

Apêndice

MATLAB

O *MATLAB* se tornou uma poderosa ferramenta de profissionais técnicos espalhados pelo mundo todo. O termo *MATLAB* é uma abreviação de *MATrix LABoratory*, o que implica que o *MATLAB* é uma ferramenta computacional que usa matrizes e vetores (ou *arrays*) para realizar tarefas de análise numérica, processamento de sinais e visualização científica. Como o *MATLAB* utiliza matrizes como componente fundamental, poderiam ser escritas expressões matemáticas envolvendo matrizes de forma tão fácil quanto seria se usássemos papel. O *MATLAB* está disponível para os sistemas operacionais Macintosh, Unix e Windows. Existe uma versão educacional do *MATLAB* para computadores pessoais (PCs).

Pode-se obter uma cópia do *MATLAB* na

The Mathworks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098
Telefone: (508) 647-7000
Site: <<http://www.mathworks.com>>

Neste apêndice, apresentamos uma breve introdução ao *MATLAB*. O que é exposto aqui é suficiente para resolver os problemas deste livro. Mais informações sobre o *MATLAB* podem ser encontradas em livros *MATLAB* e na ajuda *online*. A melhor maneira de aprender o *MATLAB* é usá-lo após ter conhecido seus fundamentos.

1 Fundamentos do *MATLAB*

A janela Command é a principal área onde interagimos com o *MATLAB*. Um pouco mais à frente, veremos como usar o editor de texto para criar arquivos M, que possibilitam a execução de sequências de comandos. Por enquanto, iremos nos concentrar em como trabalhar na janela Command. Veremos, primeiro, como usar o *MATLAB* como uma calculadora.

Uso do *MATLAB* como uma calculadora

São utilizados os seguintes operadores algébricos no *MATLAB*:

- + Adição
- Subtração

- * Multiplicação
- ^ Exponenciação
- / Divisão pela direita (a/b significa $a \div b$)
- \ Divisão pela esquerda ($a\b$ significa $b \div a$)

Para começar a usar o *MATLAB*, utilizamos esses operadores. Digite comandos no prompt “>>” do *MATLAB* na janela Command (corrija quaisquer erros usando a tecla Backspace) e pressione a tecla Enter. Por exemplo,

```
>> a = 2; b = 4; c = -6;
>> dat = b^2 - 4*a*c
dat =
64
>> e = sqrt(dat)/10
e =
0.8000
```

O primeiro comando atribui os valores 2, 4 e -6, respectivamente, às variáveis *a*, *b* e *c*. O *MATLAB* não responde porque essa linha termina com um sinal de dois pontos (:). O segundo comando configura *dat* para $b^2 - 4ac$ e o *MATLAB* retorna uma resposta igual a 64. Finalmente, a terceira linha configura *e* igual à raiz quadrada de *dat* e divide esta por 10. O *MATLAB* imprime a resposta 0,8. Outras funções matemáticas, apresentadas na Tabela 1, podem ser utilizadas de forma similar à maneira pela qual a função *sqrt* é usada aqui. A Tabela 1 apresenta apenas um pequeno exemplo das funções do *MATLAB*. Outras podem ser obtidas da ajuda *online*. Para obter ajuda, digite

```
>> help
```

Surgirá na tela uma longa lista de tópicos. Para saber sobre um tópico específico, digite o nome do comando. Por exemplo, para obter ajuda sobre o comando para “log na base 2”, digite

```
>> help log2
```

Será exibida uma mensagem de ajuda sobre a função *log*. Note que o *MATLAB* diferencia o uso de caracteres maiúsculos e minúsculos, de modo que *sen* (*a*) não é o mesmo que *sen* (*A*).

Tabela 1 • Funções matemáticas elementares típicas.

Função	Comentário
<i>abs</i> (<i>x</i>)	Valor absoluto ou amplitude complexa de <i>x</i>
<i>acos</i> , <i>acosh</i> (<i>x</i>)	Arco cosseno e cosseno hiperbólico inverso de <i>x</i> em radianos
<i>acot</i> , <i>acoth</i> (<i>x</i>)	Arco cotangente e cotangente hiperbólica inversa de <i>x</i> em radianos
<i>angle</i> (<i>x</i>)	Ângulo de fase (em radianos) de um número complexo <i>x</i>
<i>asin</i> , <i>asinh</i> (<i>x</i>)	arco seno e seno hiperbólico inverso de <i>x</i> em radianos
<i>atan</i> , <i>atanh</i> (<i>x</i>)	Arco tangente e tangente hiperbólica inversa de <i>x</i> em radianos

(Continua)

Tabela 1 • Funções matemáticas elementares típicas. (Continuação)

Função	Comentário
<code>conj(x)</code>	Conjugado complexo de x
<code>cos, cosh(x)</code>	Cosseno e cosseno hiperbólico de x em radianos
<code>cot, coth(x)</code>	Cotangente e cotangente hiperbólica de x em radianos
<code>exp(x)</code>	Exponencial de x
<code>fix</code>	Arredondamento para zero
<code>imag(x)</code>	Parte imaginária de um número complexo x
<code>log(x)</code>	Logaritmo natural de x
<code>log2(x)</code>	Logaritmo de x na base 2
<code>log10(x)</code>	Logaritmos comuns (base 10) de x
<code>real(x)</code>	Parte real de um número complexo x
<code>sin, sinh(x)</code>	Seno e seno hiperbólico de x em radianos
<code>sqrt(x)</code>	Raiz quadrada de x
<code>tan, tanh</code>	Tangente e tangente hiperbólica de x em radianos

