$$\begin{bmatrix} (A_{11}B_{11} + A_{12}B_{21} + A_{13}B_{31}) & (A_{11}B_{12} + A_{12}B_{22} + A_{13}B_{32}) \\ (A_{21}B_{11} + A_{22}B_{21} + A_{23}B_{31}) & (A_{21}B_{12} + A_{22}B_{22} + A_{23}B_{32}) \\ (A_{31}B_{11} + A_{32}B_{21} + A_{33}B_{31}) & (A_{31}B_{12} + A_{32}B_{22} + A_{33}B_{32}) \\ (A_{41}B_{11} + A_{42}B_{21} + A_{43}B_{31}) & (A_{41}B_{12} + A_{42}B_{22} + A_{43}B_{32}) \end{bmatrix}$$

A numerical example is:

$$\begin{bmatrix} 1 & 4 & 3 \\ 2 & 6 & 1 \\ 5 & 2 & 8 \end{bmatrix} \begin{bmatrix} 5 & 4 \\ 1 & 3 \\ 2 & 6 \end{bmatrix} = \begin{bmatrix} (1 \cdot 5 + 4 \cdot 1 + 3 \cdot 2) & (1 \cdot 4 + 4 \cdot 3 + 3 \cdot 6) \\ (2 \cdot 5 + 6 \cdot 1 + 1 \cdot 2) & (2 \cdot 4 + 6 \cdot 3 + 1 \cdot 6) \\ (5 \cdot 5 + 2 \cdot 1 + 8 \cdot 2) & (5 \cdot 4 + 2 \cdot 3 + 8 \cdot 6) \end{bmatrix} = \begin{bmatrix} 15 & 34 \\ 18 & 32 \\ 43 & 74 \end{bmatrix}$$

3. Operações com Matrizes

Alguns exemplos:

```
>> A=[2 1 4; 4 1 8; 2 -1 3]
```

Creating the matrix A.

```
A =
```

```

    2    1    4
    4    1    8
    2   -1    3

```

```
>> B=inv(A)
```

Use the `inv` function to find the inverse of A and assign it to B.

```
B =
```

```

  5.5000  -3.5000   2.0000
  2.0000  -1.0000   0.0000
 -3.0000   2.0000  -1.0000

```

```
>> A*B
```

Multiplication of A and B gives the identity matrix.

```
ans =
```

```

    1    0    0
    0    1    0
    0    0    1

```

Sample Problem 3-1: Solving three linear equations (array division)

Use matrix operations to solve the following system of linear equations.

$$4x - 2y + 6z = 8$$

$$2x + 8y + 2z = 4$$

$$6x + 10y + 3z = 0$$

Solution

Using the rules of linear algebra demonstrated earlier, the above system of equations can be written in the matrix form $AX = B$ or in the form $XC = D$:

$$\begin{bmatrix} 4 & -2 & 6 \\ 2 & 8 & 2 \\ 6 & 10 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 8 \\ 4 \\ 0 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} 4 & 2 & 6 \\ -2 & 8 & 10 \\ 6 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 8 & 4 & 0 \end{bmatrix}$$

The solution of both forms is shown below:

```
>> A=[4 -2 6; 2 8 2; 6 10 3];
```

Solving the form $AX = B$.

```
>> B=[8; 4; 0];
```

```
>> X=A\B
```

Solving by using left division $X = A \setminus B$.

```
X =
```

```
   -1.8049
```

```
    0.2927
```

```
    2.6341
```

```
>> Xb=inv(A)*B
```

Solving by using the inverse of A $X = A^{-1}B$.

3. Operações com Matrizes

Alguns exemplos:

```
>> x=[0:pi/6:pi]
x =
    0    0.5236    1.0472    1.5708    2.0944    2.6180    3.1416
>>y=cos(x)
y =
    1.0000    0.8660    0.5000    0.0000   -0.5000   -0.8660   -1.0000
>>
```

3. Manipulação de Matrizes

Endereçamento

- **Vector (row or column)**
 - $va(:)$ - all elements
 - $va(m:n)$ - elements m through n
- **Matrix**
 - $A(:,n)$ - all rows of column n
 - $A(m,:)$ - all columns of row m
 - $A(:,m:n)$ - all rows of columns m through n
 - $A(m:n,:)$ - all columns of rows m through n
 - $A(m:n,p:q)$ - columns p through q of rows m through n

3. Manipulação de Matrizes

Endereçamento

- Exemplos:

```
>> MAT=[3 11 6 5; 4 7 10 2; 13 9 0 8]
```

coluna 1

```
MAT = 3 11 6 5
      4 7 10 2
```

Elementos da

linha 3 e coluna 1

```
13 9 0 8
```

Linha 3

```
>> MAT(3,1)
```

ans = 13

Atribua um novo valor ao elemento da linha 3 e coluna 1

```
>> MAT(3,1)=20
```

```
MAT = 3 11 6 5
```

```
4 7 10 2
```

```
20 9 0 8
```

```
>> MAT(2,4)-MAT(1,2)
```

ans = -9

*Delete uma linha da
matriz usando []*

3. Manipulação de Matrizes

Exercício: Qual a funcionalidade das funções abaixo?

