5 FUNÇÕES

Um importante recurso do MATLAB é a possibilidade de criação de funções através de arquivos externos ao ambiente de cálculo, as quais podemos denominar de funções matemáticas. Estas funções podem incorporar na sua estrutura as operações matriciais básicas vistas anteriormente acrescidas de comandos de controle, entre outros. As funções, que diferem de um *script* ou roteiro, necessitam de um parâmetro de entrada e retornam um ou mais parâmetros para o ambiente. Normalmente as funções são agrupadas em diretórios específicos, constituindo-se assim os chamados *toolbox* ou caixa de ferramentas. Como exemplo, podemos citar as seguintes funções já incorporadas ao MATLAB:

- Integração numérica;
- Equações não-lineares e otimização;
- Solução de equações diferenciais.

As funções matemáticas são representadas no MATLAB por arquivos ".m". Por exemplo, a função

$$humps\,(x) = \frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.04} - 6$$

está disponível no MATLAB como um arquivo ".m" chamado **humps.m**:

```
function y = humps(x)
y = l ./ ((x-.3).^2 + 0.0l) + 1./((x-.9).^2 + 0.04) - 6;
```

O gráfico da função é obtido com os seguintes comandos :

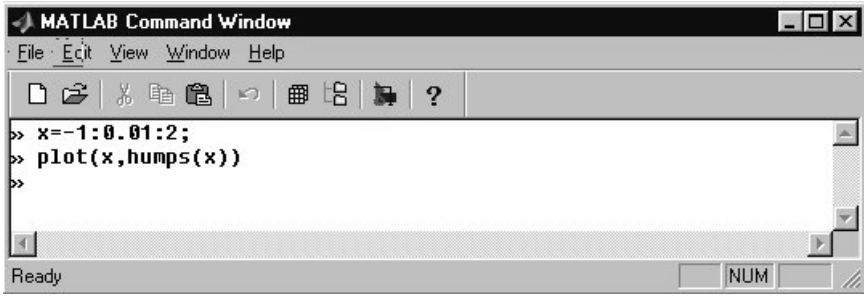

Fig. 5.1 – Comandos para gerar o gráfico da função *humps*.

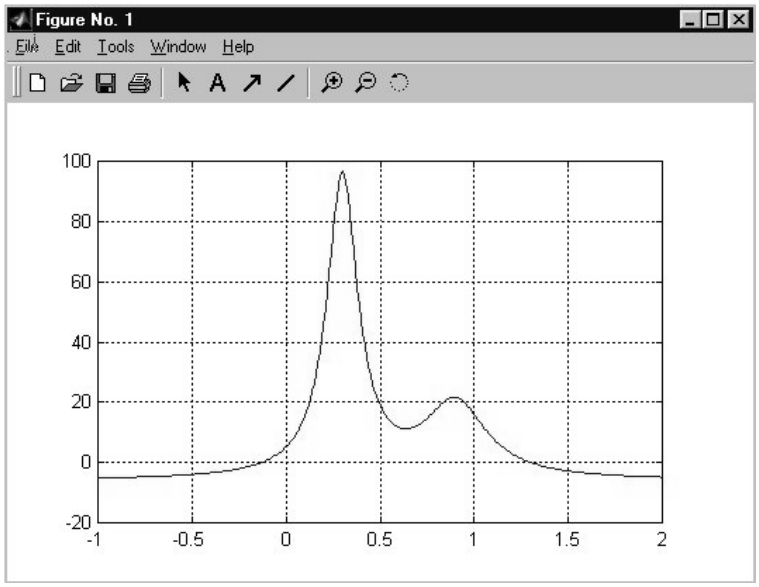

Fig. 5.2 – Gráfico da função *humps*.

5.1 Integração Numérica

A área sob a curva pode ser determinada através da integração numérica da função *humps(x)*, usando o processo chamado *quadratura*. Integrando a função *humps(x)* no intervalo de -1 a 2:

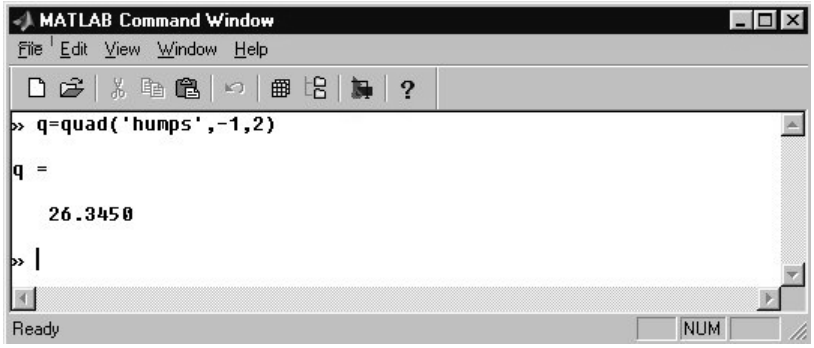

Fig. 5.3 – Resultado da integração da função *humps*.

