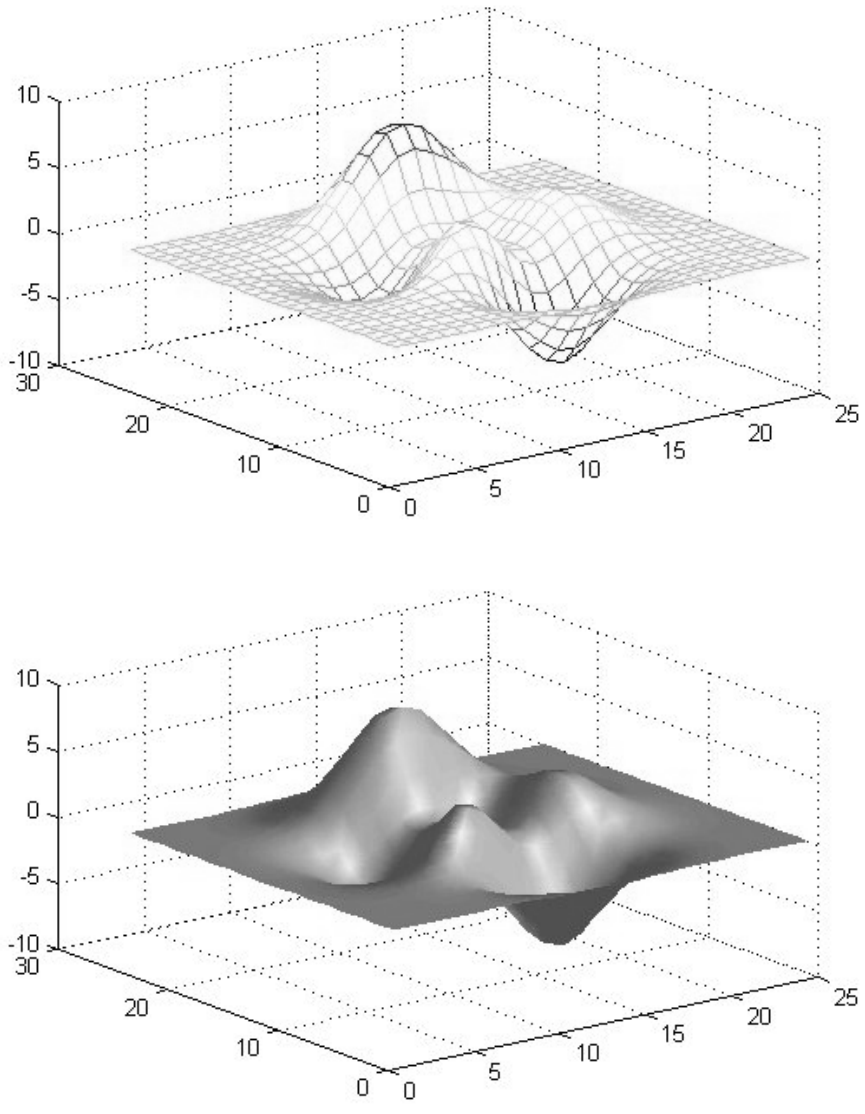
UNIVERSIDADE CATÓLICA DE PELOTAS
ESCOLA DE ENGENHARIA E ARQUITETURA
CURSO DE ENGENHARIA ELÉTRICA E ELETRÔNICA

# INTRODUÇÃO AO MATLAB ®




POR:

Prof. Delmar Broglio Carvalho, M. Eng.

# PREFÁCIO

## O que é o MATLAB?

MATLAB é um "software" interativo de alta performance voltado para o cálculo numérico. O MATLAB integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos em ambiente fácil de usar onde problemas e soluções são expressos somente como eles são escritos matematicamente, ao contrário da programação tradicional.

O MATLAB é um sistema interativo cujo elemento básico de informação é uma matriz que não requer dimensionamento. Esse sistema permite a resolução de muitos problemas numéricos em apenas uma fração do tempo que se gastaria para escrever um programa semelhante em linguagem Fortran, Basic ou C. Além disso, as soluções dos problemas são expressas no MATLAB quase exatamente como elas são escritas matematicamente.

## Carregando o MATLAB

No Gerenciador de Programas do Microsoft Windows deve-se abrir o grupo de programas do MATLAB for Windows, que contém o ícone do aplicativo MATLAB. Um duplo clique no ícone MATLAB carrega o aplicativo MATLAB.

Quando o MATLAB é carregado, uma janela é exibida: a Janela de Comando (*Command Window*) e para a visualização de gráficos uma segunda janela é ativada, a Janela Gráfica (*Graphic Window*). A Janela de Comando é ativada quando se inicializa o MATLAB, e o "prompt" padrão (>>) é exibido na tela.