- length
- size
- reshape
- diag
- inv
- det
- eig
- poly

- mean
- max
- min
- sort
- sum
- std
- cross

4. Scripts e Funções no MATLAB

Modularizar significa simplificar ➔ Solução mais rápida

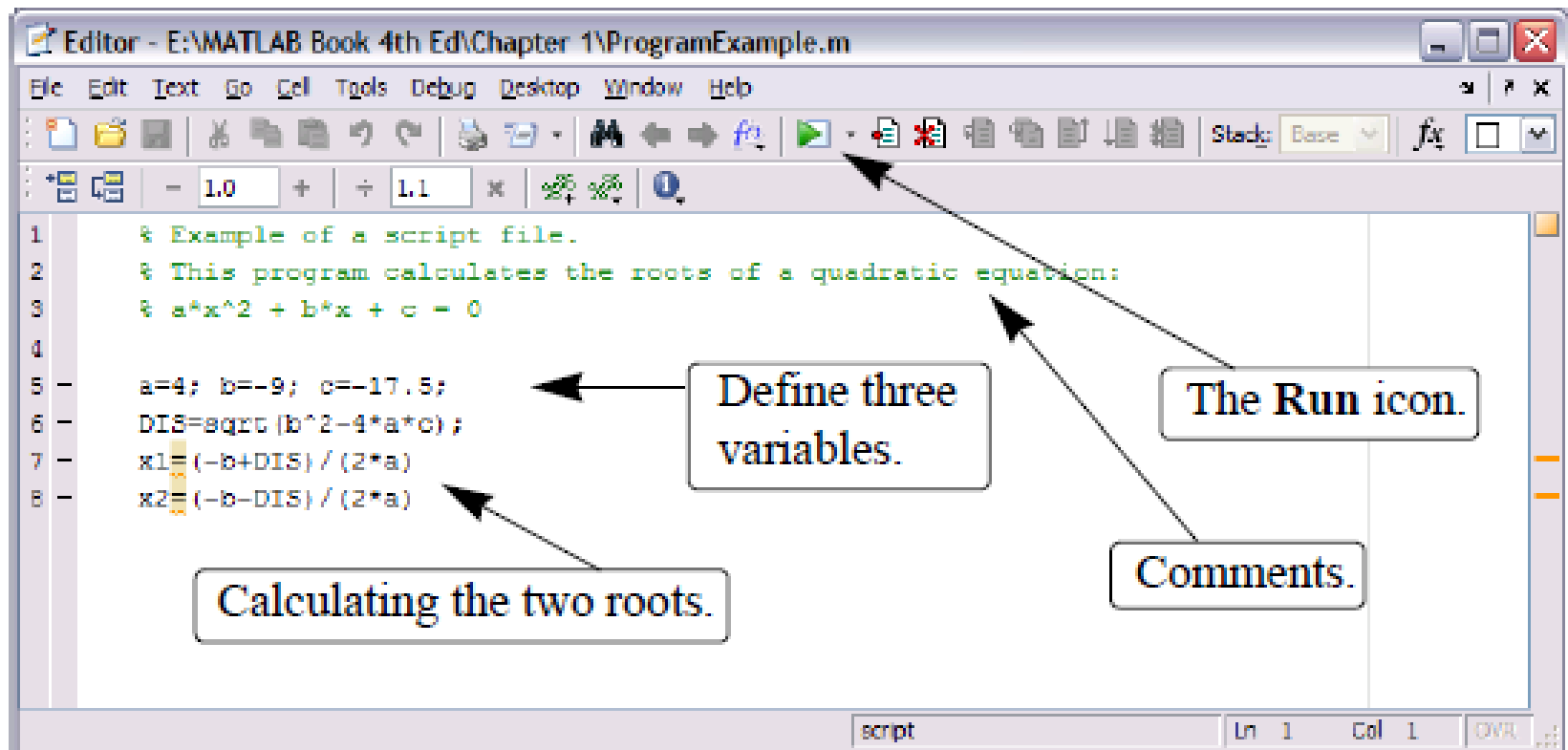
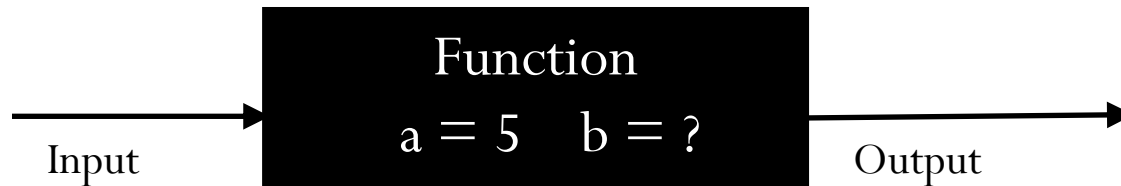


Figure 1-7: A program typed in the Editor/Debugger Window.

4. Scripts e Funções no MATLAB



Calling Program $a = ?$ $b = 9$

```
function [output arguments] = function_name(input arguments)
```

The word `function` must be the first word, and must be typed in lower-case letters.

A list of output arguments typed inside brackets.

The name of the function.

A list of input arguments typed inside parentheses.