Para o cálculo numérico das integrais, usando quadratura, temos os seguintes comandos:

quad	Calcula numericamente a integral, método para baixa ordem.
quad8	Calcula numericamente a integral, método para alta ordem.
dblquad	Calcula numericamente a integral dupla.

5.2 Equações Não-Lineares e Otimização

Os comandos para equações não-lineares e otimização incluem:

fmin	Minimizar função de uma variável.
fmins	Minimizar função de várias variáveis
fzero	Encontrar zero de função de uma variável.

para versões anteriores a 5.3R11, na qual as funções foram substituídas por:

fminbnd	Minimizar função de uma variável.
fminsearch	Minimizar função de várias variáveis
fzero	Encontrar zero de função de uma variável.

Assim, a localização do mínimo da função *humps(x)* no intervalo de [0.5 1] é obtido da seguinte maneira,

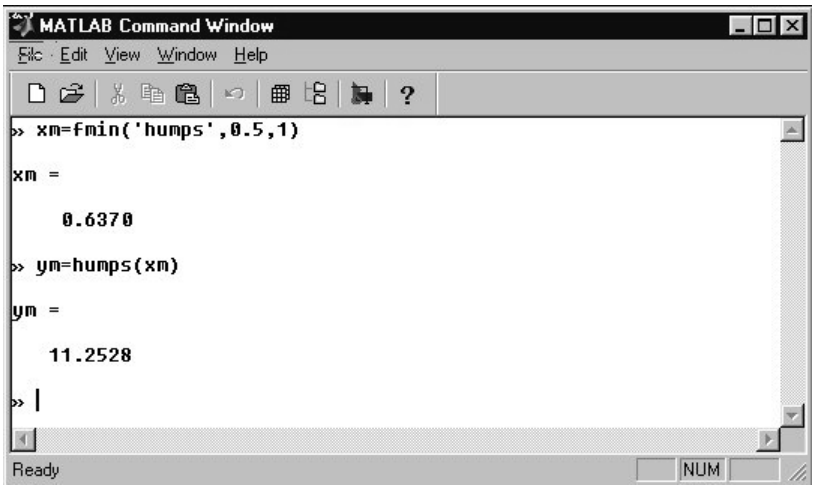

Fig. 5.4 – Cálculo do valor mínimo da função *humps*.

E o gráfico deste intervalo com o ponto de mínimo pode ser construído:

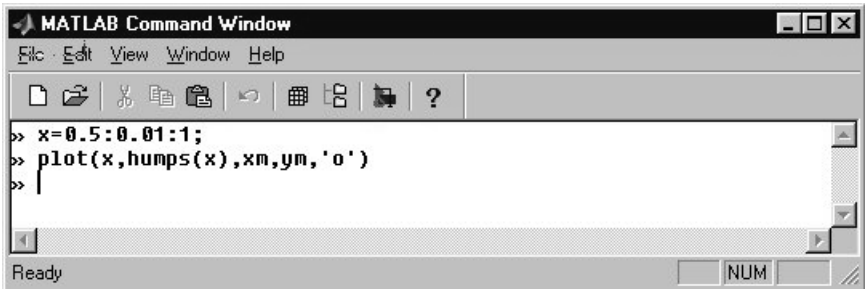

Fig. 5.5 – Comandos para exibir o valor mínimo da função *humps*.

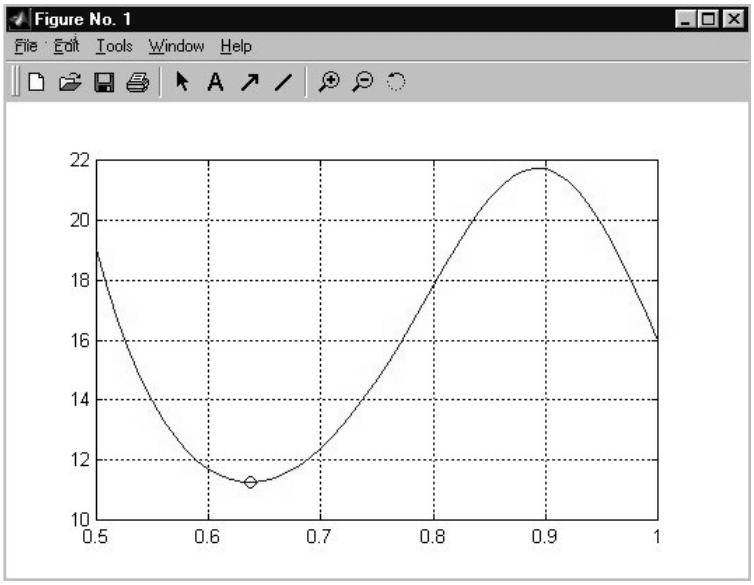

Fig. 5.6 – Ponto de mínimo para a função *humps*.