A partir desse ponto, o MATLAB espera as instruções do usuário. Por exemplo, a entrada de uma matriz poderia ser feita pela digitação da seguinte seqüência de dados:

```
>> A = [1 2 3; 4 5 6; 7 8 9]
```

onde os colchetes denotam a entrada de uma matriz e o ponto e vírgula um separador de dados, indicando uma nova linha da matriz. Quando se pressiona a tecla <enter> o MATLAB responde com

```
A =
     1     2     3
     4     5     6
     7     8     9
```

Para obter a matriz inversa usa-se

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

e o MATLAB responde com o resultado para a matriz inversa.

**Editor de Linhas de Comando**

As teclas com setas podem ser usadas para encontrar comandos digitados anteriormente, para execução novamente ou sua reedição. Por exemplo, suponha que você entre com

```
>> log (sqt(tan(pi/5)))
```

Como para calcular a raiz quadrada o comando certo é **sqrt**, o MATLAB responde com uma mensagem de erro:

```
??? Undefined funcion or variable sqt.
```

Ao invés de reescrever a linha inteira, simplesmente pressione a tecla "seta para cima". O comando errado retorna, e você pode, então, mover o cursor para trás usando a tecla "seta para esquerda" ou o ponto de inserção com o "mouse" ao lugar apropriado para inserir a letra "r". Então, o comando retorna a resposta apropriada:

```
>> log (sqrt(tan(pi/5)))
ans =
        -0.1597
```

Além das teclas com setas, pode-se usar outras teclas para reeditar a linha de comando. A seguir é dada uma breve descrição destas teclas:

| | |
|---|---|
| ↑ | retorna a linha anterior |
| ↓ | retorna a linha posterior |
| ← | move um espaço para a esquerda |
| → | move um espaço para a direita |
| **Ctrl** ← | move uma palavra para a esquerda |
| **Ctrl** → | move uma palavra para a direita |
| **Home** | move para o começo da linha |
| **End** | move para o final da linha |
| **Del** | apaga um caracter a direita |
| **Backspace** | apaga um caracter a esquerda |

# 1. INTRODUÇÃO

O MATLAB trabalha essencialmente com um tipo de objeto, uma matriz numérica retangular podendo conter elementos complexos (deve-se lembrar que um escalar é uma matriz de dimensão l x l e que um vetor é uma matriz que possui somente uma linha ou uma coluna).

## 1.1 Entrando com Matrizes Simples

As matrizes podem ser introduzidas no MATLAB por diferentes caminhos:

- digitadas na Janela de Comando (lista explícita de elementos),
- geradas por comandos e funções,
- criadas em arquivos ".m",
- carregadas a partir de um arquivo de dados externo.

O método mais fácil de entrar com pequenas matrizes no MATLAB é usando uma lista explícita. Os elementos de cada linha da matriz são separados por espaços em branco ou vírgulas e as colunas separadas por ponto e vírgula, colocando-se colchetes em volta do grupo de elementos que formam a matriz. Por exemplo, entre com a expressão

```
>> A=[ 1 2 3;4 5 6;7 8 9 ]
```

Pressionando <enter> o MATLAB mostra o resultado

```
A=
     l     2     3
     4     5     6
     7     8     9
```

A matriz **A** é salva na memória RAM do computador, ficando armazenada para uso posterior.

As matrizes podem, também, ser introduzidas linha a linha, o que é indicado para matrizes de grande dimensão. Por exemplo:

```
>>A =     [1 2 3
>>         4 5 6
>>         7 8 9]
```

Outra maneira para entrar com matrizes no MATLAB é através de um arquivo no formato texto com extensão ".m". Por exemplo, se um arquivo chamado "gera.m" contém estas três linhas de texto,

```
A=   [1 2 3
      4 5 6
      7 8 9]
```

então a expressão "gera" lê o arquivo e introduz a matriz A.

```
>>gera
```

O comando **load** pode ler matrizes geradas pelo MATLAB e armazenadas em arquivos binários ou matrizes geradas por outros programas armazenadas em arquivos ASCII.

## 1.2 Elementos das Matrizes

Os elementos de uma matriz podem ser qualquer expressão do MATLAB, por exemplo.

```
>> x = [-pi sqrt(2) ((1+2+3)*4/5)^2]
```

resulta em

```
x =
    -3.1416    1.4142   23.0400
```

Um elemento individual da matriz pode ser reverenciado como um índice entre parênteses. Continuando o exemplo,

```
>> x(6) = abs(x(l))
```

produz:

```
x =
    -3.1416   1.4142   23.0400   0   0   3.1416
```

Note que a dimensão do vetor x é modificada automaticamente para acomodar o novo elemento e que os elementos do intervalo indefinido são estabelecidos como zero. Outra informação aqui mostrada é em relação ao número “pi” o qual já é definido pelo sistema

Grandes matrizes podem ser construídas a partir de pequenas matrizes. Por exemplo, pode-se anexar outra linha na matriz **A** usando,

```
>> r= [10 11 12];
>> A= [A;r]
```

que resulta em

**A=**

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

Note que o vetor **r** não foi listado porque ao seu final foi acrescentado ";".