Estrutura de uma Função Típica

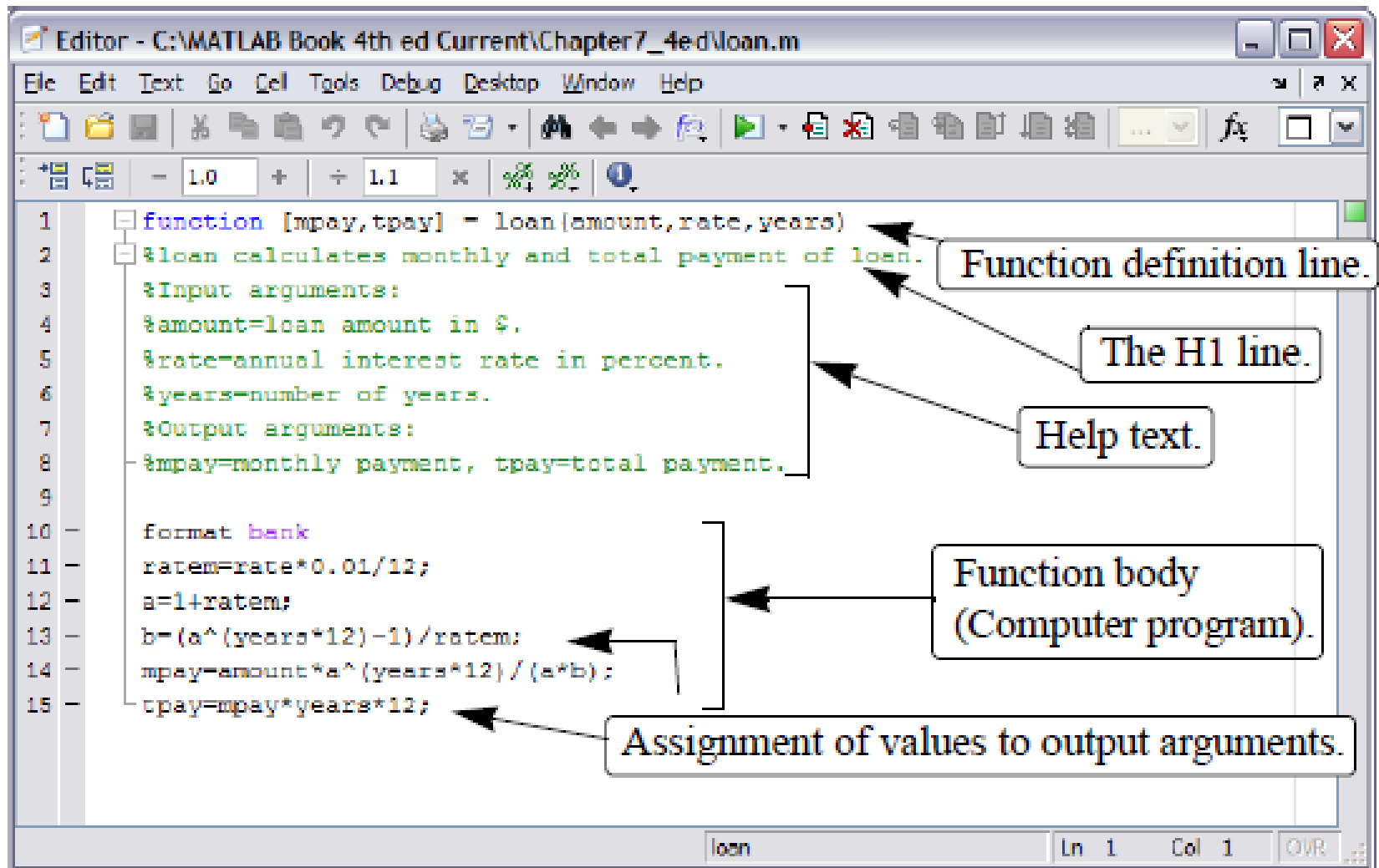


Figure 7-2: Structure of a typical function file.

5. Estruturas de Controle de Fluxo

Operadores Relacionais e
Operadores Lógicos

<u>Relational Operator</u>	<u>Description</u>
<	Less than.
>	Greater than.
<=	Less than or equal to.
>=	Greater than or equal to.
==	Equal to.
~=	Not Equal to.

INPUT		OUTPUT				
A	B	AND A&B	OR A B	XOR (A,B)	NOT ~A	NOT ~B
false	false	false	false	false	true	true
false	true	false	true	true	true	false
true	false	false	true	true	false	true
true	true	true	true	false	false	false

5. Estruturas de Controle de Fluxo

Estruturas Condicionais

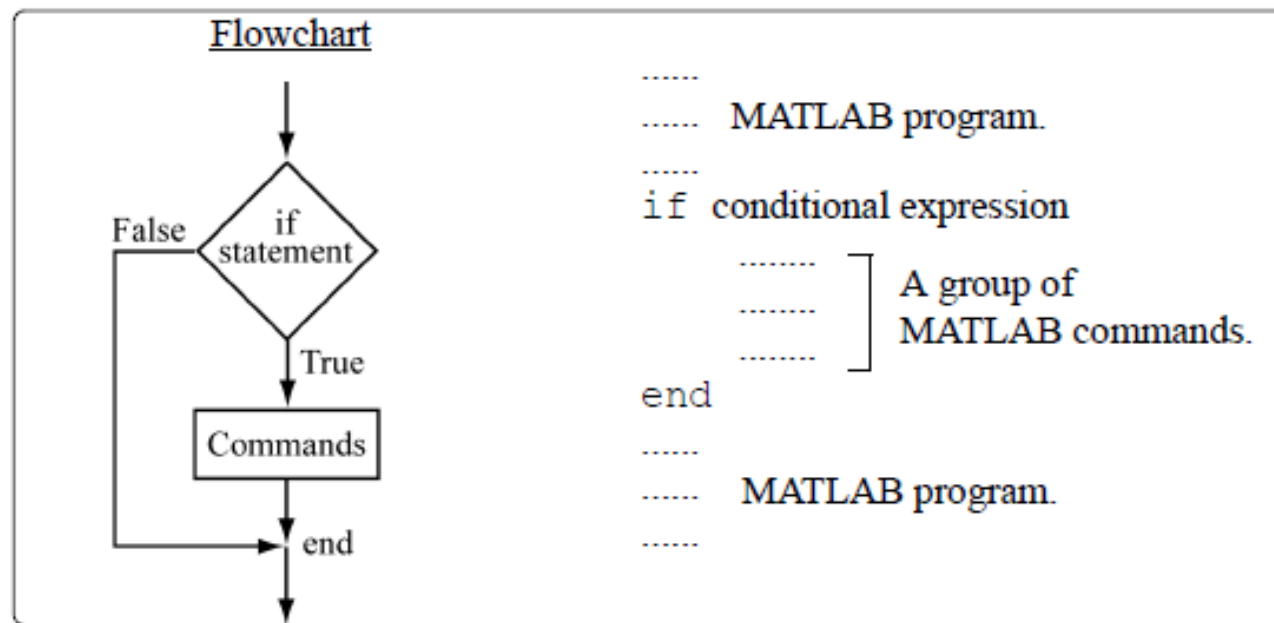


Figure 6-1: The structure of the `if-end` conditional statement.