A função *humps(x)* (Fig. 5.2) apresenta dois **"**zeros" no intervalo de [-1 2]. A localização do primeiro "zero" é próxima do ponto x = 0,

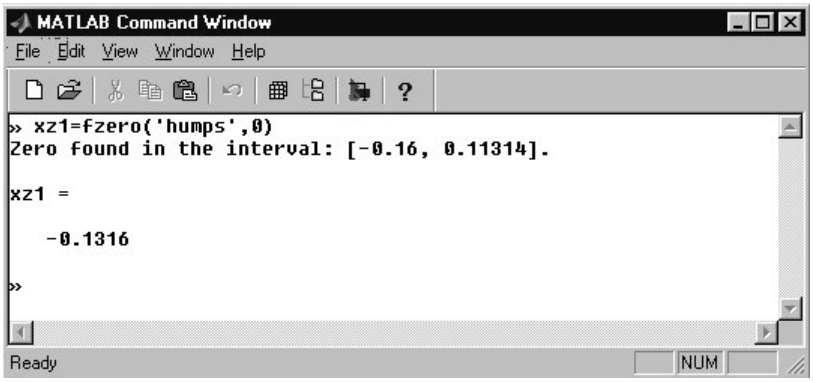

Fig. 5.7 – Localização do primeiro zero da função *humps*.

e a localização do segundo "zero" é próxima do ponto x= 1,

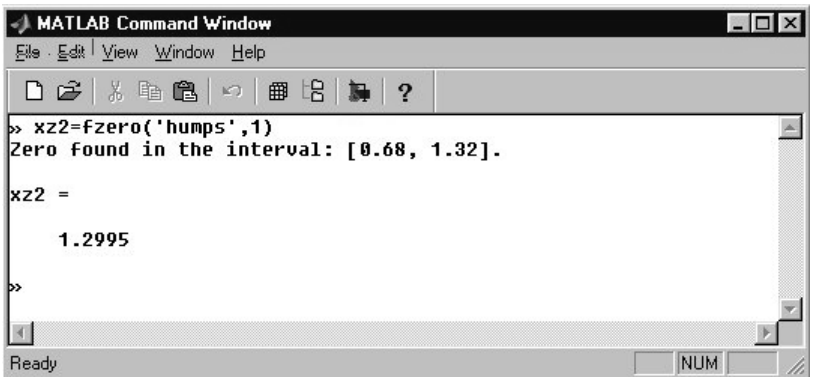

Fig. 5.8 – Localização do segundo zero da função *humps*.

O gráfico da função com os dois "zeros" é obtido através da expressão:

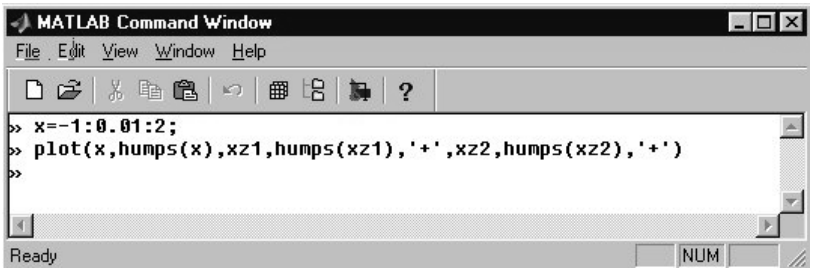

Fig. 5.9 – Comando para exibir os zeros da função *humps*.