Experimente os exemplos a seguir

```
>> 3^(log10(25.6))
>> y = 2 * sin(pi/3)
>> exp(y+4-1)
```

Além de operar sobre funções matemáticas, o *MATLAB* permite que se trabalhe facilmente com vetores e matrizes. Um vetor (ou *array*) é uma matriz especial de uma linha ou uma coluna. Por exemplo,

```
>> a = [1 -3 6 10 -8 11 14];
```

é um vetor-linha. Definir uma matriz é similar a definir um vetor. Por exemplo, uma matriz 3×3 pode ser introduzida como

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

ou como

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

Além das operações matemáticas que podem ser realizadas em uma matriz, podem ser implementadas as operações da Tabela 2.

Usando as operações na Tabela 2, podemos manipular matrizes como segue:

Tabela 2 • Operações matriciais.

Operação	Comentário
<code>A'</code>	Determina a transposta da matriz A
<code>det(A)</code>	Calcula o determinante da matriz A
<code>inv(A)</code>	Calcula o inverso da matriz A
<code>eig(A)</code>	Determina as raízes características da matriz A
<code>diag(A)</code>	Determina os elementos diagonais da matriz A

```

>> B = A'
B =
    1   4   7
    2   5   8
    3   6   9
>> C = A + B
C =
    2   6  10
    6  10  14
   10  14  18
>> D = A^3 - B*C
D =
    372   432   492
    948  1131  1314
   1524  1830  2136
>> e = [1 2; 3 4]
e =
    1   2
    3   4
>> f = det(e)
f =
   -2
>> g = inv(e)
g =
   -2.0000   1.0000
   1.5000  -0.5000

```

Tabela 3 • Constantes, variáveis e matrizes especiais.

Matriz, variável, constante	Comentário
eye	Matriz identidade
ones	Um <i>array</i> de 1 s
zeros	Um <i>array</i> de 0 s
i ou j	Unidade imaginária ou sqrt (-1)
pi	3,142
NaN	Não é um número
inf	Infinito
eps	Um número muito pequeno, 2.2e - 16
rand	Elemento aleatório

```
>> H = eig(g)
H =
-2.6861
0.1861
```

Observe que nem todas as matrizes podem ser invertidas. Uma matriz pode ser invertida se, e somente se, seu determinante não for zero. Variáveis, constantes e matrizes especiais são enumeradas na Tabela 3. Por exemplo, digite

```
>> eye(3)
ans =
1 0 0
0 1 0
0 0 1
```

para obter uma matriz identidade 3×3 .

Gráfico

Para obter gráficos usando o *MATLAB* é fácil. Para um gráfico bidimensional, use o comando *plot* com dois argumentos, como segue:

```
>> plot(xdata,ydata)
```

onde *xdata* e *ydata* são vetores de mesmo comprimento contendo os dados a serem representados em um gráfico.

Suponha, por exemplo, que queremos representar graficamente $y = 10\sin(2\pi x)$ de 0 a 5π . Prosseguiremos com os seguintes comandos:

```
% x is a vector, 0 <= x <= 5*pi, increments of pi/100
% creates a vector y
% creates the plot
```

Com isso, o *MATLAB* responde com o gráfico da Figura 1.

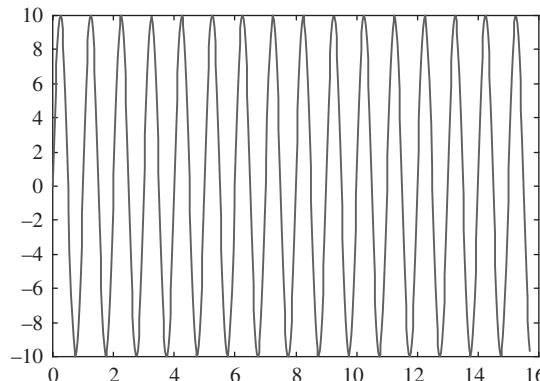


Figura 1 Gráfico do *MATLAB* de $y = 10\sin(2\pi x)$.

O *MATLAB* permitirá que se representem vários gráficos juntos e se faça a distinção entre eles por meio de cores. Isso é obtido com o formato *plot(xdata, ydata, 'color')*, onde a cor é indicada usando-se uma string de caracteres a partir das opções enumeradas na Tabela 4.

```
>> x = 0:pi/100:5*pi;
>> y = 10*sin(2*pi*x);
>> plot(x,y);
```

Tabela 4 • Vários tipos de linhas e cores.

y	Amarelo	.	Ponto
m	Magenta	o	Círculo
c	Ciano	x	Marca x
r	Vermelho	+	Mais
g	Verde	-	Cheia
b	Azul	*	Estrela
w	Branco	:	Pontilhado
k	Preto	-.	Traço-ponto
		--	Tracejado

Por exemplo,

```
>> plot (x1,y1, 'r', x2,y2, 'b', x3,y3, '--');
```

com dados de gráfico (x_1, y_1) em vermelho, dados (x_2, y_2) em azul e dados (x_3, y_3) em linha tracejada, todos colocados no mesmo gráfico.