Pequenas, matrizes podem ser extraídas de grandes matrizes usando ":". Por exemplo,

```
>> A = A(1:3,:);
```

seleciona as três primeiras linhas e todas as colunas da matriz **A** atual, modificando-a para sua forma original.

## 1.3 Declarações e Variáveis

O MATLAB é uma linguagem de expressões. As expressões usadas são interpretadas e avaliadas pelo sistema. As declarações no MATLAB são freqüentemente da forma

```
>> variável = expressão
```

ou simplesmente

```
>> expressão
```

As expressões são compostas de operadores e outros caracteres especiais, de funções e dos nomes das variáveis. A avaliação das expressões produzem matrizes, que são então mostradas na tela e atribuídas às variáveis para uso futuro. Se o nome da variável e o sinal de igualdade "=" são omitidos, a variável com o nome **ans**, que representa a palavra "answer" (resposta), é automaticamente criada pelo sistema. Por exemplo, digite a expressão

```
>> 1900/81
```

que produz

```
ans=
    23.4568
```

Se o último caractere da declaração é um ponto e vírgula, ";", a impressão na tela é suprimida, mas a tarefa é realizada. Esse procedimento é usado em arquivos com extensão ".m" e em situações onde o resultado é uma matriz de grandes dimensões e temos interesse em apenas alguns dos seus elementos.

Se a expressão é tão grande que não cabe em apenas uma linha, pode-se continuar a expressão na próxima linha usando um espaço em branco e três pontos, "...", ao final das linhas incompletas. Por exemplo,

```
>> s = l - 1/2 + 1/3 - 1/4 + 1/5 - 1/6 + 1/7 ...
>>     - 1/8 + 1/9 - l/10 + 1/11 - 1/12 + 1/13;
```

calcula o resultado da série, atribuindo a somatória à variável **s**, mas não imprime o resultado na tela. Note que os espaços em branco entre os sinais "=", "+" e "-" são opcionais, mas o espaço em branco entre "1/7" e "..." é obrigatório.

As variáveis e funções podem ser formadas por um conjunto de letras, ou por um conjunto de letras e números, onde somente os primeiros 19 caracteres do conjunto são identificados. O MATLAB faz distinção entre letras maiúsculas e minúsculas, assim **a** e **A** não são as mesmas variáveis. Todas as funções devem ser escritas em letras minúsculas: **inv(A)** calcula a inversa de **A**, mas **INV(A)** é uma função indefinida.

## 1.4 Obtendo Informações da Área de Trabalho

Os exemplos de declarações mostrados nos itens acima criaram variáveis que são armazenadas na *Área de Trabalho* do MATLAB. Executando

```
>> who
```

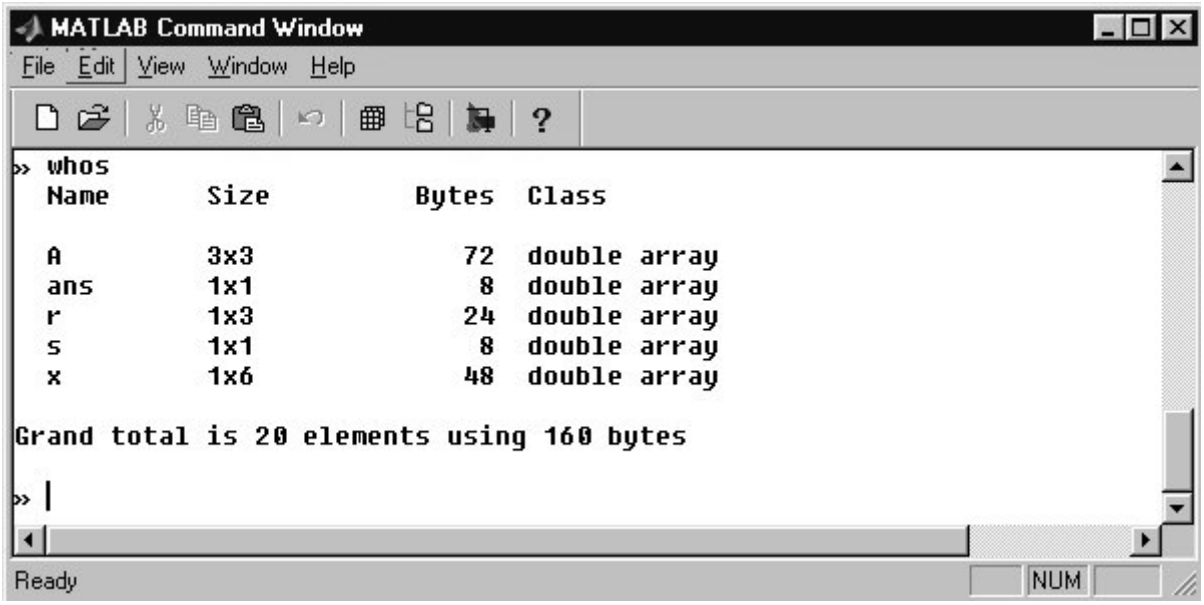
obtêm-se uma lista das variáveis armazenadas na Área de Trabalho:

```
Your variables are:

A    ans   r    s    x
```