5. Estruturas de Controle de Fluxo

Estruturas Condicionais

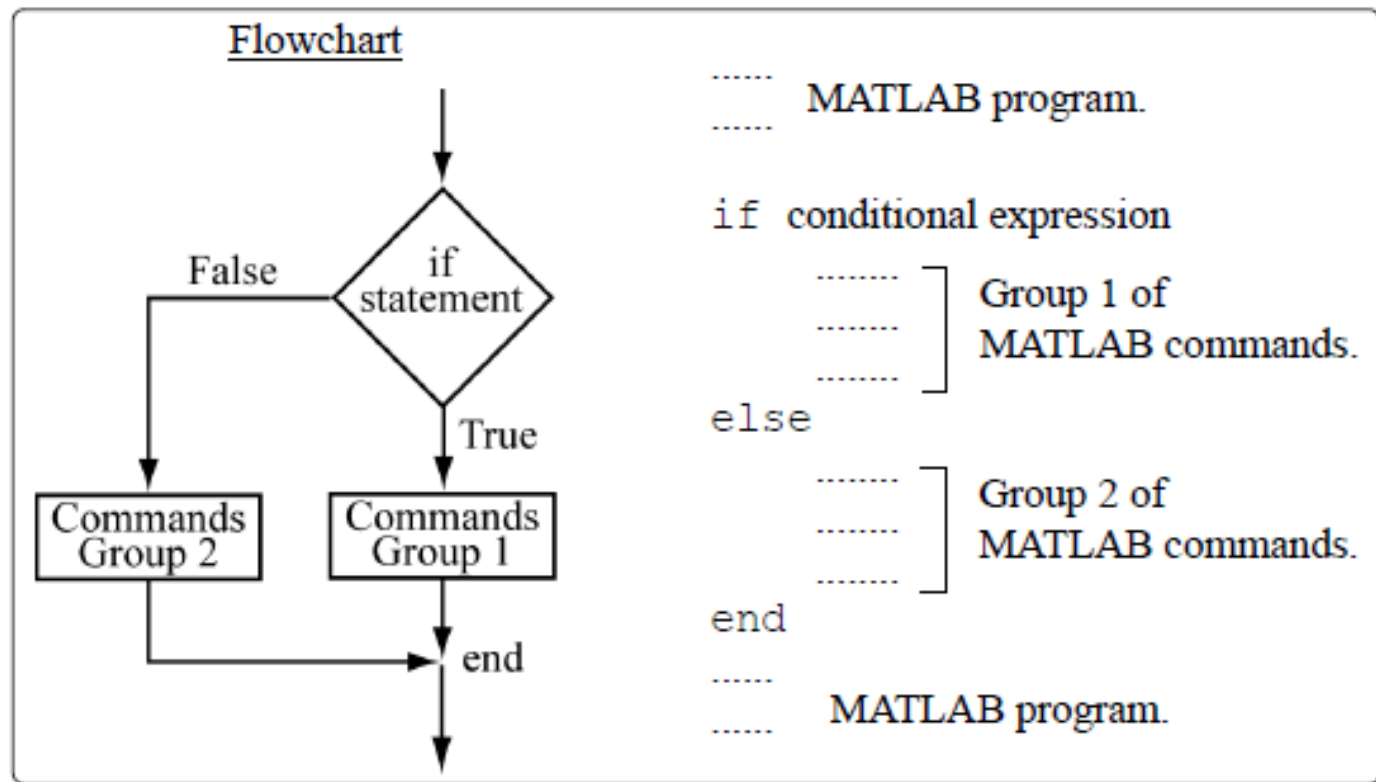


Figure 6-2: The structure of the if-else-end conditional statement.