O *MATLAB* também permite o uso da escala logarítmica. Em vez de usar o comando *plot*, utilizamos

```
loglog log(y) versus log(x)
semilogx y versus log(x)
semilogy log(y) versus x
```

Os gráficos tridimensionais são desenhados, usando-se as funções *mesh* (malha) e *meshgrid* (domínio de malhas). Por exemplo, para desenhar o gráfico de $z = x \cdot \exp(-x^2 - y^2)$ no domínio $-1 < x, y < 1$, digitamos os comandos a seguir:

```
>> xx = -1:.1:1;
>> yy = xx;
>> [x,y] = meshgrid(xx,yy);
>> z = x.*exp(-x.^2 -y.^2);
>> mesh(z);
```

(O símbolo ponto usado em *x.* e *y.* permite a multiplicação elemento por elemento.) O resultado é mostrado na Figura 2.

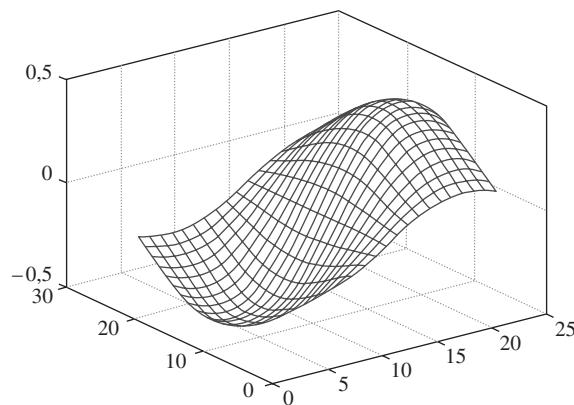


Figura 2 Gráfico tridimensional.

Programação do *MATLAB*

Até agora, usamos o *MATLAB* como uma calculadora. Também podemos utilizá-lo para criar um programa próprio. A edição de linha de comando no *MATLAB* pode ser inconveniente se uma delas tiver várias linhas para serem executadas. Para evitar esse problema, podemos criar um programa que é uma sequência de instruções a serem executadas. Se estiver na janela Command, clique em **File/New/M-files** para abrir um novo arquivo no Editor/Debugger do *MATLAB* ou em um editor de texto comum. Digite o programa e salve-o num arquivo de extensão *.m*, digamos, *filename.m*; é por essa razão que ele é chamado arquivo M. Uma vez salvo o programa como um arquivo M, saia da janela Debugger. Estamos de volta na janela Command. Digite o arquivo sem a

extensão .m para obter os resultados. Por exemplo, o gráfico que foi construído na Figura 2 pode ser aperfeiçoado acrescentando-se título e rótulos, e sendo digitado como um arquivo M chamado example1.m.

```
% x is a vector, 0 <= x <= 5*pi, increments of pi/100
% creates a vector y
% create the plot
% label the x axis
% label the y axis
% title the plot
% add grid
```

Assim que o arquivo for salvo como example1.m e você sair do editor de texto, digite

```
>> example1
```

na janela Command e pressione **Enter** para obter o resultado mostrado na Figura 3.

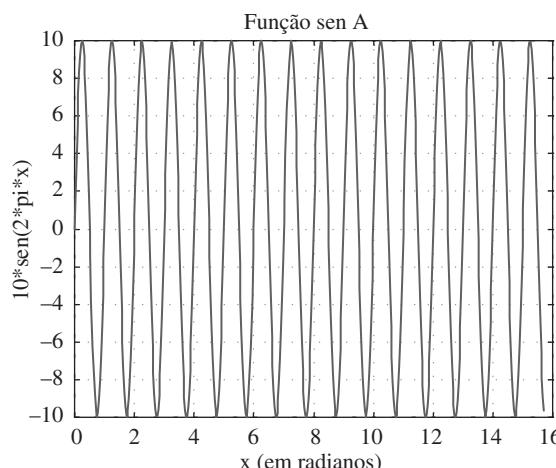


Figura 3 Gráfico MATLAB de $y = 10 \sin(2\pi x)$ com título e rótulos.

Para possibilitar o controle na sequência de execução de um programa, são necessários certos operadores relacionais e lógicos. Eles são mostrados na Tabela 5. Talvez as instruções de controle de fluxo de modo geral usadas sejam **for** e **if**. A instrução **for** é usada para criar um *loop* ou um procedimento repetitivo e possui a forma geral

```
for x = array
    [commands]
end
```

A instrução **if** é usada quando certas condições precisam ser atendidas antes de uma expressão ser executada. Ela possui a forma geral

```
if expression
    [commands if expression is True]
else
    [commands if expression is False]
end
```

```
x = 0:pi/100:5*pi;
y = 10*sin(2*pi*x);
plot(x,y);
xlabel('x (in radians)');
ylabel('10*sin(2*pi*x)');
title('A sine functions');
grid
```

Tabela 5 • Operadores relacionais e lógicos.