Que mostra as cinco variáveis geradas em nossos exemplos, incluindo **ans**.

Uma informação mais detalhada mostrando a dimensão de cada uma das variáveis correntes é obtido com **whos** que para nosso exemplo produz:




Cada elemento de uma matriz real requer 8 bytes de memória, assim nossa matriz **A** de dimensão 3x3 usa 72 bytes e todas variáveis utilizadas necessitam de um total de 160 bytes.

## 1.5 Números e Expressões Aritméticas

A notação decimal convencional, com ponto decimal opcional e o sinal de menos, é usada para números. A potência de dez pode ser incluída como um sufixo. A seguir são mostrados alguns exemplos de números aceitos:

| | | |
|---|---|---|
| 3 | -99 | 0.00001 |
| 9.637458638 | 1.602E-20 | 6.06375e23 |

As expressões podem ser construídas usando os operadores aritméticos usuais e as regras de precedência:

| | | |
|---|---|---|
| 1 | ^ | Exponenciação |
| 2 | / | divisão a direita |
| 2 | \ | divisão a esquerda |
| 3 | * | Multiplicação |
| 3 | + | Adição |
| 4 | - | Subtração |

Deve-se notar que existem dois símbolos para divisão: as expressões 1/4 e 4\1 possuem o mesmo valor numérico, isto é, 0.25. Parênteses são usados em sua forma padrão para alterar o mesmo a precedência usual dos operadores aritméticos.

## 1.6 Números e Matrizes Complexas

Números complexos são permitidos em todas operações e funções no MATLAB. Os números complexos são introduzidos usando-se as funções especiais **i** e **j**. Por exemplo

```
>> z= 3 + 4*i
```

ou

```
>> z= 3 +4*j
```

os quais produzem o mesmo número

```
z= 3.0000 +4.0000*i
```

Outro exemplo é:

```
>> w= r * exp(i*theta)
```

As seguintes declarações mostram dois caminhos convenientes para se introduzir matrizes complexas no MATLAB:

```
>> A= [1 2; 3 4]+i*[5 6;7 8]
```

e

```
>> A= [1+5*i 2+6*i; 3+7*i 4+8*i]
```

que produzem o mesmo resultado.

Se **i** ou **j** forem usados como variáveis, de forma que tenham seus valores originais modificados, uma nova unidade complexa deverá ser criada e utilizada de maneira usual:

```
>> ii = sqrt(-1);
>> z = 3 + 4*ii
```

Mesmo assim, a exibição do número complexo será feita com o sufixo **i**.

## 1.7 Formato de Saída

O formato numérico exibido na tela pode ser modificado utilizando-se o comando **format**, que afeta somente o modo como os dados serão exibidos, e não como eles são computados ou salvos (o MATLAB efetua todas operações em dupla precisão).

Se todos os elementos das matrizes são inteiros exatos, a matrizes é mostrada em um formato sem qualquer ponto decimal. Por exemplo,

```
>> x = [-1 0 1]
```

sempre resulta em

```
x =
     -1  0  1
```

Se pelo menos um dos elementos da matriz não é inteiro exato, existem várias possibilidades de formatar a saída. O formato "default", chamado de formato **short**, mostra aproximadamente 5 dígitos significativos ou usam notação científica. Por exemplo a expressão

```
>> x = [4/3 1.2345e-6]
```

é mostrada , para cada formato usado, da seguinte maneira:

| | |
|---|---|
| format short | 1.3333 0.0000 |
| format short e | 1.3333e+000 1.2345e-006 |
| format long | 1.33333333333333 0.000000123450000 |
| format long e | 1.333333333333333e+000 1.234500000000000e-006 |
| format hex | 3ff5555555555555 3eb4b6231abfd271 |
| format rat | 4/3 1/810045 |
| format bank | 1.33 0.00 |
| format + | ++ |

Com o formato **short** e **long**, se o maior elemento da matriz é maior que 1000 ou menor que 0.001, um fator de escala comum é aplicado para que a matriz completa seja mostrada. Por exemplo,

```
>> x = 1e20*x
```

resultado da multiplicação será mostrado na tela.

```
x =
     l.0e+20 *
       1.3333 0.0000
```

O formato + é uma maneira compacta de mostrar matrizes de grandes

dimensões. Os símbolos "+", "-", e "espaço em branco" são mostrados, respectivamente para elementos positivos, elementos negativos e zeros.