5. Estruturas de Controle de Fluxo

Estruturas Condicionais

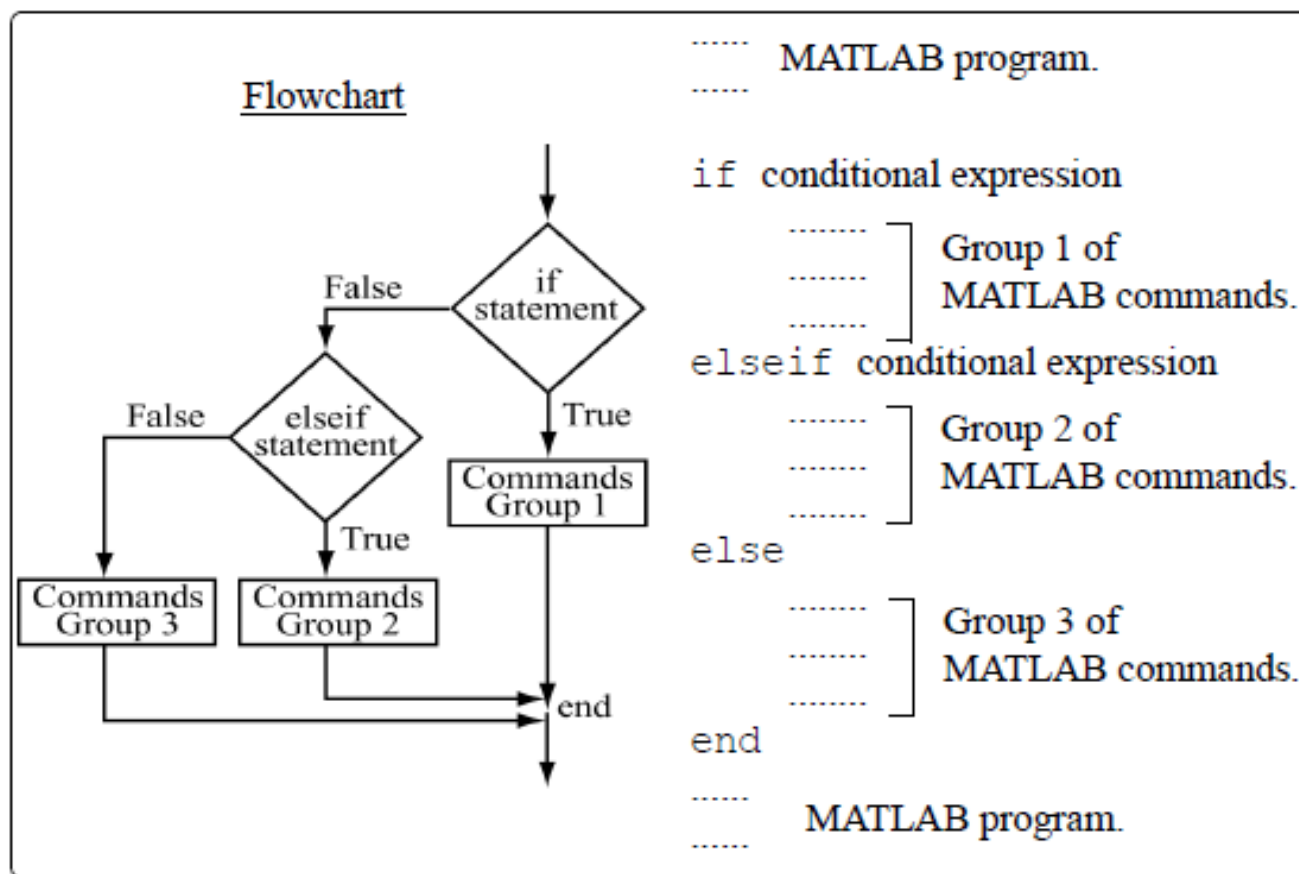


Figure 6-3: The structure of the if-elseif-else-end conditional statement.

5. Estruturas de Controle de Fluxo

O Comando *Switch*

```

..... MATLAB program.
.....

switch switch expression
    case value1
        ..... ] Group 1 of commands.
        ..... ]
    case value2
        ..... ] Group 2 of commands.
        ..... ]
    case value3
        ..... ] Group 3 of commands.
        ..... ]
    otherwise
        ..... ] Group 4 of commands.
        ..... ]
end
..... MATLAB program.
.....
    
```

Figure 6-4: The structure of a switch-case statement.

5. Estruturas de Controle de Fluxo

Estruturas de Repetição

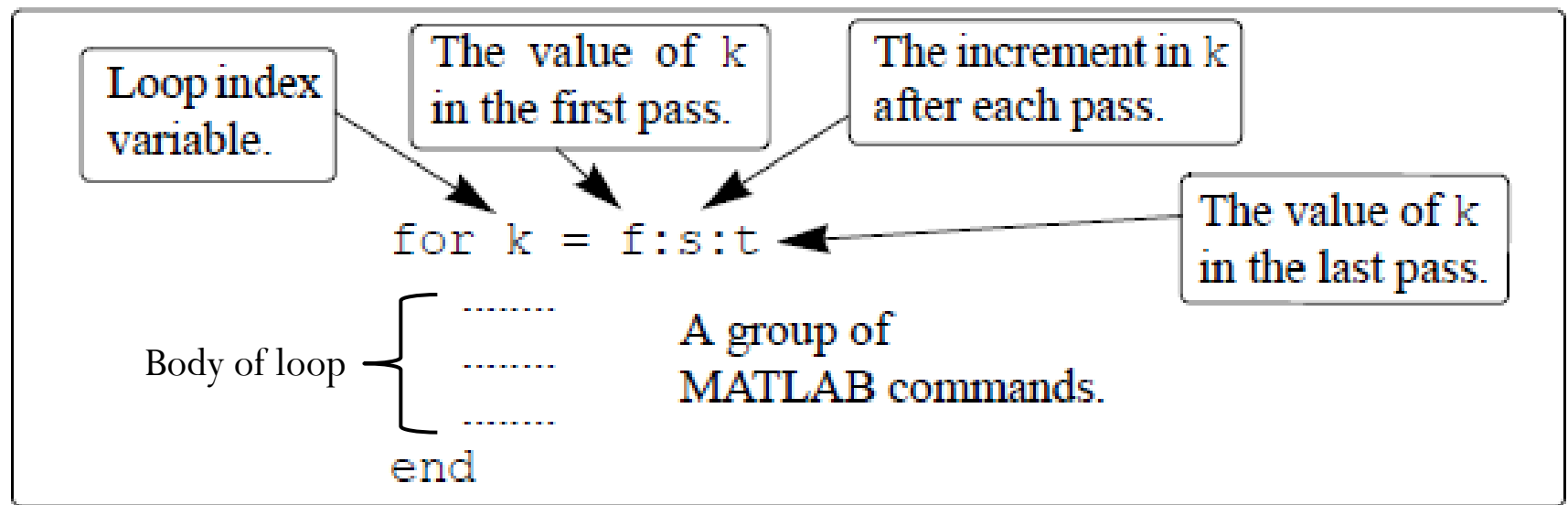


Figure 6-5: The structure of a `for-end` loop.

5. Estruturas de Controle de Fluxo

Estruturas de Repetição

Exemplo

```
for k=1:3:10
    k
    x = k^2
end
fprintf('After loop k = %d\n', k);
```

Output

k = 1

x = 1

k = 4

x = 16

k = 7

x = 49

k = 10

x = 100

After loop k = 10

5. Estruturas de Controle de Fluxo

Estruturas de Repetição

```

while conditional expression
    .....
    .....
    .....
end
    
```

A group of
MATLAB commands.