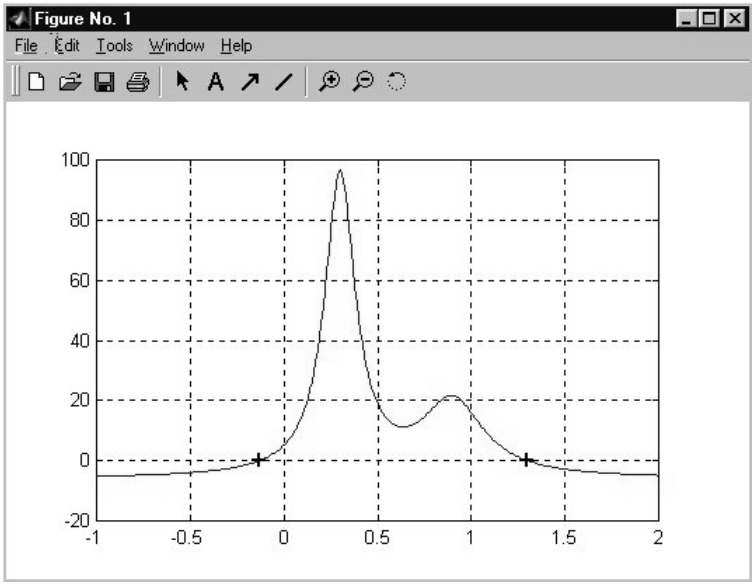

Fig. 5.10 – Gráfico dos zeros da função *humps*.

5.3 Encontrando um determinado valor

A determinação dos valores das raízes (zeros da função) de funções não-lineares, como a função *humps*, é um dos mais importantes problemas na matemática, tanto do ponto de vista de apenas encontrar as raízes, como também integrado na solução de muitos outros problemas. As soluções apresentadas, são

soluções numéricas, do tipo iterativa, exigindo uma estimativa inicial ou o intervalo que contém a raiz. São exemplos de algoritmos iterativos para o cálculo de raízes: método da Bisseção (Bolzano), método da tangente (Newton-Raphson) e o método de quebra (Golden Search), entre outros.

Uma forma alternativa é a inspeção da função dentro de um determinado domínio ou intervalo. A partir dos valores obtidos (calculados) para a função, realiza-se uma varredura em busca de mínimos, máximos ou zeros da função. O ambiente MATLAB oferece um conjunto de funções que possibilita realizar essa busca de uma forma rápida e eficiente, bem como outras funções de análise. Algumas funções relacionadas são:

max	Retorna o elemento de maior valor.
min	Retorna o elemento de menor valor.
mean	Determina o valor médio.
median	Determina o valor mediano.
std	Determina o desvio padrão.
var	Determina a variância.
sort	Organiza os elementos em ordem ascendente.
sortrows	Organiza os elementos das linhas em ordem ascendente.
sum	Somatório dos elementos.
prod	Produto dos elementos.
hist	Histograma.
trapz	Integração numérica trapezoidal.
cumsum	Soma acumulativa dos elementos.
cumprod	Produto acumulativo dos elementos.

reunidas no *\toolbox\matlab\datafun*.

Associado a esse conjunto de funções temos uma importante função de manipulação de dados denominada de *find* encontrada no diretório *\toolbox\matlab\elmat*. A sintaxe do comando é dada por:

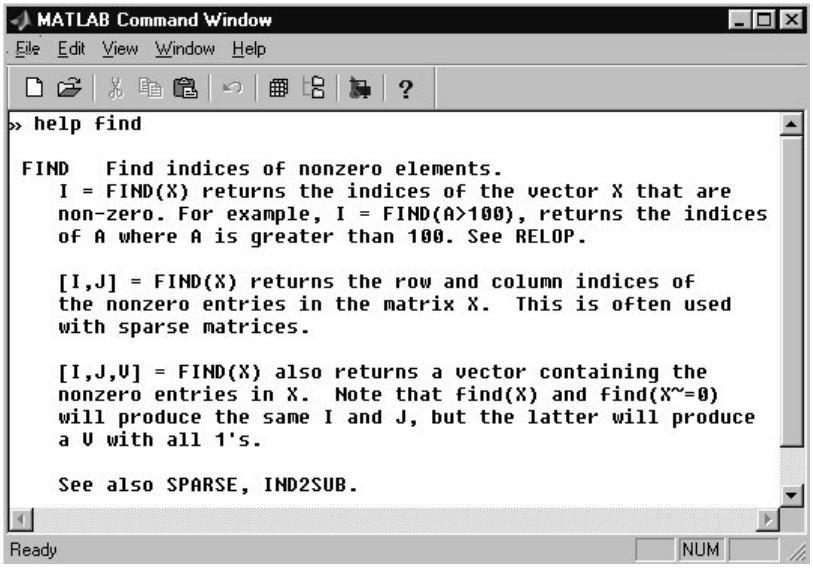

Fig. 5.11 – Sintaxe do comando *find*.

