UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA MECÂNICA

TUTORIAL MATLAB

CONCEITOS BÁSICOS

Autor: Prof. Paulo S. Varoto

São Carlos

2020

CAPÍTULO 1

1.1 Introdução

1.1.1 Iniciando o MATLAB

Um duplo clique no ícone MATLAB irá inicializar o programa. Assim, aparecerá na tela uma janela de comandos e o "prompt" padrão (EDU>> ou >>) é exibido na tela. Espera-se do usuário um comando, o qual deve ser finalizado teclando-se Enter.

1.1.2 Janelas Exibidas

A janela principal do MATLAB chama-se Command Window (Janela de Comando), onde os dados e instruções são digitados no "prompt". Quando se deseja implementar algum programa, projeto ou trabalho, utiliza-se o M-File Editor. Neste editor, cria-se um arquivo texto ".m" com os comandos desejados. Além dessas duas janela principais, existe a Graphics Window utilizada para exibir os gráficos.

1.1.3 Entrando com comandos

Cada comando tem de ser seguido por um <cr> (tecla enter) para que o comando possa ser executado. Para executar um arquivo ".m" basta digitar o nome do arquivo, sem sua extensão, na janela de comandos.

1.1.4 Expo MATLAB

Para ver a demonstração de algumas capacidades do MATLAB, entre com o comando "demo" na janela de comandos. Isto iniciará o MATLAB EXPO, o qual é um ambiente de demonstração gráfica que mostra alguns dos diferentes tipos de operações que podem ser realizados com MATLAB.

1.1.5 Abortar

Para abortar um comando em MATLAB, mantenha pressionada a tecla Ctrl e pressione a tecla c.

1.1.6 Comando clc

O comando clc limpa a janela de comandos e coloca o cursor na posição inicial.

1.1.7 Ajuda do MATLAB

Através do comando "help", o usuário pode consultar a ajuda do MATLAB. Para isso, deve-se digitar help e o nome da função desejada na janela de comandos, onde aparecerá um pequeno resumo da função seguido de uma descrição mais detalhada da mesma.

1.1.8 Declarações e Variáveis

Para criar e/ou armazenar informações em variáveis definidas pelo usuário, basta digitar o nome da variável, seguido do sinal de igual "=", e da expressão desejada.

1.1.9 Saindo do MATLAB

Os comandos quit e exit são usados para encerrar o MATLAB.

1.2 Operações aritméticas

A tabela 1.1 apresenta os símbolos utilizados para as operações aritméticas básicas.

Tabela 1.1 Operações básicas

Operação Aritmética	**Símbolo**	**Exemplo**
Adição	+	10+5 = 15
Subtração	-	10-5 = 5
Multiplicação	*	10*5 = 50
Divisão a direita	/	10/5 = 2
Divisão a esquerda	\	10\5 = 1/2
Exponenciação	^	10^5 = 100000

1.3 Formato de visualização dos números

O MATLAB possui diversas formas de apresentar um número. Estes formatos podem ser encontrados no "help", digitando na janela de comandos: help format. Alguns desses formatos são apresentados na tabela 1.2.

Tabela 1.2 Formato de exibição

Comando	**Formato**	**Exemplo** $\sqrt{2}$
Format short	4 dígitos decimais.	1.4142
Format long	14 dígitos decimais.	1.41421356237310
Format short e	Notação exponencial com 4 dígitos decimais.	1.4142e+000
Format long e	Notação exponencial com 15 dígitos decimais.	1.414213562373095e+000
Format short g	O melhor entre "short" e "short e".	1.4142
Format long g	O melhor entre "long" e "long e".	1.4142135623731
Format bank	2 dígitos decimais representando moeda.	1.41
Format compact	Elimina linhas em branco para permitir que mais linhas com informações possam ser exibidas.	
Format loose	Adiciona linhas (Oposto de "format compact").	

1.4 Funções matemáticas

O MATLAB possui uma gama de funções matemáticas que estão disponíveis ao usuário, como por exemplo, funções trigonométricas, logarítmicas, elementares, etc. Algumas dessas funções são apresentadas nas tabelas abaixo.

Tabela 1.3 Funções Trigonométricas

Função	**Descrição**
sin(x)	Calcula o seno de x, onde o x está em radianos.
cos(x)	Calcula o cosseno de x, onde o x está em radianos.
tan(x)	Calcula a tangente de x, onde o x está em radianos.
cot(x)	Calcula a cotangente de x, onde o x está em radianos.
sec(x)	Calcula a secante de x, onde o x está em radianos.
csc(x)	Calcula a cossecante de x, onde o x está em radianos.
asin(x)	Calcula o arco cujo seno é x, onde x deve estar entre -1 e 1.

acos(*x*)	Calcula o arco cujo cosseno é *x*, onde *x* deve estar entre -1 e 1.
atan(*x*)	Calcula o arco cuja tangente é *x*.
atang2(*y*,*x*)	Calcula o arco cuja tangente é *y/x*.
sinh(*x*)	Calcula o seno hiperbólico de *x*.
cosh(*x*)	Calcula o cosseno hiperbólico de *x*.
tanh(*x*)	Calcula a tangente hiperbólica de *x*.
asinh(*x*)	Calcula o arco cujo seno hiperbólico é *x*.
acosh(*x*)	Calcula o arco cujo cosseno hiperbólico é *x*.
atanh(*x*)	Calcula o arco cuja tangente hiperbólica é *x*.

Tabela 1.4 Funções Elementares

Função	**Descrição**
abs(*x*)	Calcula o valor absoluto de *x*.
sqrt(*x*)	Calcula a raiz quadrada de *x*.
round(*x*)	Arredonda o valor de x para o inteiro mais próximo.
fix(*x*)	Arredonda o valor de x para o inteiro mais próximo do zero.
floor(*x*)	Arredonda o valor de x para o inteiro em direção -∞.
ceil(*x*)	Arredonda o valor de x para o inteiro em direção +∞.
sign(*x*)	Retorna $\begin{cases} -1, se\ x < 0 \\ 0, se\ x = 0 \\ 1, se\ x > 0 \end{cases}$
rem(*x*,*y*)	Retorna o resto da divisão de *x* por *y*.
gcd(*x*,*y*)	Calcula o máximo divisor comum de *x* e *y*.
lcm(*x*,*y*)	Calcula o mínimo múltiplo comum de *x* e *y*.
exp(*x*)	Calcula o valor de e^x, onde $e \cong 2{,}718282$.
log(*x*)	Calcula o logaritmo de *x* na base e.
log10(*x*)	Calcula o logaritmo de *x* na base 10.

Tabela 1.5 Funções de números complexos

Função	**Descrição**

conj(x)	Calcula o complexo conjugado do número complexo x. Se $x = a + bi$ então conj(x) será $a - bi$.
real(x)	Retorna a parte de real do número complexo x.
imag(x)	Retorna a parte imaginária do número complexo x.
abs(x)	Calcula o valor absoluto em magnitude do número complexo x.
angle(x)	Calcula o ângulo usando o valor **atan2(imag(x),real(x)).**

As operações com os números complexos são apresentadas na tabela 1.6. Seja $c_1 = a_1 + b_1 i$ e $c_2 = a_2 + b_2 i$.

Tabela 1.6 Operações com números complexos

Operação	**Resultado**
$c_1 + c_2$	$(a_1 + a_2) + i(b_1 + b_2)$
$c_1 - c_2$	$(a_1 - a_2) + i(b_1 - b_2)$
$c_1 . c_2$	$(a_1 a_2 - b_1 b_2) + i(a_1 b_2 + a_2 b_1)$
$\frac{c_1}{c_2}$	$\frac{a_1 a_2 + b_1 b_2}{a_2^2 + b_2^2} + \frac{a_2 b_1 - a_1 b_2}{a_2^2 + b_2^2}$
$\lvert c_1 \rvert$ = abs(c_1)	$\sqrt{a_1^2 + b_1^2}$

1.5 Variáveis no MATLAB

Uma variável é formada por uma letra ou uma combinação de caracteres. Os nomes das variáveis devem ser formados por uma única palavra, conforme descrito na tabela 1.7

Tabela 1.7 Regras de construção de variável

Regra de construção das variáveis	**Exemplo**
Variáveis com letras minúsculas e maiúsculas são diferentes.	XX, Xx, xX e xx

Variáveis podem ser construídas com até 63 caracteres (MATLAB 7).	
As variáveis devem começar com uma letra que pode ser seguida de letras, números ou subscrito _.	Var_2 x34 a_b

O MATLAB possui alguns nomes que são utilizados para variáveis predefinidas. Estes são apresentados na tabela 1.8.