Figure 6-6: The structure of a `while`-`end` loop.

Típico quando não se sabe o número de iterações do loop

5. Estruturas de Controle de Fluxo

Estruturas de Repetição

Exemplo

```
x = 1
while x <= 15
    x = 2*x
end
```

Output

x =
1

x =
2

x =
4

x =
8

x =
16

5. Estruturas de Controle de Fluxo

Interprete o código
ao lado

```
n=input('Enter the number of rows ');
m=input('Enter the number of columns ');
A=[];
for k=1:n
    for h=1:m
        if k==1
            A(k,h)=h;
        elseif h==1
            A(k,h)=k;
        else
            A(k,h)=A(k,h-1)+A(k-1,h);
        end
    end
end
A
```

Define an empty matrix A

Start of the first for-end loop.

Start of the second for-end loop.

Start of the conditional statement.

Assign values to the elements of the first row.

Assign values to the elements of the first column.

Assign values to other elements.

end of the if statement.

end of the nested for-end loop.

end of the first for-end loop.

The program is executed in the Command Window to create a 4×5 matrix.

```
>> Chap6_exp8
Enter the number of rows 4
Enter the number of columns 5
```

5. Estruturas de Controle de Fluxo

O Comando break

Exemplo

Script

```
while(1)    Trick – "1" is always true so it makes loop iterate forever!
    name = input( 'Type name or q to quit: ', 's' );
    if length( name ) == 1 && name(1) == 'q' If user entered only one
        break; Only way to exit loop!           letter and it is a "q",
                                                jump out of loop
    else
        fprintf( 'Your name is %s\n', name ); Otherwise print name
    end
end
```

6. Gráficos

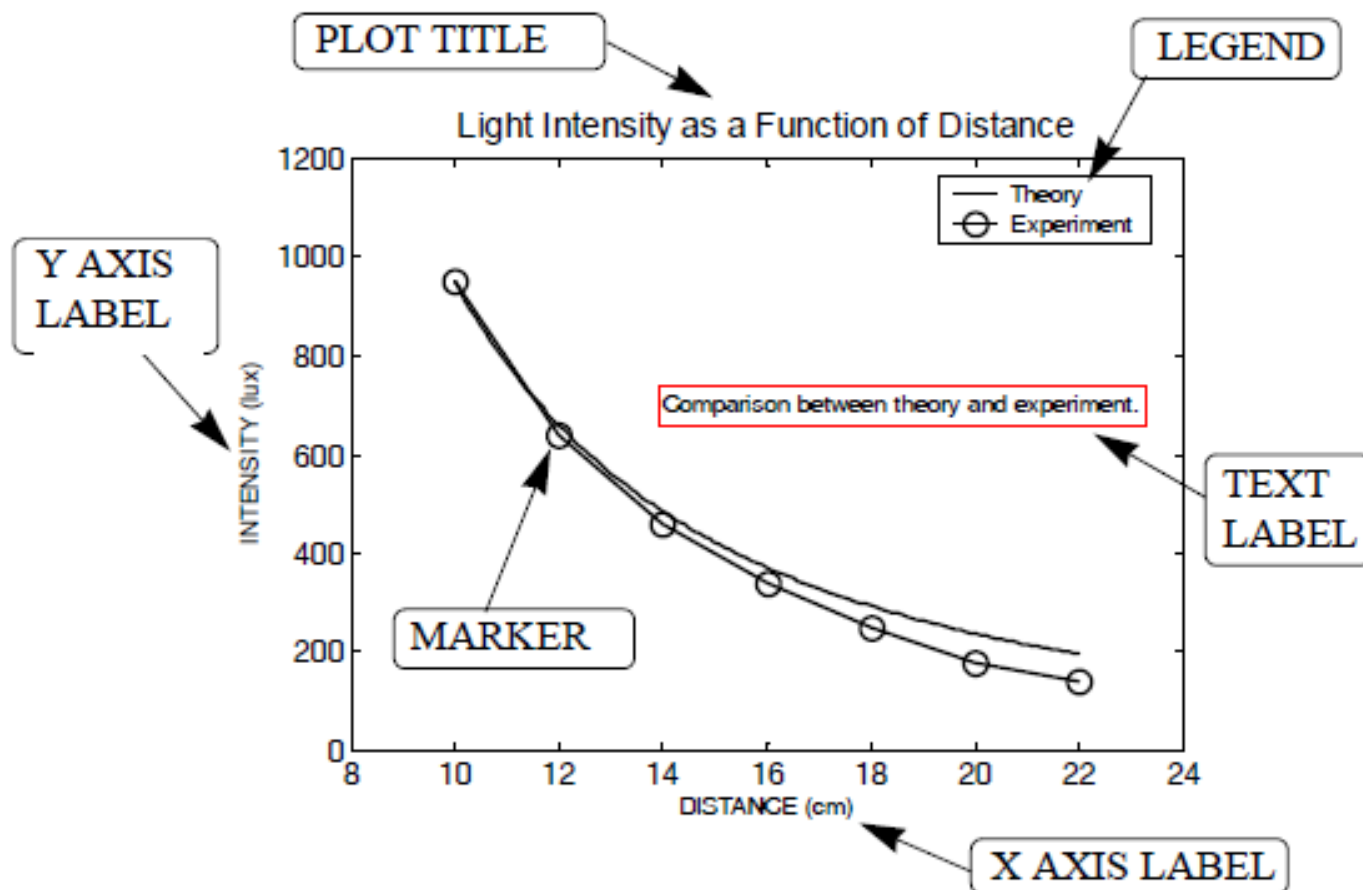


Figure 5-1: Example of a formatted two-dimensional plot.