Assim, a busca pelo valor mínimo do função *humps* no intervalo [0.5 1] poderia ser realizada da seguinte forma:

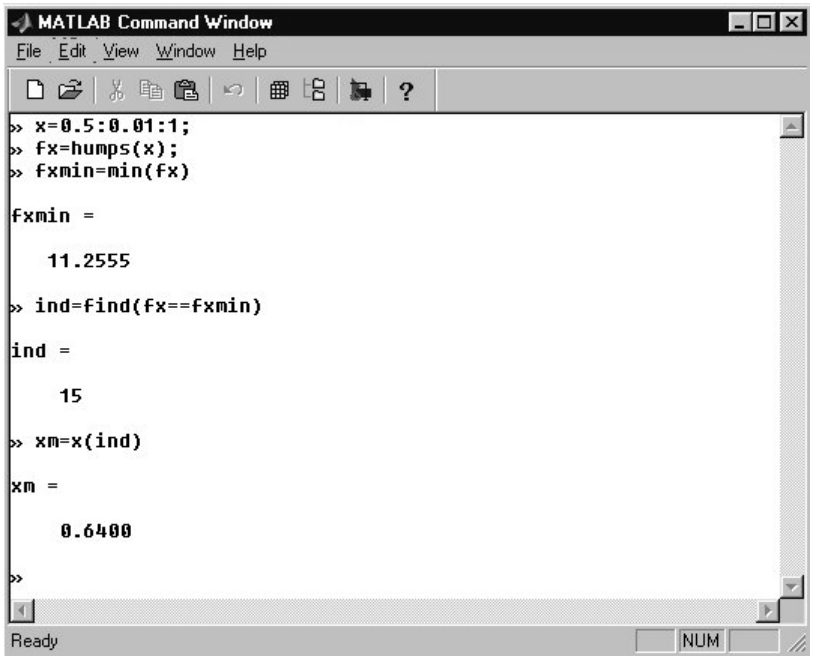

Fig. 5.12 – Forma alternativa de encontrar o mínimo da função *humps*.

Considerando que existe um erro associado aos dois processos, uma vez que são aproximados, pode-se dizer sem sombra de dúvida que os resultados são os mesmos. A principal característica neste exemplo é o uso da função *find*, a qual foi utilizada para encontrar, no vetor de dados, um valor que seja igual ao valor mínimo encontrado pela função *min*. Outro modo de obter esse mesmo índice é o uso da própria função *min*, como mostrado abaixo:

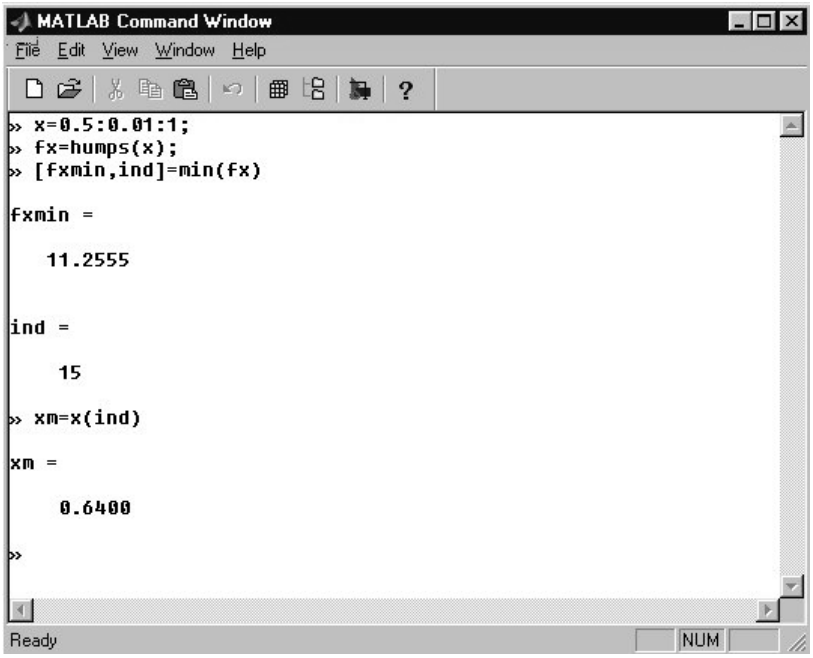

Fig. 5.13 – Estendendo o uso da função *min*.

6 GRÁFICOS

A construção de gráficos no MATLAB é mais uma das facilidades do sistema. Através de comandos simples pode-se obter gráficos bidimensionais ou tridimensionais com qualquer tipo de escala e coordenada. Existe no MATLAB uma vasta biblioteca de comandos gráficos.

