6.3 Estilos de Linha e Símbolos

Os tipos de linhas, símbolos e cores usados para traçar gráficos podem ser controlados se os padrões não são satisfatórios. Por exemplo,

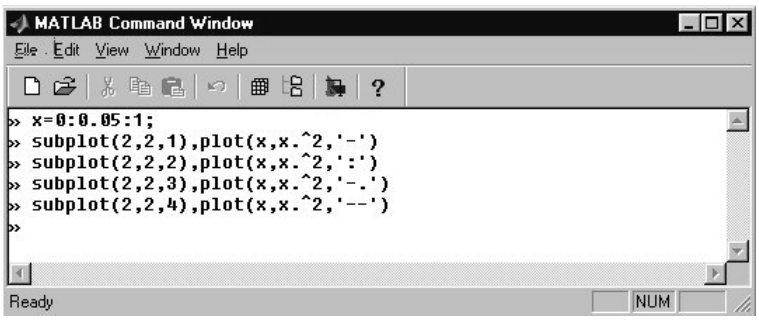

Fig. 6.9 – Graficando com diferentes estilos de linhas.

mostra na saída gráfica os resultados para o mesmo vetor de dados, traçados com estilos de linhas diferentes: contínua, pontilhada, traço-ponto e tracejada, respectivamente. O ambiente assume como padrão o estilo de linha contínuo.

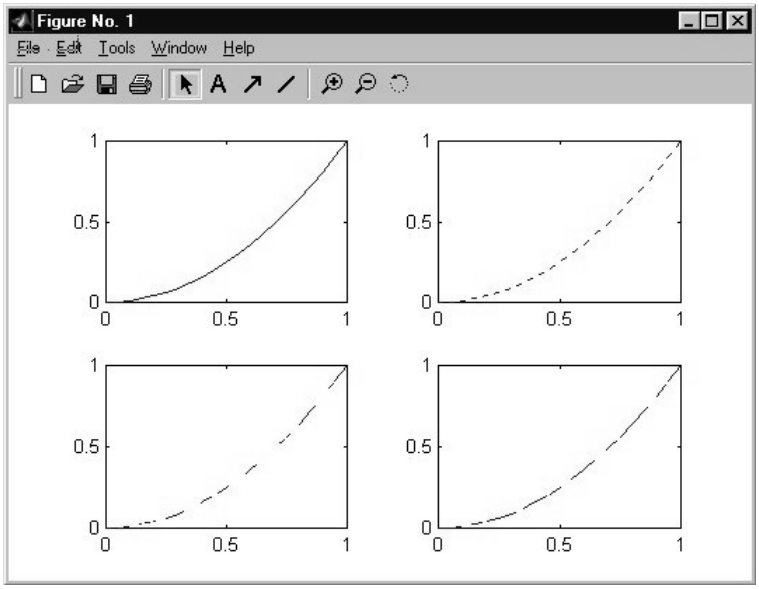

Fig. 6.10 – Gráficos com estilos de linhas diferentes.

Além dos estilos de linhas proporcionados pelo sistema, pode-se ainda utilizar o recurso de desenhar com símbolos os dados que serão mostrados na janela gráfica, com as seguintes definições: ponto, círculo, marca x, marca +, estrela, quadrado, diamante, triângulo p/ baixo, triângulo p/ cima, triângulo p/ esquerda, pentagrama e hexagrama, com a sintaxe mostrada abaixo.

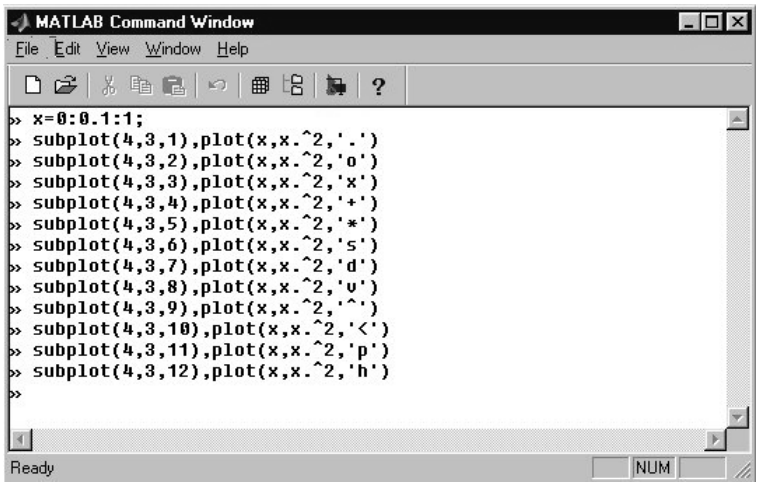

Fig. 6.11 – Desenhando com estilos de pontos.