6. Gráficos

Plots em 2D

```
plot(x,y, 'line specifiers', 'PropertyName', PropertyValue)
```

Vector

Vector

(Optional) Specifiers that define the type and color of the line and markers.

(Optional) Properties with values that can be used to specify the line width, and marker's size and edge, and fill colors.

Plots em 2D

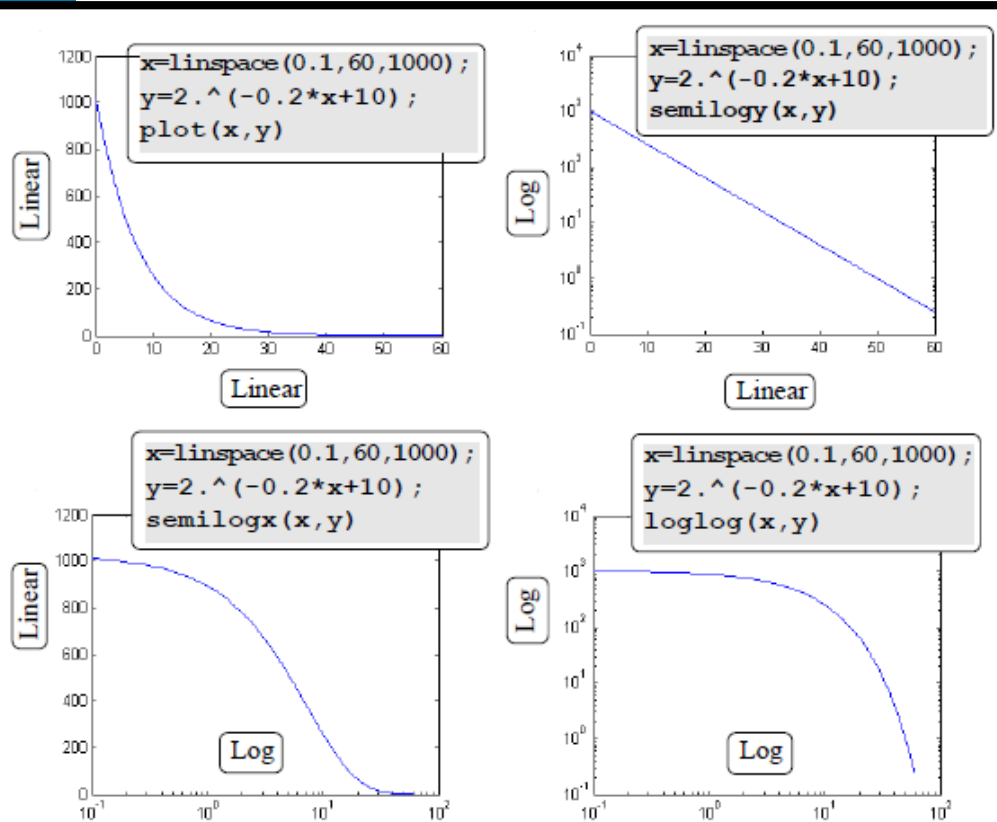


Figure 5-9: Plots of $y = 2^{(-0.2x+10)}$ with linear, semilog, and loglog scales.

LETRA	CORES
y	amarelo
m	magenta
c	cyan
r	encarnado
g	verde
b	azul
w	branco
k	preto
SÍMBOLO	TIPO DE LINHA
-	linha sólida
:	ponteada
- .	traço - ponto
--	traço interrompido
SÍMBOLO	MARKERS
.	ponto
o	círculo
x	x-mark
+	sinal +
*	Estrela

6. Gráficos

Exercício: Qual a funcionalidade das funções abaixo?

- xlabel
- ylabel
- legend
- grid
- hold
- Title
- Text
- axis
- title

Plots em 3D

Rode o programa abaixo para testar o comando *mesh*

```
» x = -7.5:.5:7.5;

» y = x;

» [X,Y] = meshgrid(x,y);

» R = sqrt(X.^2 + Y.^2) + eps;

» Z = sin(R)./R;

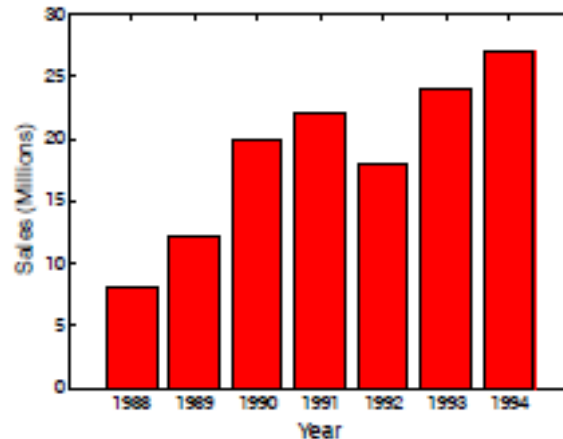
» mesh(X,Y,Z)
```

Gráficos Especiais

Vertical Bar Plot

Function format:

`bar(x, y)`

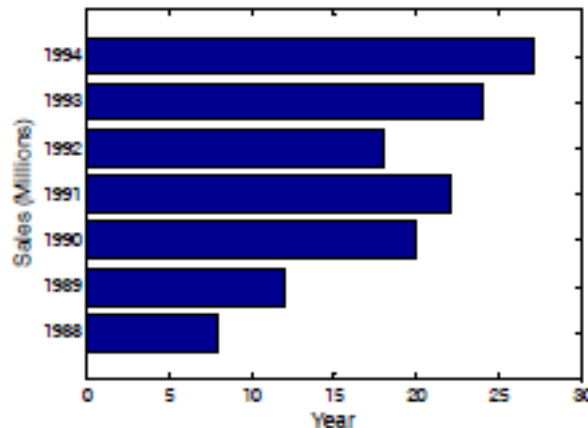


```
yr=[1988:1994];
sle=[8 12 20 22 18 24 27];
bar(yr,sle,'r') ← The bars are in red.
xlabel('Year')
ylabel('Sales (Millions)')
```