Tabela 1.8 Variáveis predefinidas

Variáveis predefinidas	**Descrição**
ans	Variável usada para exibir os resultados.
pi	Número 3,14159.
eps	Menor número tal que, quando adicionado a 1, cria um número maior que 1 no computador.
inf	Representa o infinito, por exemplo, ocorre nas divisões por zero.
i ou j	Unidade imaginária $\sqrt{-1}$.
NaN ou nan	Significa que não é um número, por exemplo, na divisão 0/0.
clock	Representa o tempo corrente num vetor de seis elementos contendo ano, mês, dia, hora, minuto e segundo.
date	Representa a data atual no seguinte formato: 06-Sep-2011.

1.6 Comandos de gerenciamento

O MATLAB possui comandos que podem ser utilizados para eliminar variáveis ou para obter informações sobre variáveis que foram criadas. Para isso a tecla Enter deve ser pressionada após digitar o comando na janela de comandos. Alguns desses comandos são descrito na tabela 1.9.

Tabela 1.9 Comandos de gerenciamento

Comando	**Descrição**
clear	Remove todas as variáveis da memória.
clear x, y, z	Remove somente as variáveis x, y e z da memória.
who	Lista as variáveis correntes armazenadas na área de trabalho.
whos	Mostra uma lista de variáveis correntes armazenadas com informações detalhadas de seus tamanhos.

1.7 Comentários e pontuação

Tabela 1.10 Comentário e Pontuação

Símbolo	**Função**
,	Separar comandos em uma mesma linha.
;	SE um ponto e vírgula é digitado ao final de um comando, a impressão na tela é suprimida, mas a tarefa é realizada.
%	Todo e qualquer caractere depois do símbolo % é tomado como comentário.
. . .	Para continuar uma expressão matemática na próxima linha utiliza-se um espaço em branco e três pontos, ao final das linhas incompletas.

1.8 Comandos gerais

As tabelas 1.11 até 1.14 mostram comandos gerais de ajuda, da área de trabalho e de diretórios.

Tabela 1.11 Help Online

Comando	**Descrição**
help	Lista todos os tópicos que a ajuda está disponível.
helpwin	Abre a janela de ajuda interativa.
helpdesk	Abre o navegador web de ajuda.

help *topic*	Fornece ajuda sobre o tópico em questão.
lookfor *string*	Lista os tópicos de ajuda contendo a expressão.

Tabela 1.12 Informações da área de trabalho

Comando	**Descrição**
what	Lista os arquivos ".m", ".mat" e "mex" do diretório.
clear all	Limpa todas as variáveis e funções da área de trabalho.
mlock *fun*	Trava a função *fun* e assim o comando clear não pode removê-la.
munlock *fun*	Destrava a função *fun* e assim o comando clear pode removê-la.
home	Tem a mesma função de clc.
clf	Limpa a janela de figuras.

Tabela 1.13 Informações de Diretório

Comando	**Descrição**
pwd	Mostra o atual diretório de trabalho.
cd	Muda de diretório de trabalho.
dir	Lista o conteúdo do diretório atual.
ls	O mesmo que dir.
path	Obtém ou define um caminho de pesquisa do MATLAB.
editpath	Modifica o caminho de pesquisa do MATLAB.
copyfile	Copia um arquivo.
mkdir	Cria um diretório.

Tabela 1.14 Informações Gerais

Comando	**Descrição**
computer	Mostra o tipo de computador que você está usando.
more	Controla a saída paginada de acordo com o tamanho da tela.
ver	Mostra a licença e informações sobre a versão do MATLAB instalado no computador.

bench	Compara as operações em MATLAB realizadas pelo seu computador em relação a outros .

CAPÍTULO 2

2.1 Gerando um vetor

A tabela 2.1 apresenta algumas maneiras de construir um vetor.

Tabela 2.1 Vetores

Vetor X	Descrição
x = primeiro : último	Cria um vetor x começando com o valor primeiro, incrementando-se de 1 em 1 até atingir o valor último, ou valor mais próximo possível de último.
x = primeiro : increm : último	Cria um vetor x começando com o valor primeiro, incrementando-se do valor de increm. até atingir o valor último, ou valor mais próximo possível de último..
x = linspace(primeiro, último, n)	Cria um vetor x começando com o valor primeiro e terminando no valor último, contendo n elementos linearmente espaçados.
x = logspace(primeiro, último, n)	Cria um vetor x começando com o valor $10^{primeiro}$ e terminando no valor $10^{último}$, contendo n elementos logaritmicamente espaçados.
x = [1 2 3]	Cria um vetor linha x com os elementos especificados.

A seguir apresentam-se alguns exemplos:

a) `>> x=1:5`

```
x = 1     2     3     4     5
```

b) `>> x=1:6.3`

```
x = 1     2     3     4     5     6
```

c) `>> x=1:0.1:1.5`

```
x = 1.0000    1.1000   1.2000   1.3000    1.4000    1.5000
```

d) `>> z=linspace(1,2,6)`

```
z = 1.0000   1.2000   1.4000    1.6000    1.8000    2.0000
```

e) `>> z=logspace(0,2,5)`

```
z =1.0000   3.1623   10.0000   31.6228  100.0000
```

f) `>> x=[1 2 3]`

```
x = 1     2     3
```

No último exemplo foi criado um vetor linha, pode-se criar um vetor coluna separando os elementos por ponto e vírgula (;). Veja no exemplo a seguir:

```
>> x=[1;2;3]
x =  1
     2
     3
```

Esses vetores coluna podem também ser criados a partir dos comandos utilizados anteriormente para criar os vetores linha, acompanhados do símbolo ('), que é o operador de transposição. Exemplo:

```
>> y=(1:0.5:2)'
y = 1.0000
    1.5000
    2.0000
```

2.2 Gerando uma Matriz

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. Veja o exemplo:

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

As linhas das matrizes também podem ser definidas através dos comandos utilizados anteriormente para se definir vetores linha. Por exemplo:

```
>> A=[1:3;linspace(4,6,3);7:1:9]
A =   1      2      3
      4      5      6
      7      8      9
```

2.3 Endereçamento de vetores e matrizes

No MATLAB, cada um dos elementos de um vetor pode ser acessado através de seu índice que identifica cada uma das colunas. Por exemplo:

x(:) – refere-se a todos os elementos do vetor x.
x(m:n) – refere-se aos elementos m até n do vetor x.

Por exemplo:

```
x=[2 5 -8 7 6 -4 1]
 y=x(2:5)
y = 5   -8      7      6
```

A tabela 2.2 apresenta como usar o endereçamento em matrizes.

Tabela 2.2 Endereçamento de matrizes

Comando	Descrição
A(:,n)	Refere-se aos elementos de todas as linhas da coluna *n* da matriz A.
A(n, :)	Refere-se aos elementos de todas as colunas da linha *n* da matriz A.

A(:, m:n)	Refere-se aos elementos de todas as linhas entre as colunas *m* e *n* da matriz A.
A(m:n, :)	Refere-se aos elementos de todas as colunas entre as linhas *m* e *n* da matriz A.
A(m:n, p:q)	Refere-se aos elementos entre as linhas *m* e *n* e colunas *p* e *q* da matriz A.

2.4 Operações com Vetores

Sejam os vetores a = [a_1 a_2 ... a_n], b = [b_1 b_2 ... b_n] e c um escalar. A tabela 2.3 apresenta as operações básicas entre vetores. Tais operações só são possíveis quando estes tiverem o mesmo tamanho e orientação (linha ou coluna).