Operador	Comentário
<	menor
<=	menor ou igual a
>	maior que
>=	maior ou igual a
==	igual
~=	diferente
&	e (<i>and</i>)
	ou (<i>or</i>)
~	não (<i>not</i>)

Suponha que tenhamos um *array* $y(x)$ e queiramos determinar o valor mínimo de y e seu índice x . Isso pode ser feito criando-se um arquivo M, conforme mostrado aqui.

```
% example2.m
% This program finds the minimum y value and
% its corresponding x index
x = [1 2 3 4 5 6 7 8 9 10]; %the nth term in y
y = [3 9 15 8 1 0 -2 4 12 5];
min1 = y(1); for k = 1:10
    min2 = y(k);
    if(min2 < min1)
        min1 = min2;
        xo = x(k);
    else
        min1 = min1;
    end
end
diary
min1, xo
diary off
```

Observe o uso das instruções *for* e *if*. Quando esse programa é salvo como *example2.m*, nós o executamos na janela Command e obtemos o valor mínimo de y como sendo igual a -2 e o valor correspondente de x , igual a 7 , como esperado.

```
>> example2
min1 =
-2
xo =
7
```

Se não estivermos interessados no índice correspondente, podemos fazer o mesmo usando o comando

```
>> min(y)
```

As dicas, a seguir, são úteis para se trabalhar de forma eficaz com o *MATLAB*:

- Coloque comentários em seu arquivo M adicionando linhas que se iniciam com um caractere %.
- Para suprimir saída, termine cada comando com um ponto e vírgula (;); pode-se eliminar o ponto e vírgula ao depurar o arquivo.
- Pressione as teclas de seta para cima e seta para baixo para recuperar comandos executados anteriormente.
- Se sua expressão não couber em uma linha, use três pontos (...) no final da linha e continue na linha seguinte. Por exemplo, o *MATLAB* considera

```
y = sin(x + log10(2x + 3)) + cos(x + ...
log10(2x + 3));
```

como uma linha de expressão.

- Tenha em mente que os nomes de funções e de variáveis fazem uma distinção entre caracteres maiúsculos e minúsculos.

Resoluções de equações

Consideremos o sistema geral de n equações simultâneas:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

ou na forma matricial

$$\mathbf{AX} = \mathbf{B}$$

onde

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$$

\mathbf{A} é uma matriz quadrada e é conhecida como matriz-coeficiente, enquanto os vetores são \mathbf{X} e \mathbf{B} . \mathbf{X} é o vetor-solução que estamos procurando. Existem duas maneiras de se determinar \mathbf{X} no *MATLAB*. Em primeiro lugar, podemos usar o operador (\) de modo que

$$\mathbf{X} = \mathbf{A} \backslash \mathbf{B}$$

Em segundo lugar, podemos determinar \mathbf{X} como

$$\mathbf{X} = \mathbf{A}^{-1} \mathbf{B}$$

que no *MATLAB* é o mesmo que

$$\mathbf{X} = \text{inv}(\mathbf{A}) * \mathbf{B}$$

EXEMPLO 1

Use o *MATLAB* para resolver o Exemplo 2 que está no livro da quinta edição.

Solução: A partir do Exemplo 2, obtemos a matriz \mathbf{A} e o vetor \mathbf{B} , e os introduzimos no *MATLAB*, como segue:

```
>> A = [25 -5 -20; -5 10 -4; -5 -4 9]
A =
25 -5 -20
-5 10 -4
-5 -4 9
>> B = [50 0 0]'
B =
50
```

```

0
>> X = inv(A)*B
X =
29.6000
26.0000
28.0000
>> X = A\B
X =
29.6000
26.0000
28.0000
Portanto, x1 = 29.6, x2 = 26 e x3 = 28.

```

PROBLEMA PRÁTICO 1

Resolva o Problema prático 2 (que está na quinta edição do livro) usando o *MATLAB*.

Resposta: x₁ = 3 = x₃, x₂ = 2.

2 Análise de circuitos CC

Não há nada de especial em aplicar o *MATLAB* em circuitos resistivos. Aplicamos análise de malhas e nodal como de praxe e resolvemos as equações simultâneas resultantes utilizando o *MATLAB* como descrito na Seção 1. Os Exemplos 2 a 5 ilustram isso.

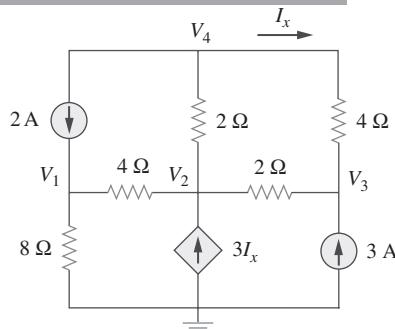
EXEMPLO 2

Figura 4 Esquema para o Exemplo 2.

Use análise nodal para resolver as tensões nodais no circuito da Figura 4.

Solução: No nó 1,

$$2 = \frac{V_1 - V_2}{4} + \frac{V_1 - 0}{8} \rightarrow 16 = 3V_1 - 2V_2 \quad (2.1)$$

No nó 2,

$$3I_x = \frac{V_2 - V_1}{4} + \frac{V_2 - V_3}{2} + \frac{V_2 - V_4}{2}$$

Porém,

$$I_x = \frac{V_4 - V_3}{4}$$

De modo que

$$3\left(\frac{V_4 - V_3}{4}\right) = \frac{V_2 - V_1}{4} + \frac{V_2 - V_3}{2} + \frac{V_2 - V_4}{2} \rightarrow \\ 0 = -V_1 + 5V_2 + V_3 - 5V_4 \quad (2.2)$$

No nó 3,

$$3 = \frac{V_3 - V_2}{2} + \frac{V_3 - V_4}{4} \rightarrow 12 = -2V_2 + 3V_3 - V_4 \quad (2.3)$$

No nó 4,

$$0 = 2 + \frac{V_4 - V_2}{2} + \frac{V_4 - V_3}{4} \rightarrow -8 = -2V_2 - V_3 + 3V_4 \quad (2.4)$$