## 1.8 As Facilidades do HELP (Ajuda)

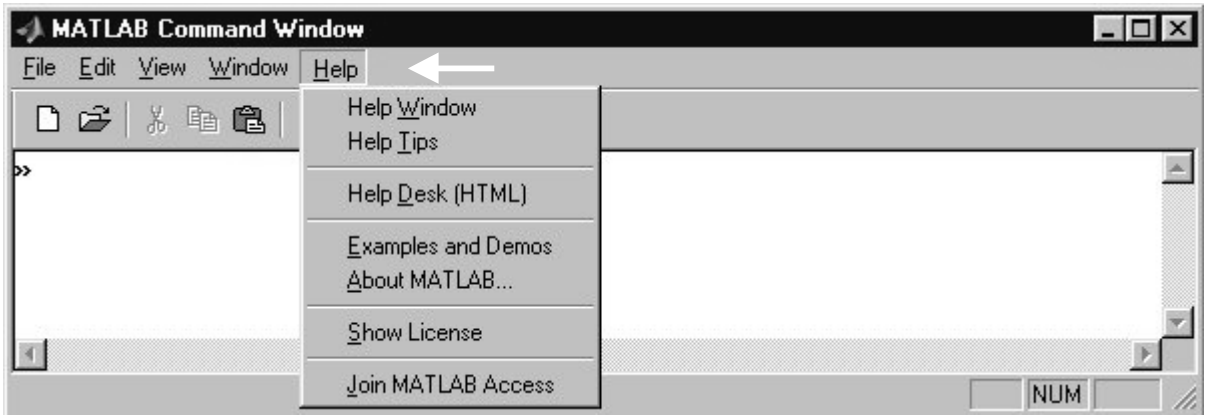
O MATLAB possui um comando de ajuda **(help)** que fornece informações sobre a maior parte dos tópicos. Digitando

```
>> help
```

obtêm-se uma lista desses tópicos disponíveis:

```
HELP topics:
c:\matlab                - Establish MATLAB session parameters.
Matlab\general           - General purpose commands.
Matlab\ops               - Operators and special characters.
Matlab\lang              - Programming language constructs.
Matlab\elmat             - Elementary matrices and matrix manipulation
Matlab\elfun             - Elementary math functions
Matlab\specfun           - Specialized math functions
Matlab\specmat           - Specialized matrices
Matlab\matfun            - Matrix functions - numerical linear algebra.
Matlab\datafun           - Data analysis and Fourier transform.
Matlab\polyfun           - Polynomial and interpolation functions
Matlab\funfun            - Function functions and ODE solvers
Matlab\sparfun           - Sparse matrix functions.
Matlab\graph2d           - Two dimensional graphs.
Matlab\graph3d           - Three dimensional graphs
Matlab\specgraph         - Specialized graphs.
Matlab\graphics          - General purpose graphics functions.
Matlab\uitools           - Graphical user interface tools.
Matlab\strfun            - Character strings functions
Matlab\iofun             - File input/output.
Matlab\timefun           - Time and dates.
Matlab\demos             - Examples and demonstrations.
Simulink\simulink        - SIMULINK model analysis.
Simulink\ blocks         - SIMULINK block library
Simulink\simdemos.       - SIMULINK demonstrations and samples
nnet\ examples           - Neural Network Toolbox examples.
nnet\nnet                - Neural Network Toolbox.
        ⋮                        ⋮
        ⋮                        ⋮
```

Sendo aqui listados apenas algumas das ferramentas básicas. Para obter mais informações, nada mais conveniente do que recorrer ao próprio sistema de ajuda do programa, como ilustrado abaixo:

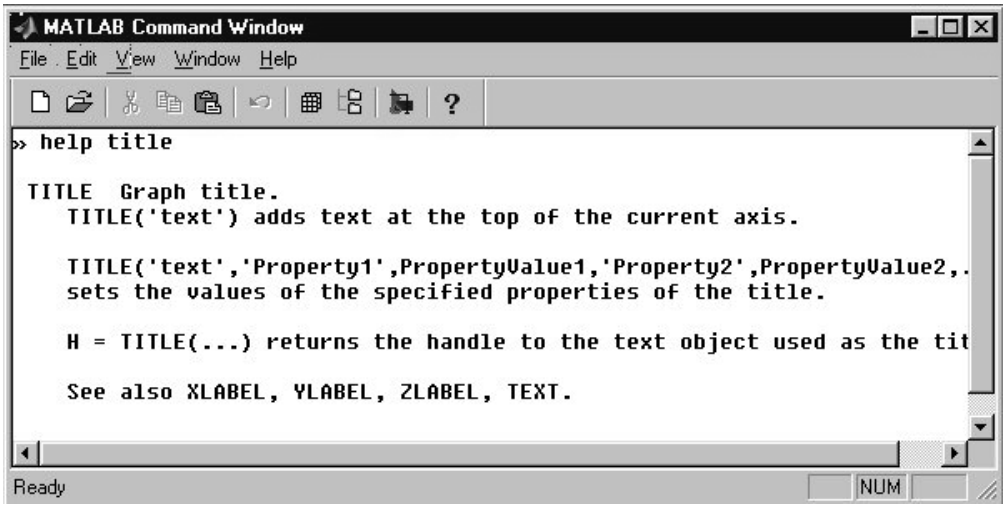



Acessando a opção "Help Window" podemos visualizar as opções disponíveis e acessando com um duplo "click" o item, temos a listagem de todos os comandos associados com o tópico.

Ainda na janela de comando, para obter informações sobre um comando específico, por exemplo **title**, digite:

```
>> help title
```

e informações mais detalhadas sobre este comando serão exibidas:




Note que no exemplo mostrado para, adicionar o título a um gráfico, **TITLE ('TEXT')** está escrito em letras maiúsculas somente para destacar. Devemos lembrar que todos os comandos do MATLAB devem ser escritos em letras minúsculas, portanto, para adicionar o texto "*Título do Gráfico*" em um gráfico, digite:

```
>> title ('Título do Gráfico')
```

## 1.9 Funções

A "força" do MATLAB vem de um conjunto extenso de funções. O MATLAB possui um grande número de funções intrínsecas que não podem ser alteradas pelo usuário. Outras funções estão disponíveis em uma biblioteca externa distribuídas com o programa original (MATLAB TOOLBOX), que são na realidade arquivos com a extensão ".m" criados a partir das funções intrínsecas ou de funções criadas pelo usuário. A biblioteca externa (MATLAB TOOLBOX) pode ser constantemente atualizada à medida que novas aplicações são desenvolvidas. As funções do MATLAB, intrínsecas ou arquivos ".m", podem ser utilizadas apenas no ambiente MATLAB.

As categorias gerais de funções matemáticas disponíveis no MATLAB incluem:

- ✓ Matemática elementar;
- ✓ Funções especiais;
- ✓ Matrizes elementares;
- ✓ Matrizes especiais;
- ✓ Decomposição e fatorização de matrizes;
- ✓ Análise de dados;
- ✓ Polinômios;
- ✓ Solução de equações diferenciais;
- ✓ Equações não-lineares e otimização;
- ✓ Integração numérica;
- ✓ Processamento de sinais.

As seções subseqüentes mostram mais detalhes dessas diferentes categorias de funções.

# 2 OPERAÇÕES COM MATRIZES

As operações com matrizes no MATLAB são as seguintes:

- Adição;
- Subtração;
- Multiplicação;
- Divisão a direita;
- Divisão a esquerda;
- Exponenciação;
- Transposta;

A seguir cada uma dessas operações é mostrada com mais detalhe.

## 2.1 Transposta

O caracter apóstrofo, " ' " , indica a transposta de uma matriz. A declaração

```
>> A = [1 2 3; 4 5 6; 7 8 9]
>> B = A`
```

que resulta em

```
A =
     1     2     3
     4     5     6
     7     8     9
B =
     l     4     7
     2     5     8
     3     6     9
```

e

```
>> x = [-1 0 2]`
```

produz

```
x =
    -1
     0
     2
```

Se **Z** é uma matriz complexa, **Z'** será o complexo conjugado composto. Para obter simplesmente a transposta de **Z** deve-se usar **Z.'**, como mostra o exemplo

```
>> Z = [1 2; 3 4] + [5 6; 7 8]*i
>> Z1 = Z'
>> Z2 = Z.'
```

que resulta em

```
Z =
   1.0000 + 5.0000i   2.0000 + 6.0000i
   6.0000 + 7.0000i   4.0000 + 8.0000i
Z1 =
```

```
                1.0000 - 5.0000i   3.0000 - 7.0000i
                2.0000 - 6.0000i   4.0000 - 8.0000i

        Z2 =
                1.0000 + 5.0000i   3.0000 + 7.0000i
                2.0000 + 6.0000i   4.0000 + 8.0000i
```

## 2.2 Adição e Subtração

A adição e subtração de matrizes são indicadas, respectivamente, por "+" e "-". As operações são definidas somente se as matrizes as mesmas dimensões. Por exemplo, a soma com as matrizes mostradas acima, **A + x**, não é correta porque **A** é 3x3 e **x** é 3x1. Porém,

```
>> C = A + B
```