Tabela 2.3 Operações com vetores

Operação	**Expressão**	**Resultado**
Adição escalar	$a+c$	$[a_1 + c \quad a_2 + c \ldots a_n + c]$
Adição vetorial	$a+b$	$[a_1 + b_1 \quad a_2 + b_2 \ldots a_n + b\ ]$
Multiplicação escalar	$a*c$	$[a_1 * c \quad a_2 * c \ldots a_n * c]$
Multiplicação vetorial	$a.*b$	$[a_1 * b_1 \quad a_2 * b_2 \ldots a_n * b\ ]$
Divisão	$a./b$	$[a_1 / b_1 \quad a_2 / b_2 \ldots a_n / b\ ]$
Potenciação	$a.\hat{}c$	$[a_1 \hat{}c \quad a_2 \hat{}c \ldots a_n \hat{}c]$
	$c.\hat{}a$	$c[\hat{}a_1 \quad c\hat{}a_2 \ldots c\hat{}a_n]$
	$a.\hat{}b$	$[a_1 \hat{}b_1 \quad a_2 \hat{}b_2 \ldots a_n \hat{}b_n]$

2.5 Operações com Matrizes

2.5.1 Transposta

O caractere apóstrofo, " ' " , indica a transposta de uma matriz. Veja o exemplo abaixo:

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

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

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

2.5.2 Adição e Subtração

A adição (+) e subtração (-) de matrizes são realizadas elemento a elemento, por isso só são realizadas se as matrizes tiverem a mesma dimensão.

Exemplos:

```
>> C=A+B
C =   2     6    10
      6    10    14
     10    14    18
```

```
>> D=A-B
D =   0    -2    -4
      2     0     -2
      4     2      0
```

2.5.3 Multiplicação

A multiplicação de matrizes é indicada por "*". A multiplicação A*B é definida somente se o número de linhas de A for igual ao número de colunas de B.

```
>> E=A*B
E =   14    32    50
      32    77   122
      50   122   194
```

Pode-se também multiplicar matrizes por escalares, como no exemplo abaixo:

```
>> F=0.5*E
```

```
F =   7.0000   16.0000   25.0000
     16.0000   38.5000   61.0000
     25.0000   61.0000   97.0000
```

Além da multiplicação matricial e escalar, podemos ter a multiplicação elemento por elemento de matrizes de mesma dimensão. Esse tipo de operação é feita utilizando-se um ponto (.) antes do operador de multiplicação (*). Ou seja, se A e B são matrizes definidas por A=[a_{11} a_{12} ... a_{1n} ; a_{21} a_{22} ... a_{2n} ; ... ; a_{m1} a_{m2} ... a_{mn}] e B=[b_{11} b_{12} ... b_{1n} ; b_{21} b_{22} ... b_{2n} ; ... ; b_{m1} b_{m2} ... b_{mn}], então A.*B =a_{ij}*b_{ij}.

```
>> A.*B
ans =   1      8      21
        8     25     48
        21     48     81
```

2.5.4 Divisão

Existem dois símbolos para divisão de matrizes no MATLAB “ / ” que representa a divisão matricial a direita e “ \ ” que representa a divisão matricial a esquerda. 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 seja, inv(A)*B e B*inv(A).

Assim como na multiplicação, também existe a divisão elemento por elemento de matrizes, definida de forma similar. Por exemplo, A./B = a_{ij}/b_{ij} e A.\B = $a_{ij}\backslash b_{ij}$.

2.5.5 Potenciação

A expressão A^n representa a matriz A elevada a n-ésima potência, onde A é uma matriz quadrada e *n* um escalar. Se *n* é um inteiro maior do que um, a exponenciação é computada como múltiplas multiplicações. Por exemplo:

```
>> A^3
ans =  468          576          684
       1062         1305         1548
       1656         2034         2412
```

No caso da exponenciação elemento a elemento, tem-se: A.^B = $a_{ij}^{\;b_{ij}}$, onde A e B possuem a mesma dimensão. Da mesma forma, a exponencial de uma matriz por em escalar *c* é dada por: A.^c = $a_{ij}^{\;c}$.

2.6 Álgebra Linear

2.6.1 Determinante

Seja A uma matriz. O comando **det(A)** retorna o determinante da matriz A. Por exemplo:

```
>> A=[1 2;3 4]
A =    1     2
       3     4
>> det(A)
ans = -2
```

2.6.2 Matriz Inversa

A matriz B é a inversa de A, quando as duas matrizes são multiplicadas e o produto é a matriz identidade. O comando **inv(A)** retorna a matriz inversa da matriz A.

```
>> B=inv(A)
B =   -2.0000    1.0000
       1.5000   -0.5000
```

2.6.3 Auto vetores e Auto valores

O MATLAB determina os autovalores e autovetores da matriz A. O comando **eig(A)** calcula um vetor coluna contendo os autovalores de A. O comando **[Q,d] = eig(A)** calcula uma matriz quadrada Q contendo os autovetores de A como colunas e uma matriz quadrada d contendo os autovalores de A na diagonal.

Exemplo:

```
>> eig(A)
ans =   -0.3723
         5.3723

>> [Q,d]=eig(A)
```

```
Q =  -0.8246   -0.4160
      0.5658   -0.9094


d = -0.3723         0
      0          5.3723
```

2.6.4 Fatoração triangular superior-inferior (LU)

A fatoração triangular expressa uma matriz quadrada como produto de duas matrizes triangulares, sendo uma matriz triangular inferior e outra matriz triangular superior. O comando **[L,U] = lu(A)** expressa uma matriz triangular inferior L e uma matriz triangular superior U, tal que o produto de L por U é igual a matriz A.

```
>> [L,U]=lu(A)
L =  0.3333    1.0000
     1.0000         0


U =  3.0000    4.0000
      0        0.6667
```

2.6.5 Fatoração QR

O método de fatoração QR da matriz A expressa um par de matrizes Q e R tais que Q é ortogonal e R é triangular superior. O produto de Q por R é igual a matriz A. O comando **[Q,R] = qr(A)** encontra tais matrizes Q e R.

```
>> [Q,R]=qr(A)
Q =  -0.3162   -0.9487
     -0.9487    0.3162


R =  -3.1623   -4.4272
       0       -0.6325
```

Se a matriz A é *m x n*, então Q será *m x m* e R será *m x n*.

2.6.6 Decomposição em valores singulares (SVD)

A decomposição em valores singulares decompõe a matriz A em um produto de três matrizes tal que A= USV, onde U e V são matrizes ortogonais e S é uma matriz diagonal. Os valores da diagonal da matriz S são chamados valores singulares. Para este tipo de fatoração utiliza-se o comando **[U,S,V] = svd(A).**

```
>> [U,S,V]=svd(A)
U =  -0.4046   -0.9145
     -0.9145    0.4046

S =  5.4650         0
      0          0.3660


V = -0.5760    0.8174
    -0.8174   -0.5760
```

O comando **svd(A)** retorna os elementos da diagonal de S, que são os valores singulares de A.

2.7 Funções de gerenciamento de matrizes e vetores

Algumas funções para gerenciamento de manipulações de matrizes são descritas na tabela 2.4.

Tabela 2.4 Funções de gerenciamento

Função	**Descrição**	**Exemplo**
length(A)	Retorna o número de elementos de um vetor A.	>> A = [1 3 6 0] >>length(A) ans = 4

size(A)	Retorna um vetor linha [m,n], onde *m* e *n* é o tamanho *m* x *n* de A.	>> A = [1 2 3 4; 5 6 7 8] A = 1 2 3 4 5 6 7 8 >>size(A) ans = 2 4
reshape(A,m,n)	Rearranja a matriz A que têm r linhas e s colunas para ter m linhas e n colunas.	>> A=[3 1 4; 9 0 7] A = 3 1 4 9 0 7 >>B = reshape(A,3,2) B = 3 0 9 4 1 7
diag(*v*)	Quando *v* é um vetor, cria uma matriz quadrada com os elementos de *v* na diagonal principal.	>> v = [3 2 1]; >>A = diag(v) A = 3 0 0 0 2 0 0 0 1
diag(A)	Quando A é uma matriz, cria um vetor com os elementos da diagonal de A.	>> A = [1 8 3; 4 2 6;7 8 3] A = 1 8 3 4 2 6 7 8 3 >> vec = diag(A) vec = 1 2 3

2.8 Análise de Dados de Vetores e Matrizes

A tabela 2.5 apresenta algumas funções de análise de matrizes e vetores.