Combinar as Equações (2.1) a (2.4) resulta em

$$\begin{bmatrix} 3 & -2 & 0 & 0 \\ -1 & 5 & 1 & -5 \\ 0 & -2 & 3 & -1 \\ 0 & -2 & -1 & 3 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} 16 \\ 0 \\ 12 \\ -8 \end{bmatrix}$$

ou

$$\mathbf{AV} = \mathbf{B}$$

Usamos, agora, o *MATLAB* para determinar as tensões nodais contidas no vetor \mathbf{V} .

```
>> A = [ 3 -2 0 0 ;
          -1 5 1 -5 ;
          0 -2 3 -1 ;
          0 -2 -1 3 ];
>> B = [16 0 12 -8]';
>> V = inv(A)*B
V =
-6.0000
-17.0000
-13.5000
-18.5000
```

Portanto, $V_1 = -6,0$, $V_2 = -17$, $V_3 = -13,5$ e $V_4 = -18,5$ V.

Determine as tensões nodais no circuito da Figura 5 usando o *MATLAB*.

PROBLEMA PRÁTICO 2

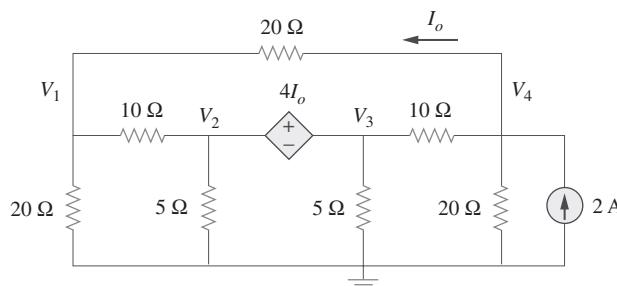


Figura 5 Esquema para o Problema prático 2.

Resposta: $V_1 = 14,55$, $V_2 = 38,18$, $V_3 = -34,55$ e $V_4 = -3,636$ V.

Use o *MATLAB* para determinar as correntes de malha no circuito da Figura 6.

EXEMPLO 3

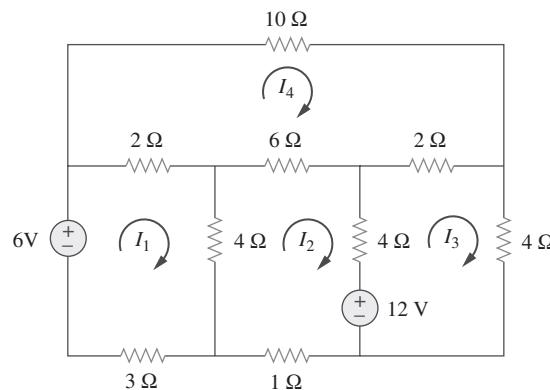


Figura 2.6 Esquema para o Exemplo 3.

Solução: Para as quatro malhas,

$$-6 + 9I_1 - 4I_2 - 2I_4 = 0 \longrightarrow 6 = 9I_1 - 4I_2 - 2I_4 \quad (3.1)$$

$$12 + 15I_2 - 4I_1 - 4I_3 - 6I_4 = 0 \longrightarrow \quad (3.2)$$

$$-12 = -4I_1 + 15I_2 - 4I_3 - 6I_4$$

$$-12 + 10I_3 - 4I_2 - 2I_4 = 0 \longrightarrow 12 = -4I_2 + 10I_3 - 2I_4 \quad (3.3)$$

$$20I_4 - 2I_1 - 6I_2 - 2I_3 = 0 \longrightarrow 0 = -2I_1 - 6I_2 - 2I_3 + 20I_4 \quad (3.4)$$

Colocando as Equações (3.1) a (3.4) juntas na forma matricial, temos

$$\begin{bmatrix} 9 & -4 & 0 & -2 \\ -4 & 15 & -4 & -6 \\ 0 & -4 & 10 & -2 \\ -2 & -6 & -2 & 20 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} 6 \\ -12 \\ 12 \\ 0 \end{bmatrix}$$

ou $\mathbf{AI} = \mathbf{B}$, onde o vetor \mathbf{I} contém as correntes de malha desconhecidas. Agora, usamos o *MATLAB* para determinar \mathbf{I} , como segue:

```
>> A = [9 -4 0 -2; -4 15 -4 -6;
          0 -4 10 -2; -2 -6 -2 20]
A =
      9   -4   0   -2
     -4   15   -4   -6
      0   -4   10   -2
     -2   -6   -2   20
>> B = [6 -12 12 0] '
B =
      6
     -12
      12
      0
>> I = inv(A)*B
I =
    0.5203
   -0.3555
    1.0682
    0.0522
```

Portanto, $I_1 = 0,5203$, $I_2 = -0,3555$, $I_3 = 1,0682$ e $I_4 = 0,0522$ A.

PROBLEMA PRÁTICO 3

Encontre as correntes de malha no circuito da Figura 7 usando o *MATLAB*.

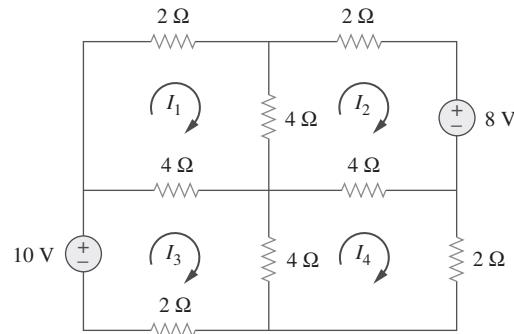


Figura 7 Esquema para o Problema prático 3.