6.1 Gráficos Bidimensionais

Estes são os comandos básicos para gráficos bidimensionais:

plot	Gráfico linear.
loglog	Gráfico em escala logarítmica.
semilogx	Gráfico logarítmico em x.
semilogy	Gráfico logarítmico em y.
fill	Desenhar polígono 2D.
polar	Gráficos em coordenadas polares.
bar	Gráfico de barras.
stem	Gráfico de uma seqüência discreta.
stairs	Gráfico em degrau.
errorbar	Gráfico erro.
hist	Gráfico em histograma.
rose	Gráfico histograma em ângulo.
compass	Gráfico em forma de bússola.
feather	Gráfico em forma de pena.
fplot	Gráfico de uma função.
comet	Gráfico com trajetória de cometa.

Se **Y** é um vetor, **plot(Y)** produz um gráfico linear dos elementos de **Y** versos o índice dos elementos de **Y.** Por exemplo, para plotar os números [0.0, 0.48, 0.84, 1.0, 0.91, 0.6, 0,14], entre com o vetor e execute o comando **plot:**

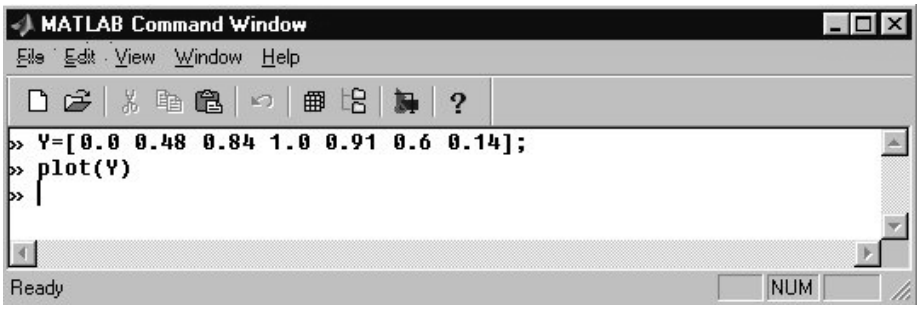

Fig. 6.1 – O comando *plot*.

e o resultado é mostrado na Janela Gráfica:

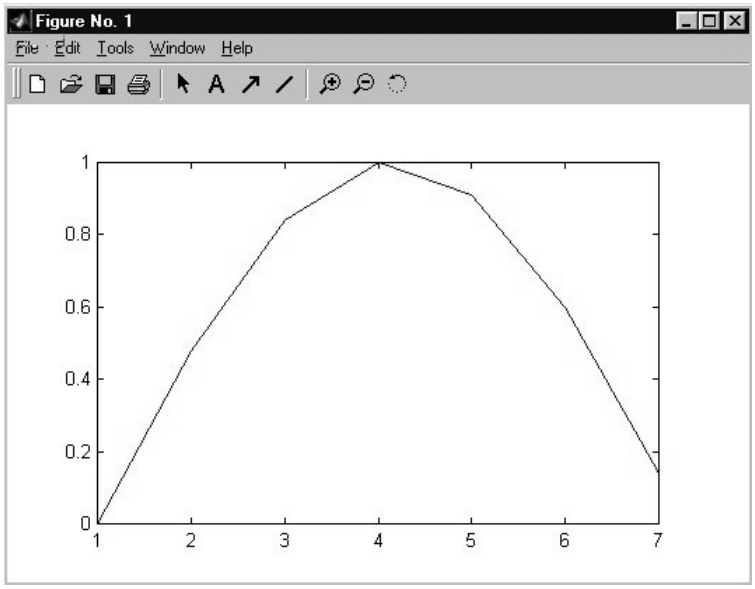

Fig. 6.2 – Resultado do comando *plot*.

Se **X** e **Y** são vetores com dimensões iguais, o comando **plot(X,Y)** produz um gráfico bidimensional dos elementos de **X** versus os elementos de **Y**, por exemplo:

Fig. 6.3 – Comando *plot* com vetores X e Y.

resulta em:

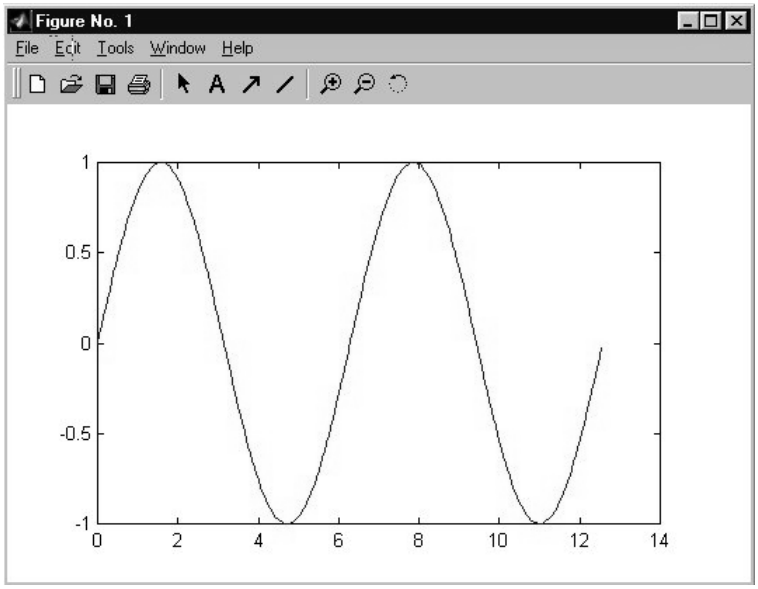

Fig. 6.4 – Resultado gráfico da função seno.

O MATLAB pode também exibir múltiplas linhas em apenas um gráfico. Existem duas maneiras, a primeira é usado apenas dois argumentos, como em **plot(X,Y)**, onde **X** e/ou **Y** são vetores ou matrizes. Então:

- Se **Y** é uma matriz e **X** um vetor, **plot(X,Y)** plota sucessivamente as linhas ou colunas de **Y** versos o vetor **X**.
- Se **X** é uma matriz e **Y** é um vetor, **plot(X,Y)** plota sucessivamente as linhas ou colunas de **X** versos o vetor **Y**.
- Se **X** e Y são matrizes com mesma dimensão, **plot(X,Y)** plota sucessivamente as colunas de **X** versos as colunas de **Y**.
- Se **Y** é uma matriz, **plot(Y)** plota sucessivamente as colunas de **Y** versos o índice de cada elemento da linha de **Y**.

A segunda maneira de exibir gráficos com múltiplas linhas é usando o comando **plot** com múltiplos argumentos. Por exemplo:

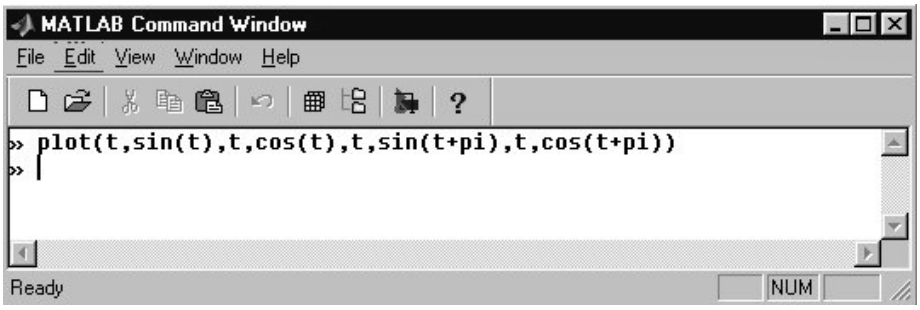

Fig. 6.5 – Comando plot com múltiplas entradas.

o que resulta em:

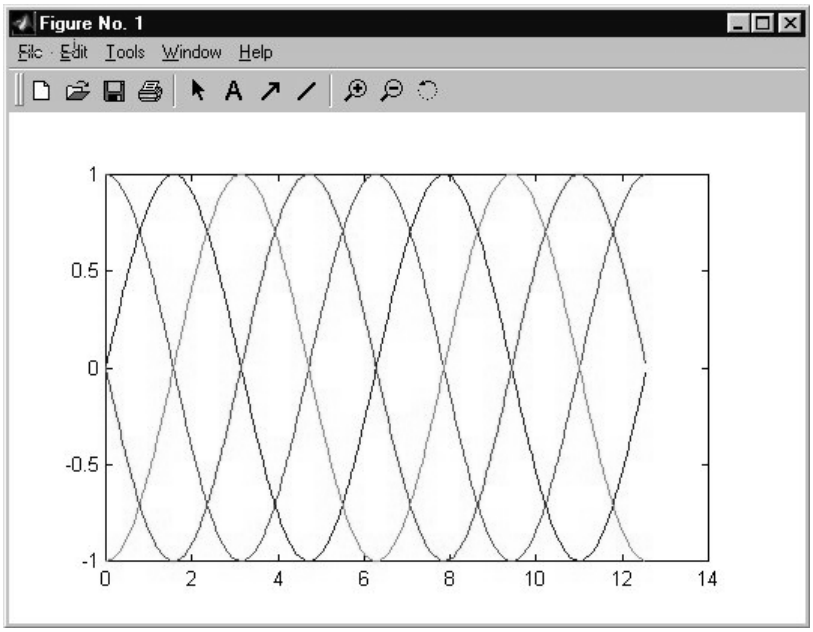

Fig. 6.6- Resultado gráfico de múltiplas entradas.