Tabela 2.5 Análise de dados

Função	**Descrição**	**Exemplo**

mean(v)	Se *v* é um vetor, retorna o valor médio dos elementos de *v*.	>> v=[1 5 6 8 48 56 2 0]; >> mean(s) ans = 15.7500
C = max(A)	Se A é um vetor, C é o valor do maior elemento. Se a A é uma matriz, C é um vetor linha contendo o maior elemento de cada coluna de A.	>> A=[4 6 9 4 15 2 0]; >> C=max(A) C = 15
[d,n] = max(A)	Se A é um vetor, d é o maior elemento de A, n é a posição do elemento (a primeira, se tiver diversos valores máximos).	>> [d,n]=max(A) d = 15 n = 5
min(A) [d,n] = min(A)	O mesmo de max(A), só que min(A) retorna o menor elemento.	>> A=[2 7 8 0 4]; >> min(A) ans = 2
sum(A)	Se A é um vetor, retorna a soma dos elementos do vetor.	>> A=[2 3 9 5 4]; >> sum(A) ans = 23
sort(A)	Se A é um vetor, coloca os elementos em ordem crescente.	>> A=[2 5 1 10 3]; >> sort(A) ans = 1 2 3 5 10
median(A)	Se A é um vetor, coloca os elementos em ordem crescente e retorna um valor mediano desses elementos.	>> A=[-1 0 2 8 9 13]; >> median(A) ans = 5
std(A)	Se A é um vetor, retorna o desvio padrão dos elementos de A.	>> A=[-1 0 2 3]; >> std(A) ans =1.8257
dot(A)	Calcula o produto escalar de dois vetores *a* e *b*.	>> a= [5 6 7]; b= [4 3 2]; >> dot(a,b) ans = 52

cross(a,b)	Calcula o produto cruzado de dois vetores *a* e *b*. Os dois vetores devem ter 3 elementos.	>> a= [5 6 7]; >> b= [4 3 2]; >> cross(a,b) ans = -9 18 -9

2.9 Gerando Números Randômicos

Cria uma matriz de elementos pseudo-aleatórios com distribuição uniforme entre 0 e 1, dados os números de linhas e colunas. Alguns comandos são apresentados na tabela 2.6.

Tabela 2.6 Números randômicos

Comando	**Descrição**	**Exemplo**
rand (1,n)	Gera um vetor linha de *n* elementos, com números randômicos entre 0 e 1.	>> a = rand(1,3) a = 0.8147 0.9058 0.1270
rand(n)	Gera uma matriz *n* x *n*, com números randômicos entre 0 e 1.	>> b= rand(3) b = 0.9134 0.2785 0.9649 0.6324 0.5469 0.1576 0.0975 0.9575 0.9706
rand(m,n)	Gera uma matriz *m* x *n*, com números randômicos entre 0 e 1.	>> c=rand(2,3) c = 0.9572 0.8003 0.4218 0.4854 0.1419 0.9157
randperm(n)	Gera um vetor linha com *n* elementos que são permutações randômicas dos inteiros de 1 até *n*.	>> randperm(5) ans = 4 3 1 5 2

2.10 Funções Polinomiais

Um polinômio $f(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_2 x^2 + a_1 x + a_0$ é interpretado no MATLAB como um vetor, começando com o termo de maior grau, $[a_n \ \ a_{n-1} \ \ ... \ a_2 \ \ a_1 \ \ a_0]$.

Por exemplo:

$f(x) = 3x^4 + 5x^3 + 7x + 3$

```
>> f=[3 5 0 7 3]
f = 3     5     0     7     3
```

Assim, listam-se na tabela 2.7 algumas funções que trabalham com polinômios.

Tabela 2.7 Funções polinomiais

Função	**Descrição**	**Exemplo** $f(x) = x^2 + 5x + 6$**,** $g(x) = 2x + 4$
roots	Retorna um vetor linha com as raízes do polinômio de entrada.	>> f=[1 5 6]; >> roots(f) ans = -3.0000 -2.0000
polyval	Retorna o valor ou a imagem de um polinômio, dados respectivamente o polinômio e o valor de sua variável independente.	>> polyval(f,2) ans = 20 Ou seja, $1(2)^2 + 5(2) + 6 = 20$
poly	Cria um polinômio a partir de um vetor de entrada contendo suas raízes.	>> poly([-3 -2]) ans = 1 5 6
conv	Multiplica, de forma distributiva, dois polinômios. (mas não dois vetores)	>> conv(f,g) ans =2 14 32 24
deconv	Divide dois polinômios.	>> deconv(f,g) ans = 0.5000 1.5000

2.11 Expressões Simbólica

No MATLAB, é possível manipularmos expressões que além de números e variáveis numéricas, contêm também variáveis simbólicas. Para definir as variáveis simbólicas, utiliza-se o comando *syms.*

Por exemplo:

```
>> syms x
```

Uma vez definido que a variável x é uma variável simbólica, podemos definir expressões que envolvem esta variável.

Por exemplo:

```
>> syms x
>> f= 3*x^2+5*x-3;
>> g=2*x+1;

>> f+g
ans = 3*x^2+7*x-2

>> f-g
ans = 3*x^2+3*x-4

>> f*g
ans =(3*x^2+5*x-3)*(2*x+1)
```

Aqui pode-se usar o comando *expand* para obter o resultado da expressão.

```
>> expand(ans)
ans =6*x^3+13*x^2-x-3

>> f/g
ans =(3*x^2+5*x-3)/(2*x+1)

>> expand(ans)
ans =  3/(2*x+1)*x^2+5/(2*x+1)*x-3/(2*x+1)
```

O MATLAB pode realizar operações mais avançadas sobre expressões simbólicas. A função *compose* calcula a composição das funções f(x) e g(x) em f(g(x)), a função *finverse* encontra a inversa funcional de uma expressão e a função *subs* substitui uma variável por um número (ou por outra variável) em uma expressão. A tabela 2.8 resume algumas das manipulações com expressões algébricas.

Tabela 2.8 Manipulações Simbólicas

Função	**Descrição**	**Exemplo**
diff(f)	Calcula a derivada de f	>> f=3*x^2+5*x-3; >> diff(f) ans =6*x+5
int(f)	Calcula a integral indefinida de f.	>> int(f) ans = x^3+5/2*x^2-3*x
compose (f,g)	Determina a composta f(g(x)).	>> compose(f,g) ans =3*(2*x+1)^2+10*x+2
expand(expres)	Expande uma expressão.	>> h=(x-1)^2; >> expand(h) ans =x^2-2*x+1
finverse(g)	Determina a função inversa de g.	>> finverse(g) ans = -1/2+1/2*x
pretty(expres)	Escreve uma expressão expres de forma mais bonita.	>> pretty(g) 2 x + 1
simple(expres)	Procura encontrar uma forma mais simples de escrever uma expressão.	
simplify (expres)	Simplifica a expressão.	

solve (expres)	Acha a(s) solução(es) da equação expres = 0.	>> syms a b c x >> solve(a*x^2+b*x+c) ans= -1/2*(b-(b^2-4*a*c)^(1/2))/a -1/2*(b+(b^2-4*a*c)^(1/2))/a
subs(expres, x,a)	Substitui na expressão a variável x por a.	>> subs(f,x,2) ans = 19

Outras funções utilizando manipulações simbólicas podem ser encontradas digitando *help symbolic* na janela de comandos.

CAPÍTULO 3

3.1 Gráficos

O MATLAB é um software muito eficiente na criação e manipulação de gráficos, apresentando diversas funções que auxiliam essas operações. Podem ser confeccionados gráficos bidimensionais, tridimensionais, malhas e superfícies.

3.1.1 Gráficos bidimensionais

O comando básico para plotar um gráfico bidimensional simples é:

plot(valores de x, valores de y, opção de estilo)

Os valores de x e de y são vetores contendo as coordenadas x-y de pontos do gráfico. As opções de estilo são as especificações de cor, estilo de linha e marcador de pontos. A tabela 3.1 apresenta estes estilos.