Resposta: $I_1 = 0,2222$, $I_2 = -0,6222$, $I_3 = 1,1778$ e $I_4 = 0,2222$ A.

3 Análise de circuitos CA

Utilizar o *MATLAB* na análise de um circuito CA é similar ao uso do *MATLAB* na análise de circuitos CC. Primeiro, devemos aplicar análise nodal ou de malhas ao circuito e, em seguida, usar o *MATLAB* para resolver o sistema de equações. Porém, o circuito se encontra no domínio da frequência e estamos lidando com fasores ou números complexos. Portanto, além do que aprendemos na Seção 2, precisamos entender como o *MATLAB* trata os números complexos.

O *MATLAB* expressa números complexos da maneira usual, exceto pelo fato de a parte imaginária ser tanto j como i representando $\sqrt{-1}$. Consequentemente, $3 - j4$ pode ser escrito no *MATLAB* como $3 - j4$, $3 - j*4$, $3 - i4$ ou $3 - I*4$. A seguir, são apresentadas as demais funções complexas:

<code>abs(A)</code>	Valor absoluto da amplitude de A
<code>angle(A)</code>	Ângulo de A em radianos
<code>conj(A)</code>	Conjugado complexo de A
<code>imag(A)</code>	Parte imaginária de A
<code>real(A)</code>	Parte real de A

Tenha em mente que um ângulo em radianos deve ser multiplicado por $180/\pi$ para convertê-lo em graus e vice-versa. Da mesma forma, o operador de transposição ('') fornece a transposição do conjugado complexo, enquanto o operador ponto-transposição ('.') transpõe um array sem conjugá-lo.

EXEMPLO 4

No circuito da Figura 8, faça $v = 4 \cos(5t - 30^\circ)$ V e $i = 0,8 \cos 5t$ A. Determine v_1 e v_2 .

Solução: Como de praxe, convertemos o circuito no domínio do tempo para seu equivalente no domínio da frequência.

$$\begin{aligned} v &= 4 \cos(5t - 30^\circ) \longrightarrow \mathbf{V} = 4 \angle -30^\circ, \quad \omega = 5 \\ i &= 0,8 \cos 5t \longrightarrow \mathbf{I} = 8 \angle 0^\circ \\ 2 \text{ H} &\longrightarrow j\omega L = j5 \times 2 = j10 \\ 20 \text{ mF} &\longrightarrow \frac{1}{j\omega C} = \frac{1}{j10 \Omega \times 10^{-3}} = -j10 \end{aligned}$$

Portanto, o circuito equivalente no domínio da frequência é aquele apresentado na Figura 9. Aplicamos, agora, a análise nodal a ele.

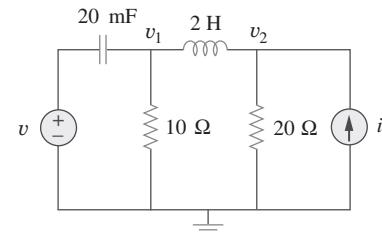


Figura 8 Esquema para o Exemplo 4.

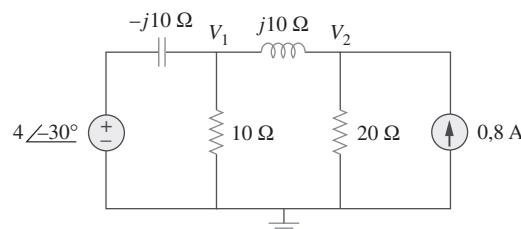


Figura 9 Circuito equivalente no domínio da frequência do circuito da Figura 8.

No nó 1,

$$\begin{aligned} \frac{4 \angle -30^\circ - V_1}{-j10} &= \frac{V_1}{10} + \frac{V_1 - V_2}{j10} \longrightarrow 4 \angle -30^\circ = 3,468 - j2 \\ &= -jV_1 + V_2 \quad (4.1) \end{aligned}$$

No nó 2,

$$0,8 = \frac{V_2}{20} + \frac{V_2 - V_1}{j10} \longrightarrow j16 = -2V_1 + (2 + j)V_2 \quad (4.2)$$

As Equações (4.1) e (4.2) podem ser formuladas na forma matricial, como segue

$$\begin{bmatrix} -j & 1 \\ -2 & (2 + j) \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 3,468 - j2 \\ j16 \end{bmatrix}$$

ou $\mathbf{AV} = \mathbf{B}$. Usamos o *MATLAB* para inverter \mathbf{A} e multiplicar a inversa por \mathbf{B} para obter \mathbf{V} .

```
>> A = [-j 1; -2 (2+j)]
A =
    0 - 1.0000i 1.000
   -2.0000 2.0000 + 1.000 i
>> B = [(3.468 - 2j) 16j].' %note the dot-transpose
B =
    3.4680 - 2.0000i
    0 + 16.0000i
>> V = inv(A)*B
V =
    4.6055 - 2.4403i
    5.9083 + 2.6055i
>> abs(V(1))
ans =
    5.2121
>> angle(V(1))*180/pi %converts angle from
radians to degrees
ans =
    -27.9175
>> abs(V(2))
ans =
    6.4573
>> angle(V(2))*180/pi
ans =
    23.7973
```

Portanto,

$$V_1 = 4,6055 - j2,4403 = 5,212 \angle -27,92^\circ$$

$$V_2 = 5,908 + j2,605 = 6,457 \angle 23,8^\circ$$

No domínio do tempo,

$$v_1 = 4,605 \cos(5t - 27,92^\circ) \text{ V}, \quad v_2 = 6,457 \cos(5t + 23,8^\circ) \text{ V}$$

Calcule v_1 e v_2 no circuito da Figura 10, dado $i = 4 \cos(10t + 40^\circ) \text{ A}$ e $v = 12 \cos 10t \text{ V}$.