é aceitável, e o resultado da soma é

```
C =
     2     6    10
     6    10    14
    10    14    18
```

A adição e subtração também são definidas se um dos operadores é um escalar, ou seja, uma matriz l x l. Neste caso, o escalar é adicionado ou subtraído de todos os elementos do outro operador. Por exemplo:

```
>> y = x - 1
```

resulta em

```
y =

    -2
    -1
     1
```

## 2.3 Multiplicação

A multiplicação de matrizes é indicada por "*". A multiplicação **x*y** é definida somente se a segunda dimensão de x for igual à primeira dimensão de **y.** A multiplicação

```
>> x'* y
```

é aceitável, e resulta em

```
ans =
        4
```

É evidente que o resultado da multiplicação **y'*x** será o mesmo. Existem dois outros produtos que são transpostos um do outro.

```
>> x*y'

ans =

     2     l     -l
     0     0     0
    -4    -2     2

>> y*x'

ans =

     2     0     -4
     1     0     -2
    -1     0      2
```

O produto de uma matriz por um vetor é um caso especial do produto entre matrizes. Por exemplo **A** e **X,**

```
>> b =     A'x
```

que resulta em

```
b =
        5
        8
        -7
```

Naturalmente, um escalar pode multiplicar ou ser multiplicado por qualquer matriz.

```
>> pi*x

ans =
```

```
-3.1416
0
6.2832
```

## 2.4 Divisão

Existem dois símbolos para divisão de matrizes no MATLAB "\" e "/". Se A é uma matriz quadrada não singular, então **A\B** e **B/A** correspondem respectivamente à multiplicação à esquerda e à direita da matriz **B** pela inversa da matriz **A**, ou **inv(A)*B** e **B*inv(A)N,** mas o resultado é obtido diretamente. Em geral,

- **X** = **A\B** é a solução de **A*X = B**
- **X = B/A** é a solução de **X*A = B**

Por exemplo, como o vetor **b** foi definido como **A*x,** a declaração

```
>> z = A\b
```

resulta em

```
z =
     -1
      0
      2
```

## 2.5 Exponenciação

A expressão **A^p** eleva **A** à **p**-ésima potência e é definida se **A** é matriz quadrada e **p** um escalar. Se **p** é um inteiro maior do que um, a exponenciação é computada como múltiplas multiplicações. Por exemplo,

```
>> A^3

ans =

   279   360   306
   684   873   684
   738   900   441
```

# 3 OPERAÇÕES COM CONJUNTOS

O termo *operações com conjuntos* é usado quando as operações aritméticas são realizadas entre os elementos que ocupam as mesmas posições em cada matriz (elemento por elemento). As operações com conjuntos são feitas como as operações usuais, utilizando-se dos mesmos caracteres (“*****”, ”/”, “\”, “^” e “ ‘ “) precedidos por um ponto “**.**” (“**.***”, ”**./**”, “**.**”, “.^” e “ **.‘** “).

## 3.1 Adição e Subtração

Para a adição e a subtração, a operação com conjuntos e as operações com matrizes são as mesmas. Deste modo os caracteres "+" e "-" podem ser utilizados tanto para operações com matrizes como para operações com conjuntos.

## 3.2 Multiplicação e Divisão

A multiplicação de conjuntos é indicada por “**.***”. Se **A** e **B** são matrizes com as mesmas dimensões, então **A.*B** indica um conjunto cujos elementos são simplesmente o produto dos elementos individuais de **A** e **B**. Por exemplo, se

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

então,

```
>> z = x .* y
```

resulta em

```
z=
        4       10      18
```

As expressões **A./B** e **A.\B** formam um conjunto cujos elementos são simplesmente os quocientes dos elementos individuais de **A** e **B.** Assim,

```
>> z = x .\ y
```

resulta em

```
z =
    4.0000    2.5000    2.0000
```

## 3.3 Exponenciação

A exponenciação de conjuntos é indicada por "**.**^". A seguir são mostrados alguns exemplos usando os vetores **x** e **y.** A expressão

```
>> z = x .^ y
```

resulta em

```
z =
    l     32    729
```

A exponenciação pode usar um escalar.

```
>> z = x. ^2

z =
    l     4     9
```

Ou, a base pode ser um escalar.

```
>> z = 2.^[x y]

z =
    2     4     8    16    32    64
```

## 3.4 Operações Comparativas

Estes são os seis operadores usados para comparação de duas matrizes com as mesmas dimensões:

| | |
|---|---|
| < | menor |
| <= | menor ou igual |
| > | maior |
| >= | maior ou igual |
| == | igual |
| ~= | diferente |

A comparação é feita entre os pares de elementos correspondentes e o resultado é uma matriz composta dos números um e zero, com um representando **VERDADEIRO** e zero, **FALSO**. Por exemplo**,**

```
>> 2 + 2 ~= 4

ans =
        0
```

Pode-se usar, também os operadores lógicos **&** (e) e **|**(ou). Por exemplo,

```
>> 1= = 1 & 4 = = 3

ans =
        0

>> 1 = = 1 | 4 = = 3

ans =
        1
```

## 4 MANIPULAÇÃO DE VETORES E MATRIZES

O MATLAB permite a manipulação de linhas, colunas, elementos individuais e partes de matrizes.