Tabela 3.1 Cor, estilo de linha e opções de marcadores

Opção de cor	**Estilo de linha**	**Opção de marcador**

y amarelo	- linha sólida	+ cruz
m magenta	- - linha tracejada	o círculo
c azul claro	: linha pontilhada	* asterístico
r vermelho	-. linha tracejada e pontilhada	x xis
g verde		. ponto
b azul		^ triângulo para cima
w branco		s quadrado
k preto		d losango

Existem várias funções especializadas em plotar gráficos bidimensionais no MATLAB. A tabela 3.2 apresenta algumas delas:

Tabela 3.2 Comandos de gráficos

Função	**Descrição**
area	Cria um gráfico com área preenchida
bar	Cria um gráfico de barras
barh	Cria um gráfico de barras horizontais
comet	Faz um plano de animação bidimensional
compass	Cria um gráfico de setas para números complexos
contour	Cria um gráfico de contorno
contourf	Cria um gráfico de contorno preenchido
errorbar	Cria um gráfico e coloca barras de erros
feather	Cria um gráfico de setas
fill	Desenha polígonos da cor especificada
fplot	Plota funções de uma única variável
hist	Faz histogramas
loglog	Cria gráfico com escala logarítmica nas coordenadas x-y
pareto	Cria gráfico de barras em ordem decrescente
pcolor	Faz um gráfico pseudo colorido de matriz

pie	Cria um gráfico de pizza
plotyy	Faz gráfico com dois eixos y
plotmatrix	Faz gráfico de dispersão de uma matriz
polar	Plota curvas em coordenadas polares
quiver	Plota campo de vetores
rose	Faz histogramas angulares
scatter	Cria gráficos de dispersão
semilogx	Cria um gráfico com escala logarítmica no eixo x
semilogy	Cria um gráfico com escala logarítmica no eixo y
stairs	Cria um gráfico em forma de escada
stem	Cria um gráfico de linhas verticais

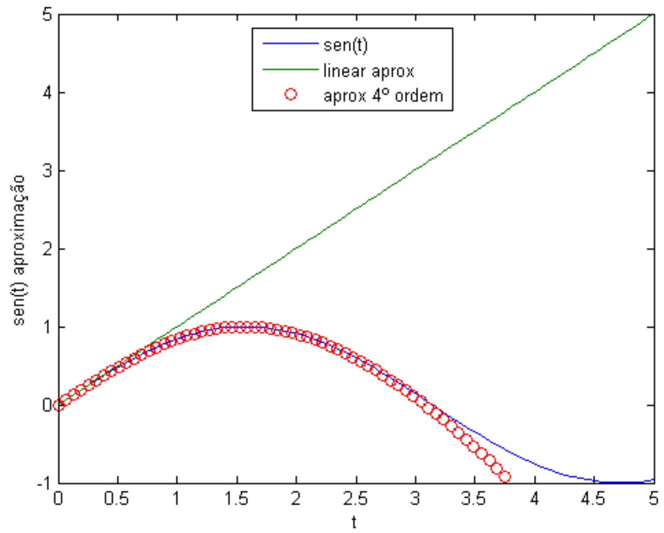
No MATLAB existem três maneiras de criar gráficos em sobreposições: o comando *plot*, o comando *hold* e o comando *line*.

Por exemplo: Plotar y_1 = sen(t); y_2 = t e $y_3 = t - \frac{t^3}{3!} + \frac{t^5}{5!} + \frac{t^7}{7!}; \quad 0 \le t \le 2\pi$

a) Comando plot

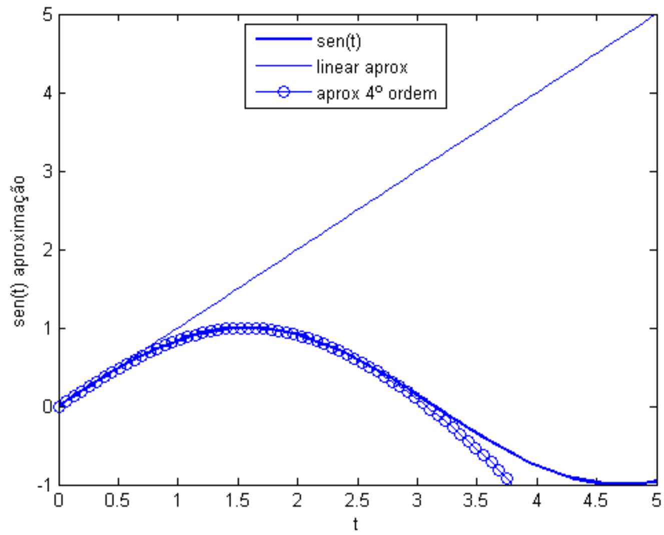
```
>> plot(t, y1, t, y2, '-', t, y3, 'o')
>> xlabel('t')
>> ylabel('sen(t) aproximação')
```

```
>>legend('sen(t)',linear aprox', 'aprox 4° ordem')
```



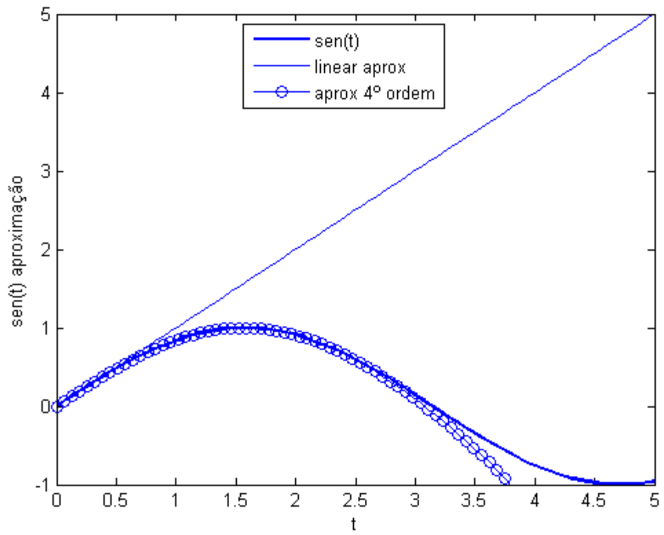
b) comando hold

```
>> plot(t,y1, 'linewidth',2)
>> hold on
>> plot(t,y2,'-')
>> plot(t,y3,'o')
>> axis([0 5 -1 5])
>> xlabel('t')
>> ylabel('sen(t) aproximação')
>>legend('sen(t)',linear aprox', 'aprox 4° ordem')
>> hold off
```



c) comando line

```
>> plot(t,y1,'linewidth',2)
>> line(t, y2, 'linestyle', '-')
>> line(t, y3, 'marker', 'o')
>> axis([0 5 -1 5])
>> xlabel('t')
>> ylabel('sin(t) approximation')
>> legend('sin(t)', 'linear approx', '7th order approx')
```



3.1.2 Gráficos tridimensionais

A tabela 3.3 apresenta alguns comandos para plotar gráficos tridimensionais e de contorno.

Tabela 3.3 Comandos de gráficos 3D

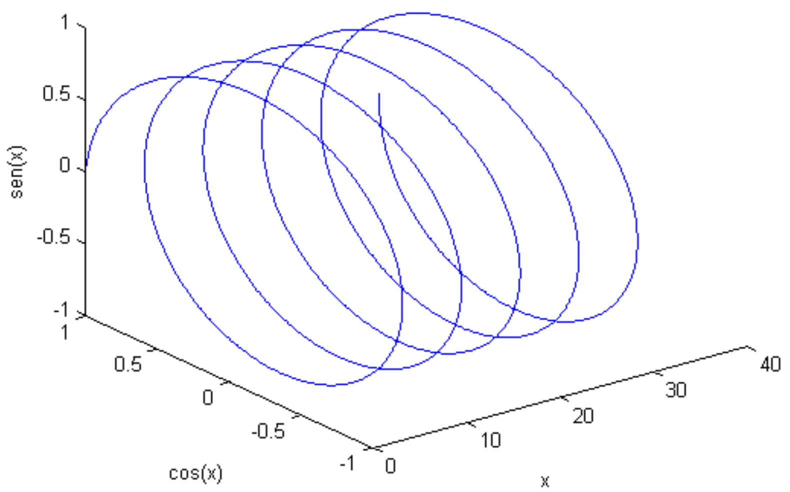
Comando	Descrição
plot3	Plota um gráfico no espaço 3D
fill3	Desenha um polígono 3D
comet3	Plota um 3D uma trajetória de cometa
contour	Plota um gráfico de contorno 2D
contour3	Plota um gráfico de contorno 3D

clabel	Plota gráfico de contorno com valores
quiver	Plota gradiente
mesh	Plota malha 3D
meshc	Combinação de mesh e contour
surf	Plota superfície 3D
surfc	Combinação de surf e contour
surfil	Plota superfície 3D com iluminação
slice	Plota visualização volumétrica
cylinder	Gera um cilindro
sphere	Gera uma esfera