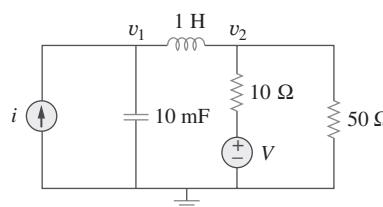


Figura 10 Esquema para o Problema prático 4.

Resposta: $63,58 \cos(10t - 10,68^\circ) \text{ V}$, $40 \cos(10t - 50^\circ) \text{ V}$.

PROBLEMA PRÁTICO 4

EXEMPLO 5

No sistema trifásico desequilibrado mostrado na Figura 11, determine as correntes I_1 , I_2 , I_3 e I_{Bb} . Faça

$$Z_A = 12 + j10 \Omega, \quad Z_B = 10 - j8 \Omega, \quad Z_C = 15 + j6 \Omega$$

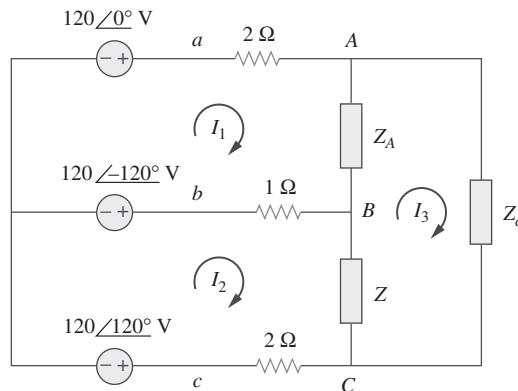


Figura 11 Esquema para o Exemplo 5.

Solução:

Para a malha 1,

$$120\angle-120^\circ - 120\angle0^\circ + I_1(2 + 1 + 12 + j10) - I_2 - I_3(12 + j10) = 0$$

ou

$$I_1(15 + j10) - I_2 - I_3(12 + j10) = 120\angle0^\circ - 120\angle-120^\circ \quad (5.1)$$

Para a malha 2,

$$120\angle120^\circ - 120\angle-120^\circ + I_2(2 + 1 + 10 - j8) - I_1 - I_3(10 - j8) = 0$$

ou

$$-I_1 + I_2(13 - j8) - I_3(10 - j8) = 120\angle-120^\circ - 120\angle120^\circ \quad (5.2)$$

Para a malha 3,

$$I_3(12 + j10 + 10 - j8 + 15 + j6) - I_1(12 + j10) - I_2(10 - j8) = 0$$

ou

$$-I_1(12 + j10) - I_2(10 - j8) - I_3(37 + j8) = 0 \quad (5.3)$$

Na forma matricial, podemos expressar as Equações (5.1) a (5.3) como

$$\begin{bmatrix} 15 + j10 & -1 & -12 - j10 \\ -1 & 13 - j8 & -10 + j8 \\ -12 - j10 & -10 + j8 & 37 + j8 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 120\angle0^\circ - 120\angle-120^\circ \\ 120\angle-120^\circ - 120\angle120^\circ \\ 0 \end{bmatrix}$$

ou

$$\mathbf{ZI} = \mathbf{V}$$

Introduzimos as matrizes \mathbf{Z} e \mathbf{V} no *MATLAB* para obter \mathbf{I} .

```

>> z = [(15 + 10j) -1 (-12 - 10j);
         -1 (13 - 8j) (-10 + 8j);
         (-12 - 10j) (-10 + 8j) (37 + 8j)];
>> c1=120*exp(j*pi*(-120)/180);
>> c2=120*exp(j*pi*(-120)/180);
>> a1=120 - c1; a2=c1 - c2;
>> V = [a1; a2; 0]
>> I = inv(z)*V
I=
16.9910 - 6.5953i
12.4023 - 16.9993i
5.6621 - 6.0471i
>> IbB = I(2) - I(1)
IbB =
-4.5887 - 10.4039i
>> abs(I(1))
ans =
18.2261
>> angle(I(1))*180/pi
ans =
-21.2146
>> abs(I(2))
ans =
21.0426
>> angle(I(2))*180/pi
ans =
-53.8864
>> abs(I(3))
ans =
8.2841
>> angle(I(3))*180/pi
ans =
-46.8833
>> abs(IbB)
ans =
11.3709
>> angle(IbB)*180/pi
ans =
-113.8001

```

Portanto, $I_1 = 18,23\angle-21,21^\circ$, $I_2 = 21,04\angle-58,89^\circ$,
 $I_3 = 8,284\angle-46,88^\circ$ e $I_{bB} = 11,37\angle-113,8^\circ A$.

PROBLEMA PRÁTICO 5

No sistema trifásico estrela-estrela da Figura 12, determine as correntes de linha I_1 , I_2 e I_3 e a tensão de fase V_{CN} .

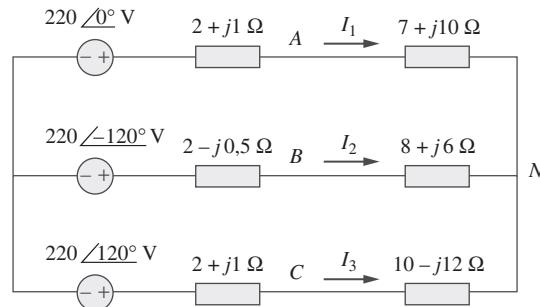


Figura 12 Esquema para o Problema prático 5.