Horizontal Bar Plot

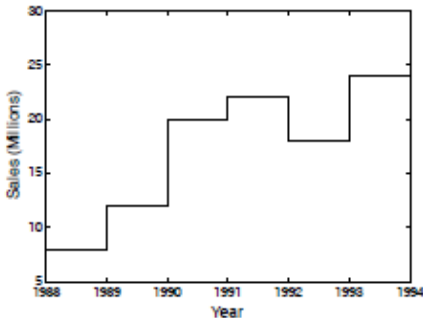
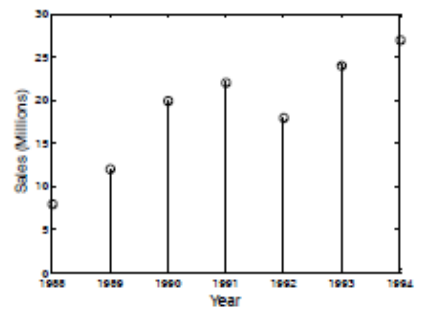
Function format:

`barh(x, y)`



```
yr=[1988:1994];
sle=[8 12 20 22 18 24 27];
barh(yr,sle)
xlabel('Sales (Millions)')
ylabel('Year')
```


Gráficos Especiais

<p><u>Stairs Plot</u></p> <p>Function format:</p> <p><code>stairs(x,y)</code></p>		<pre>yr=[1988:1994]; sle=[8 12 20 22 18 24 27]; stairs(yr,sle)</pre>
<p><u>Stem Plot</u></p> <p>Function Format</p> <p><code>stem(x,y)</code></p>		<pre>yr=[1988:1994]; sle=[8 12 20 22 18 24 27]; stem(yr,sle)</pre>

```
t=linspace(0,2*pi,200);
r=3*cos(0.5*t).^2+t;
polar(t,r)
```

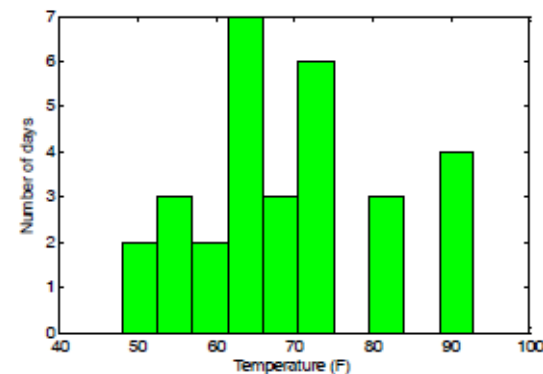
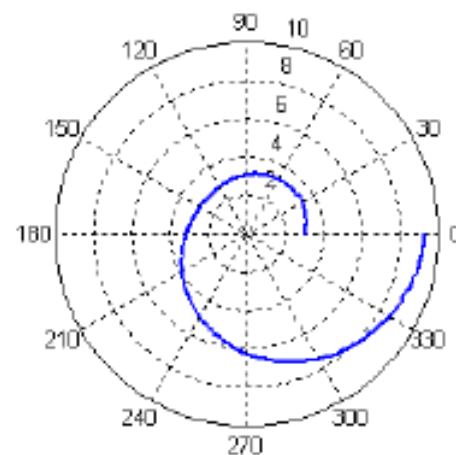


Figure 5-11: Histogram of temperature data.



7. Exemplos de Aplicação

Analizador de Espectro

```
% =====
% Um exemplo de aplicação da transformada de Fourier
% =====

% Inicialização
clc, clear all, close all

% Dados de amostragem
ts = 0.001;
fs = 1/ts;
% t = [0:ts:10];
t = 0:ts:2;

% Espectro de potência usando a função FFT_pot2
[X,x1,f,df] = FFT_pot2(x,ts);

figure, plot(f,10*log10((fftshift(abs(X)))));

% ----- Potência e Psd do Sinal -----
% Potência
p = 10*log10((norm(x)^2)/length(x))
h = spectrum.welch;
% figure, msspectrum(h,x,'Fs',fs,'NFFT',length(x1))
figure,
msspectrum(h,x,'Fs',fs,'NFFT',length(x1),'SpectrumType',
'TwoSided')

% Sinal no domínio do tempo
% x = cos(2*pi*47*t) + cos(2*pi*219*t);
x = 3*cos(2*pi*t*200);
```

7. Exemplos de Aplicação

```
% =====
function [Sinal_ff,sinal_tf,f,df] = FFT_pot2(sinal,ts)
%
% FFT_pot2 --> Gera a transformada de Fourier de um sinal de tempo discreto
%           A sequencia é preenchida com zeros para determinar a resolução
%           em frequencia final df e o o novo sinal é sinal_tf.
% =====
%
fs = 1/ts;           % Taxa de amostragem
ni = length(sinal);  % Tamanho do sinal de entrada
nf = 2^(nextpow2(ni)); % Novo tamanho do sinal

% A transformada via FFT
Sinal_ff = fft(sinal,nf);
Sinal_ff = Sinal_ff/fs; % Qual a razão disto?

% O novo sinal no domínio do tempo
sinal_tf = [sinal,zeros(1,nf-ni)];

% A resolução na frequencia
df = fs/nf;

% Vetor frequencia
f = (0:df:df*(length(sinal_tf)-1)) - fs/2;
```

TPC: Um teste

```
% sound
fs = 1000;
ts = 0:1/fs:2;
f = 250 + 240*sin(2*pi*ts);
x = sin(2*pi*f.*ts);
strips(x,0.25,fs)
sound(x,fs)
figure, plot(ts,x)
figure, plot(ts(1:200),x(1:200))
```