Exemplos de gráficos tridimensionais

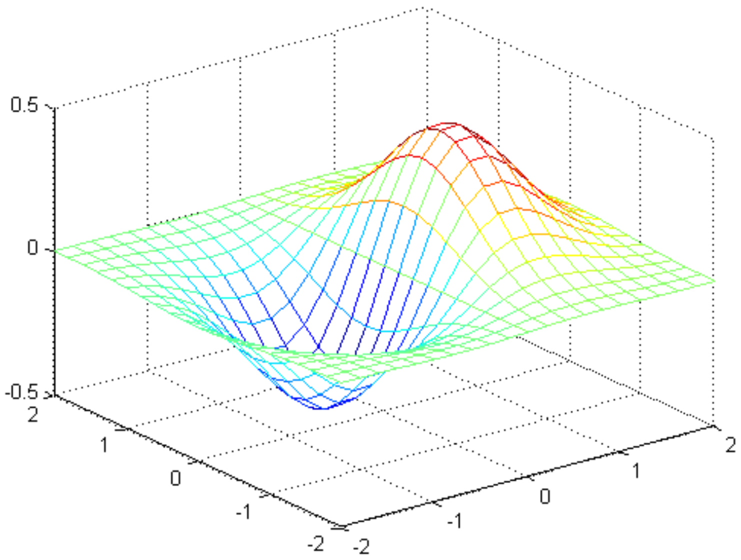
a) Usando o comando plot3

```
>> x=0:0.01:10*pi;
>> plot3(x,cos(x),sin(x))
>> xlabel('x')
>> ylabel('cos(x)')
>> zlabel('sen(x)')
```



b) Usando o comando mesh

```
>> [X,Y]=meshgrid(-2:0.2:2,-2:0.2:2);
>> Z = X .* exp(-X.^2 - Y.^2);
>> mesh(X,Y,Z)
```



3.1.3 Anotações em gráficos

A tabela 3.4 apresenta comandos de fácil utilização para adicionar informações em um gráfico.

Tabela 3.4 Comandos de anotações

Comando	**Descriçao**	**Exemplo**
title	Adiciona um título ao gráfico	Title('título')
xlabel	Título no eixo x	Xlabel('nome em x')
ylabel	Título no eixo y	Xlabel('nome em y')
zlabel	Título no eixo z	Xlabel('nome em z')
text	Inserir anotação no gráfico	
gtext	Inserir anotação com o mouse	
grid	Inserir linhas de grade	

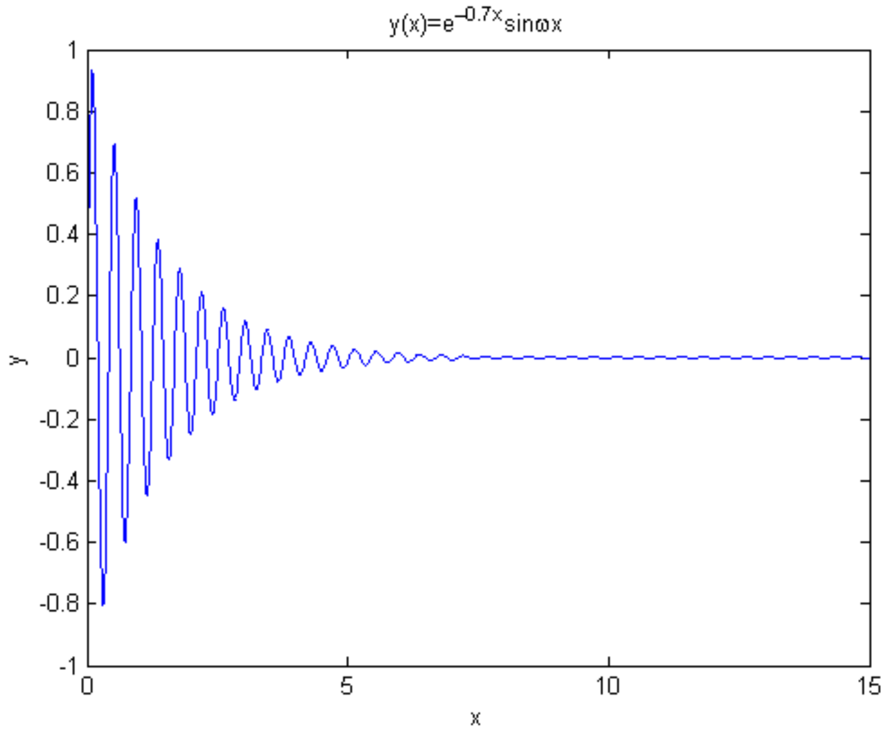
CAPÍTULO 4

4.1 Exemplos

1) Gere o gráfico da função $y(x) = e^{-0.7x} sen(\omega x)$para $\omega = 15 rad / s$ e $0 \le x \le 15$.

Solução:

```
>> x=[0:0.01:15];
>> w=15;
>> y=exp(-0.7*x).*sin(w*x);
>> plot(x,y)
>> title('y(x)=e^-^0^.^7^xsin\omegax')
>> xlabel('x')
>> ylabel('y')
```



2) Uma expressão analítica para a resposta amortecida de um sistema de um grau de liberdade dadas as condições iniciais de deslocamento e velocidade é dada por:

$x(t) = Ce^{-\zeta \omega_n t} \cos(\omega_d t - \phi)$

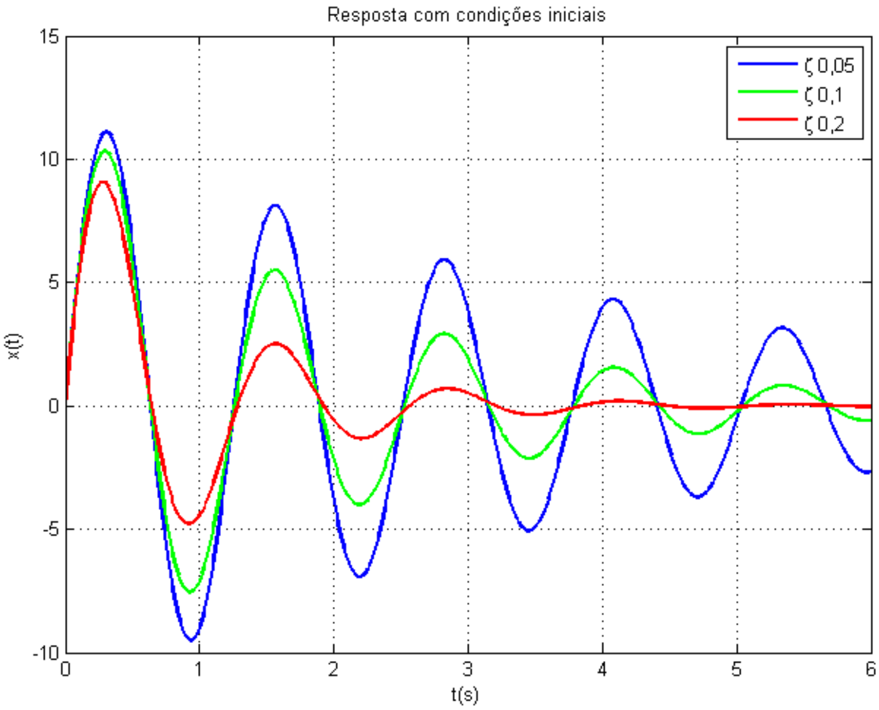
onde C e ϕ representam a amplitude e o ângulo de fase da resposta do sistema, respectivamente e são dados por:

$$C = \sqrt{x_0^2 + \left(\frac{\zeta\omega_n x_0 + v_0}{\omega_d}\right)^2}\ , \ \phi = \tan^{-1}\left(\frac{\zeta\omega_n x_0 + v_0}{\omega_d x_0}\right) \text{ e } \omega_d = \sqrt{1-\zeta^2}\omega_n$$

Plote a resposta do sistema usando o MATLAB para $\omega = 5rad/s$, $\zeta = 0{,}05$, $\zeta = 0{,}1$ e $\zeta = 0{,}2$ sabendo que as condições iniciais são $x(0) = 0$ e $\dot{x}(0) = v_0 = 60cm/s$.