Resposta: $22,66 \angle -26,54^\circ A$, $6,036 \angle -150,48^\circ A$, $19,93 \angle 138,9^\circ A$,
 $94,29 \angle 159,3^\circ V$.

4 Resposta de frequência

A resposta de frequência envolve colocar em gráfico a amplitude e a fase da função de transferência $H(s) = D(s)/N(s)$ ou obter os gráficos de Bode de amplitude e fase de $H(s)$. Uma forma trabalhosa de obter esses gráficos é gerar dados usando o *loop for* para cada valor de $s = j\omega$ para um dado intervalo de ω , e então colocar os dados em um gráfico, como fizemos na Seção 1. Entretanto, existe uma maneira mais fácil que nos permite usar um dos dois comandos do *MATLAB*: *freqs* e *bode*. Para cada um desses comandos, temos de especificar, primeiro, $H(s)$ na forma *num* e *den*, onde *num* e *den* são os vetores dos coeficientes do numerador $N(s)$ e o denominador $D(s)$ em potências decrescentes de s , isto é, da maior potência para o termo constante. A forma geral da função de Bode é

```
bode(num, den, range);
```

onde *range* é um intervalo de frequências especificado para o gráfico. Se *range* for omitido, o *MATLAB* selecionará automaticamente o intervalo de frequências. *range* pode ser linear ou logarítmico. Por exemplo, para $1 < \omega < 1.000$ rad/s e 50 pontos colocados em gráfico, especificamos um intervalo linear *range*, como segue

```
range = linspace(1,1000,50);
```

Para *range* logarítmico com $10^{-2} < \omega < 10^4$ rad/s e 100 pontos colocados em gráfico, no intervalo, especificamos *range* como

```
range = logspace(-2,4,100);
```

Para a função *freqs*, a forma geral é

```
hs = freqs(num, den, range);
```

onde *hs* é a resposta de frequência (geralmente, complexa). Precisamos calcular ainda a amplitude em decibéis, como segue

```
mag = 20*log10(abs(hs))
```

e a fase em graus como

```
phase = angle(hs)*180/pi
```

e colocá-los em um gráfico, enquanto a função *bode* faz tudo isso de uma vez só. Ilustraremos isso por meio de um exemplo.

EXEMPLO 6

Use o *MATLAB* para obter os gráficos de Bode de

$$G(s) = \frac{s^3}{s^3 + 14,8s^2 + 38,1s + 2.554}$$

Solução: Com a explicação dada anteriormente, desenvolvemos o código *MATLAB* conforme mostrado aqui.

```
% para o exemplo 6
num=[1 0 0 0];
den = [1 14.8 38.1 2554];
w = logspace(-1,3);
bode(num, den, w);
title('gráfico de Bode para um filtro passa-altas')
```

Executando o programa, obtemos os gráficos de Bode da Figura 13. Fica evidente no gráfico de amplitude que $G(s)$ representa um filtro passa-altas.

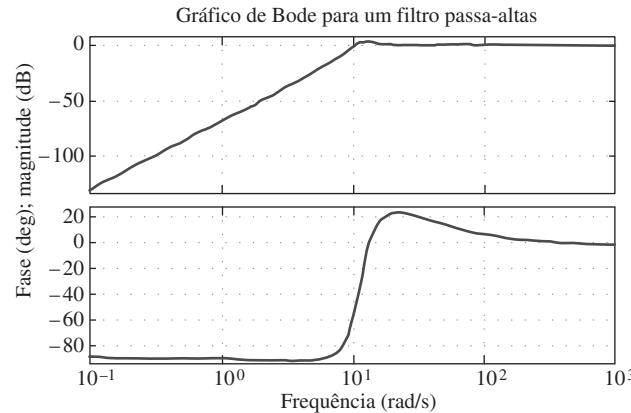


Figura 13 Esquema para o Exemplo 6.

PROBLEMA PRÁTICO 6

Use o *MATLAB* para determinar a resposta de frequência de

$$H(s) = \frac{10(s + 1)}{s^2 + 6s + 100}$$

Resposta: Ver Figura 14.

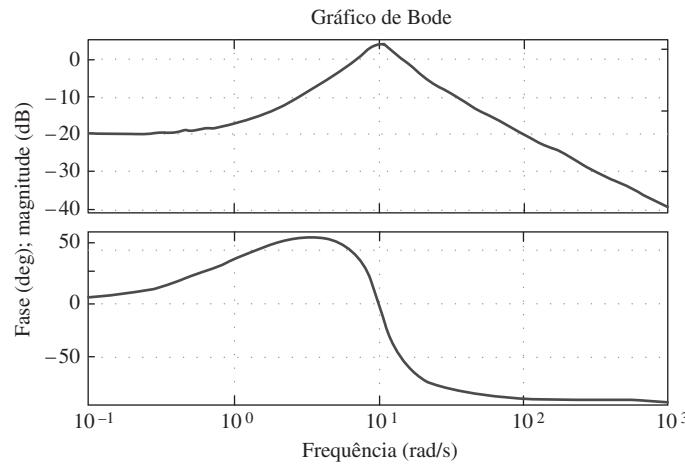


Figura 14 Esquema para o Problema prático 6.