6.2 Dividindo uma janela gráfica

Muitas vezes é necessário mostrar vários resultados gráficos para que se tenha uma idéia geral dos resultados. Umas opção seria abrir mais de uma janela gráfica, mas existe o inconveniente de ter que mudar de janela. Uma forma alternativa e utilizar a mesma janela gráfica, mas com uma divisão do espaço, de forma a reunir as diversas saídas. Esta divisão é realizada através do comando *subplot* que divide a janela gráfica em regiões cujas coordenadas são do tipo linha x coluna, preparando para que a próxima saída do comando *plot* seja ativada em uma destas regiões. Supondo que desejamos mostrar os resultados anteriores em quatro gráficos, teremos a seguinte seqüência de comandos:

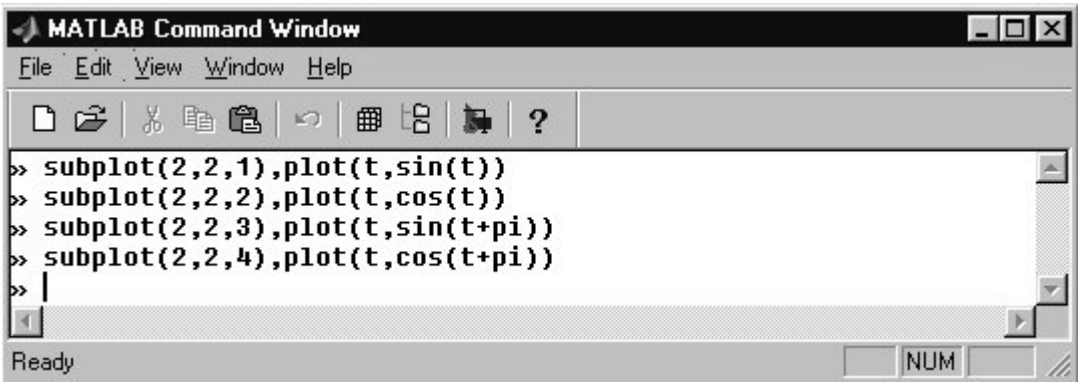

Fig. 6.7 – Dividindo uma janela gráfica via *subplot*.

e o resultado será:

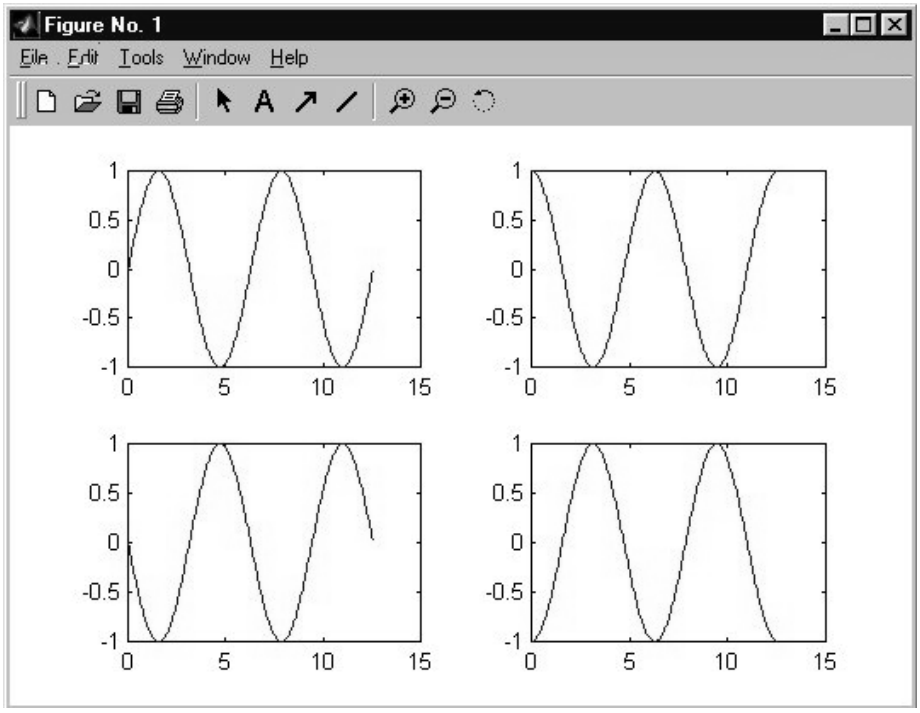

Fig. 6.8 – Resultado gráfico através do comando *subplot*.