Solução:

```
wn=5; % Frequência natural
zeta=[0.05;0.1;0.2]; % razão de amortecimento
x0=0; % delocamneto inicial
v0=60; % velocidade inicial
t0=0; % tempo inicial
deltat=0.01; % intervalo no tempo
tf=6; % tempo final
t=[t0:deltat:tf];

for i=1:length(zeta)
  wd=sqrt(1-zeta(i)^2)*wn;
   x=exp(-
zeta(i)*wn*t).*(((zeta(i)*wn*x0+v0)/wd)*sin(wd*t)+x0*cos(wd
*t));

    plot(t,x,'r')
    hold on
end

title('Resposta com condições iniciais')
xlabel('t(s)')
ylabel('x(t)')
legend('\zeta 0,05','\zeta 0,1','\zeta 0,2')
grid
```



3) Resolva o sistema de equações de Lorenz

$$\begin{cases} \dfrac{dx}{dt} = -\sigma x + \sigma y \\ \dfrac{dy}{dt} = \rho x - y - xz \\ \dfrac{dz}{dt} = -\beta z + xy \end{cases}$$

onde $\sigma = 10$, $\beta = 8/3$ e $\rho = 28$. As condições iniciais são $x(0) = -8$, $y(0) = 8$ e $z(0) = 27$.

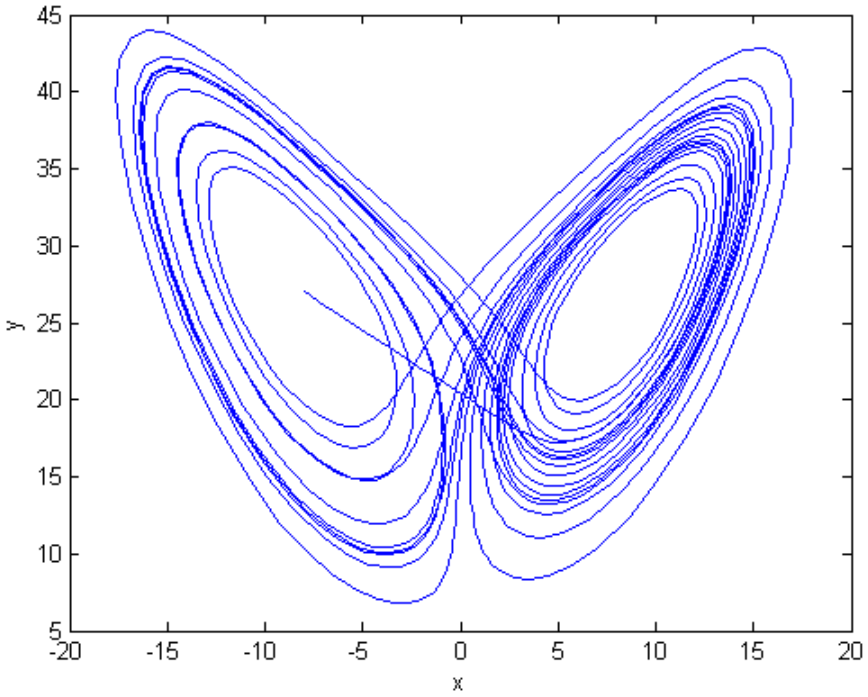
Solução:

Neste caso criamos um arquivo .m

```
function exemplo= lorenz(t,x);

sigma=10;
beta=8/3;
ro=28;

exemplo=[-sigma*x(1)+sigma*x(2);ro*x(1)-x(2)-x(1)*x(3);-
beta*x(3)+x(1)*x(2)];
```

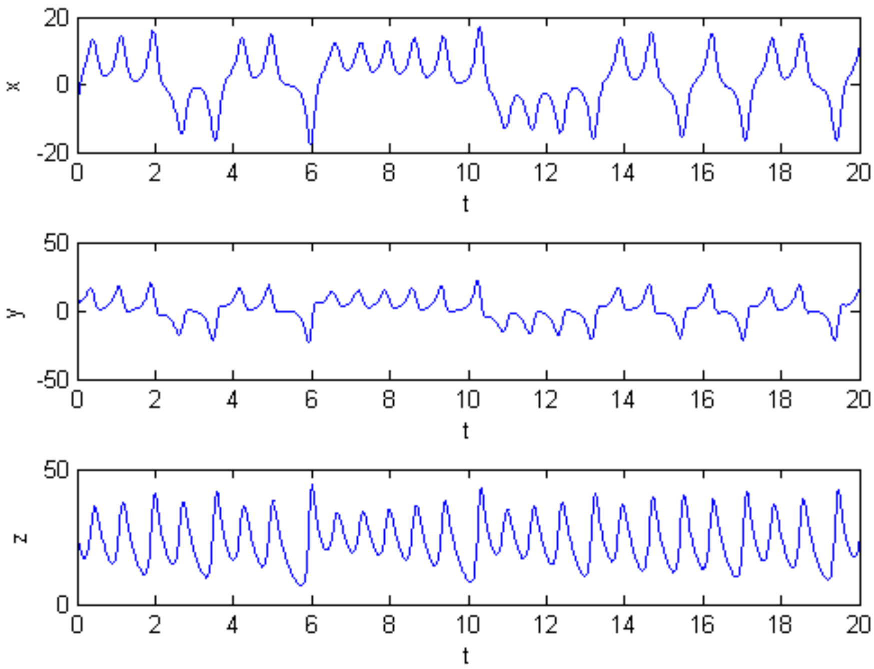
A seguir podemos digitar os seguintes comandos na janela de comando:

```
>> tspan=[0.0 20.0];
>> x0=[-8 8 27];
>> [t,x]=ode45(@Lorenz,tspan,x0);
>> plot(x(:,1),x(:,3))
>> xlabel('x')
>> ylabel('y')
```



Podemos também plotar a resposta no tempo:

```
>> subplot(3,1,1)
>> plot(t,x(:,1))
>> xlabel('t')
>> ylabel('x')
>> subplot(3,1,2)
>> plot(t,x(:,2))
>> xlabel('t')
>> ylabel('y')
>> subplot(3,1,3)
>> plot(t,x(:,3))
>> xlabel('t')
>> ylabel('z')
```

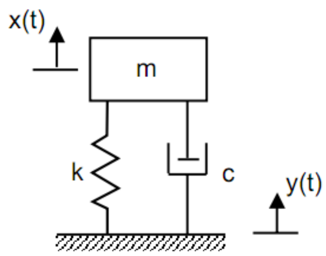


4) Obtenha 5 termos do desenvolvimento em série de Taylor da função $f(x) = \cos(x)$.

Solução:

```
>> syms x
>> taylor(cos(x),5)

ans =
 1-1/2*x^2+1/24*x^4
```

5) Plote no MATLAB a magnitude da resposta adimensional e o ângulo de fase para o sistema com movimento harmônico da base mostrado na figura abaixo:



Solução:

A magnitude da resposta em frequência é dada por:

$$|G(i\omega)| = \frac{1}{\left[\left[\left(1-\frac{\omega}{\omega_n}\right)^2\right]^2 + \left(2\zeta\frac{\omega}{\omega_n}\right)^2\right]^{1/2}}$$

A magnitude de $X(i\omega)$é dado por: $|X(i\omega)| = \left[1+\left(\frac{2\zeta\omega}{\omega_n}\right)^2\right]^{1/2} |G(i\omega)|A$

onde $y(t) = \operatorname{Re} A^{i\omega t}$

$x(t) = X(i\omega)e^{i\omega t}$

O ângulo de fase é dado por: $\phi(\omega) = \tan^{-1}\left[\frac{2\zeta\left(\frac{\omega}{\omega_n}\right)^3}{1-\left(\frac{\omega}{\omega_n}\right)^2+\left(\frac{2\zeta\omega}{\omega_n}\right)^2}\right]$