### 4.1 Gerando Vetores

Os dois pontos, " : ", é um caracter importante no MATLAB. A declaração

```
>> x = 1 : 5
```

gera um vetor linha contendo os números de 1 a 5 com incremento unitário. Produzindo

```
x =
     l     2     3     4     5
```

Outros incrementos, diferentes de um, podem ser usados.

```
>> y = 0 : pi/4 : pi
```

que resulta em

```
y =
    0.0000  0.7854  1.5708  2.3562  3.1416
```

Incrementos negativos também são possíveis.

```
>> z = 6 : -l : l

z =
     6     5     4     3     2     1
```

Pode-se, também, gerar vetores usando a função **linspace**. Por exemplo,

```
>> k = linspace (0, l, 6)

k =
    0  0.2000  0.4000  0.6000  0.8000  1.0000
```

gera um vetor linearmente espaçado de 0 a 1, contendo 6 elementos.

## 4.2 Elementos das Matrizes

Um elemento individual da matriz pode ser indicado incluindo os seus subscritos entre parênteses. Por exemplo, dada a matriz A:

```
A =

     l    2    3
     4    5    6
     7    8    9
```

a declaração

```
>> A(3,3) = A(1,3) + A(3,l)
```

resulta em

```
A =
     l    2     3
     4    5     6
     7    8     10
```

Um subscrito pode ser um vetor. Se X e V são vetores, então **X(V)** é **[X(V(1)), X(V(2)), .... X(V(n))]**. Para as matrizes, os subscritos vetores permitem o acesso à submatrizes contínuas e descontínuas. Por exemplo, suponha que A é uma matriz 10x10.

A =

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 92 | 99 | 11 | 18 | 15 | 67 | 74 | 51 | 58 | 40 |
| 98 | 80 | 17 | 14 | 16 | 73 | 55 | 57 | 64 | 41 |
| 14 | 81 | 88 | 20 | 22 | 54 | 56 | 63 | 70 | 47 |
| 85 | 87 | 19 | 21 | 13 | 60 | 62 | 69 | 71 | 28 |
| 86 | 93 | 25 | 12 | 19 | 61 | 68 | 75 | 52 | 34 |
| 17 | 24 | 76 | 83 | 90 | 42 | 49 | 26 | 33 | 65 |
| 23 | 15 | 82 | 89 | 91 | 48 | 30 | 32 | 39 | 66 |
| 79 | 16 | 13 | 95 | 97 | 29 | 31 | 38 | 45 | 72 |
| 10 | 12 | 94 | 96 | 78 | 35 | 37 | 44 | 46 | 53 |
| 11 | 18 | 100 | 77 | 84 | 36 | 43 | 50 | 27 | 59 |

então

```
>> A(1:5,3)

ans =
     11
     17
     88
     19
     25
```

especifica uma submatriz 5x1, ou vetor coluna, que consiste dos cinco primeiros elementos da terceira coluna da matriz **A**. Analogamente,

```
>> A(1:5,7:10)

ans =

     74    51    58    40
     55    57    64    41
     56    63    70    47
     62    69    71    28
     68    75    52    34
```

é uma submatriz 5x4, consiste das primeiras cinco linhas e as últimas quatro colunas.

Utilizando os dois pontos no lugar de um subscrito denota-se todos elementos da linha ou coluna. Por exemplo,

```
>> A(1:2:5,:)

ans =
    92    99    11    18    15    67    74    51    58    40
    14    81    88    20    22    54    56    63    70    47
    86    93    25    12    19    61    68    75    52    34
```

é uma submatriz 3x10 que consiste da primeira, terceira e quinta linhas e todas colunas da matriz **A**.

Muitos efeitos sofisticados são obtidos usando submatrizes em ambos os lados das declarações. Por exemplo, sendo **B** uma matriz 10x10 unitária,

```
>> B = ones (10)

B =
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
```

a declaração,

```
>> B(1:2:7,6:l0) = A(5:-1:2,1:5)
```

produz

```
     1     1     1     1     1    86    93    25    12    19
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1    85    87    19    21    13
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1    14    81    88    20    22
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1    98    80    17    14    16
```

```
1   1   1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1   1   1
```