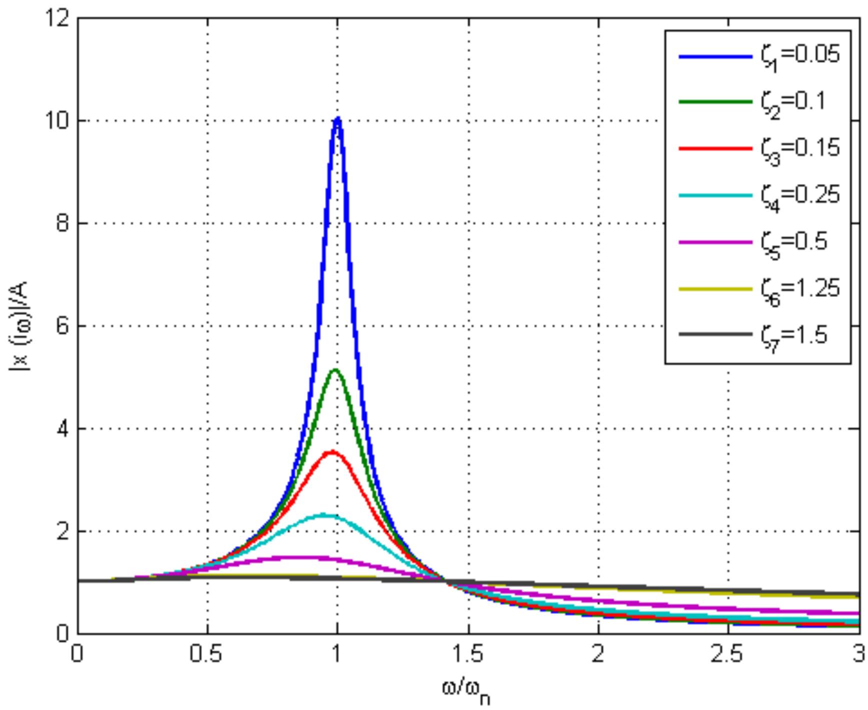
A razão de frequência é: $r = \frac{\omega}{\omega_n}$

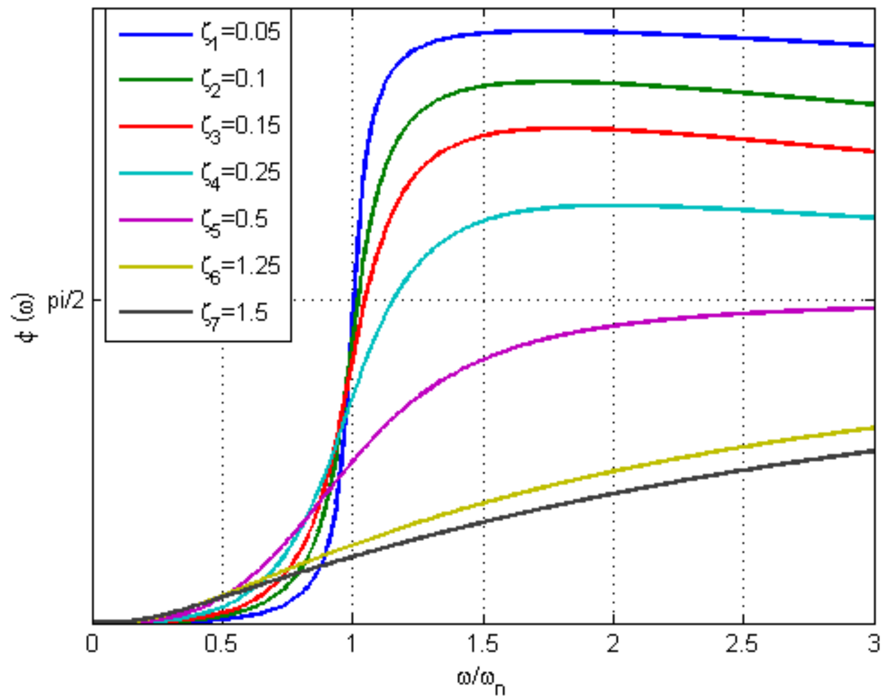
A magnitude da resposta adimensional é:

$$\frac{|X(i\omega)|}{A} = \left[\frac{1+\left(\frac{2\zeta\omega}{\omega_n}\right)^2}{1-\left(\frac{\omega}{\omega_n}\right)^2+\left(\frac{2\zeta\omega}{\omega_n}\right)^2}\right]$$

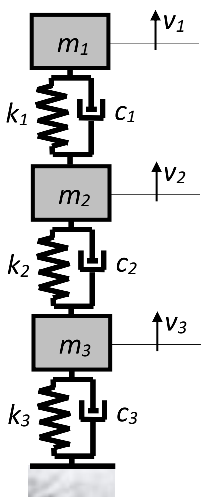
```
zeta= [0.05; 0.1; 0.15; 0.25; 0.5; 1.25; 1.5]; % damping
factors
r= [0:0.01:3]; %frequency ratio
for k=1: length (zeta)

     G(k,:)=sqrt((1+(2*zeta(k)*r).^2)./((1-
r.^2).^2+(2*zeta(k)*r).^2));

     phi(k,:)=atan2(2*zeta(k)*r.^3,1-
r.^2+(2*zeta(k)*r).^2);

end
figure (1)
plot(r, G,'linewidth',1.5)
xlabel ('\omega/\omega_n')
ylabel ('|x (i\omega)|/A')
grid
```

```
legend('\zeta_1=0.05','\zeta_2=0.1','\zeta_3=0.15','\zeta_4
=0.25','\zeta_5=0.5','\zeta_6=1.25','\zeta_7=1.5')

figure (2)
plot(r, phi,'linewidth',1.5)
xlabel ('\omega/\omega_n')
ylabel ('\phi (\omega)')
grid
ha=gca;
set (ha,'ytick',[0:pi/2:pi])
set(ha,'yticklabel',{[];'pi/2';'p'})
legend('\zeta_1=0.05','\zeta_2=0.1','\zeta_3=0.15','\zeta_4
=0.25','\zeta_5=0.5','\zeta_6=1.25','\zeta_7=1.5')
```





6) Uma viga em balanço tem seu modelo dinâmico aproximado com 3 graus de liberdade correspondentes aos deslocamentos verticais *w*, conforme mostrado abaixo. No modelo de 3 graus de liberdade mostrado, os deslocamentos verticais w_1, w_2 e w_3 são os deslocamentos absolutos das massas m_1, m_2 e m_3, respectivamente. Este sistema possui os seguintes parâmetros físicos: massas $m_1=m_2=m_3= 2\ kg$ e rigidez $k_1 = 600\ N/m$, $k_2 = 1200\ N/m$ e $k_3 = 2400\ N/m$. Com a ajuda do MATLAB obtenha: as frequências naturais, os modos de vibrar correspondentes, a matriz modal, a matriz de massa modal e a matriz de rigidez modal.



Solução:

```
%Parâmetros
%Massa
m1=2;
m2=2;
m3=2;
%Rigidez
k1=600;
k2=1200;
k3=2400;

%Matriz massa
m=[m3 0 0;0 m2 0;0 0 m1];

%Matriz Rigidez
k=[k2+k3 -k2 0;-k2 k1+k2 -k1;0 -k1 k1];

A=inv(m)*k

%Autovalores de A => frequências naturais
%Autovetores de A => Modos de vibrar

[Fi,lamb]=eig(A)

Fi =

    0.1706   -0.4317   -0.8857
    0.4732   -0.7526    0.4579
    0.8643    0.4973   -0.0759

lamb =

  1.0e+003 *

    0.1357         0         0
         0    0.7540         0
         0         0    2.1102


wn=sqrt(lamb) %Frequência natural em rad/s
wn =

       11.6511         0            0
         0        27.4598           0
         0              0        45.9370

fn=wn/(2*pi) %Frequência natural em Hz
fn =

    1.8543         0            0
         0      4.3704          0
         0         0        7.3111
```

```
w1=wn(1,1)
w2=wn(2,2)
w3=wn(3,3)

w1 = 11.6511 %Primeira frequência natural


w2 = 27.4598%Segunda frequência natural


w3 = 45.9370%Terceira frequência natural

%Massa modal
Mr=Fi'*m*Fi
Mr =

    2.0000         0   -0.0000
         0    2.0000    0.0000
   -0.0000    0.0000    2.0000

%Rigidez modal
Kr=Fi'*k*Fi
Kr =

  1.0e+003 *

    0.2715   -0.0000    0.0000
         0    1.5081    0.0000
         0    0.0000    4.2204


i=1:3;
Fi1=Fi(i,1) %Primeiro modo
Fi2=Fi(i,2) %Segundo modo
Fi3=Fi(i,3) %Terceiro modo

Fi1 =

    0.1706
    0.4732
    0.8643


Fi2 =

   -0.4317
   -0.7526
    0.4973
```

```
Fi3 =

   -0.8857
    0.4579
   -0.0759
```