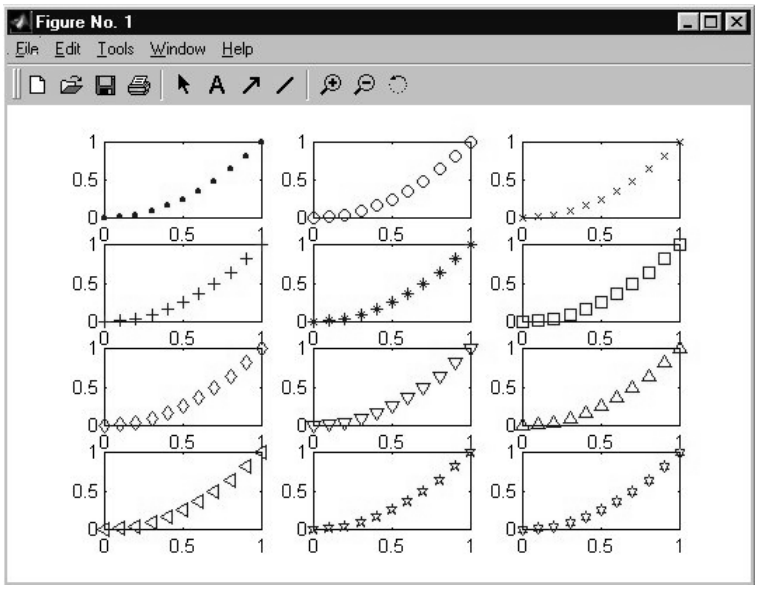

Fig. 6.12 – Resultados para diferentes estilos de pontos.

Associado aos estilos de linhas e pontos, pode-se ainda modificar as cores que serão atribuídas os elementos na janela gráfica, conforme a tabela abaixo.

CORES	
y	amarelo
m	lilás
c	azul claro
r	vermelho
g	verde
b	azul escuro
w	branco
k	preto

6.4 Valores Complexos

Quando os argumentos para traçar o gráfico são complexos, a parte imaginária é ignorada, exceto quando é dado simplesmente um argumento complexo. Para este caso especial é realizado um traçado da parte real versus a parte imaginária. Então, **plot(Z)**, quando **Z** é um vetor complexo, é equivalente a **plot(real(Z), imag(Z))**.

6.5 Escala Logarítmica, Coordenada Polar e Gráfico de Barras

O uso de **loglog**, **semilogx**, **semilogy** e **polar** é idêntico ao uso de **plot**. Estes comandos são usados para traçar gráficos em diferentes coordenadas e escalas:

- **polar(Theta,R):** realiza um traçado em coordenadas polares do ângulo **THETA**, em radianos, versus o raio **R**;
- **loglog:** cria um gráfico usando a escala $\log_{10}$ x $\log_{10}$;
- **semilogx:** cria um gráfico usando a escala semi-logarítmica. O eixo x é $\log_{10}$ e o eixo y é linear;
- **semilogy:** cria um gráfico usando a escala semi-logarítmica. O eixo x é linear e o eixo y é $\log_{10}$;

O comando **bar(X)** mostra um gráfico de barras dos elementos do vetor **X**, e não aceita múltiplos argumentos.

6.6 Plotando Gráficos Tridimensionais e Contornos

Estes são alguns comandos para traçar gráficos tridimensionais e contornos.

Plot3	Desenha no espaço 3D.
fill3	Desenhar polígono 3D.
comet3	Desenha em 3D com trajetória de cometa.
contour	Desenha contornos 2D.
contour3	Desenha contornos 3D.
clabel	Desenha contornos com valores.
quiver	Desenha gradiente.
mesh	Desenha malha 3D.
meshc	Combinação mesh/contour.
surf	Desenha superfície 3D.
surfc	Combinação surf/contour.
surfl	Desenha superfície 3D com iluminação.
slice	Desenha visualização volumétrica.
cylinder	Gerar cilindro.
sphere	Gerar esfera.

O comando **mesh(X,Y,Z)** cria uma perspectiva tridimensional desenhando os elementos da matriz **Z** em relação ao plano definindo pelas matrizes **X** e **Y**. Por exemplo,

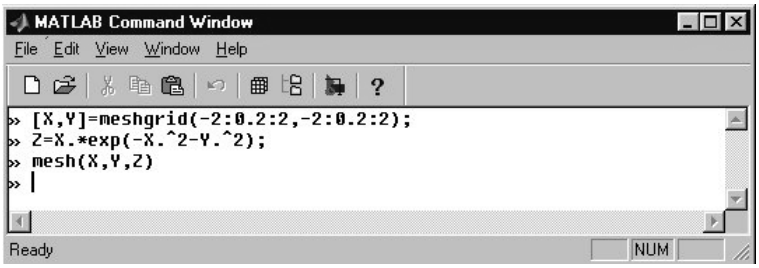

Fig. 6.13 – Comandos para traçar um gráfico 3D.

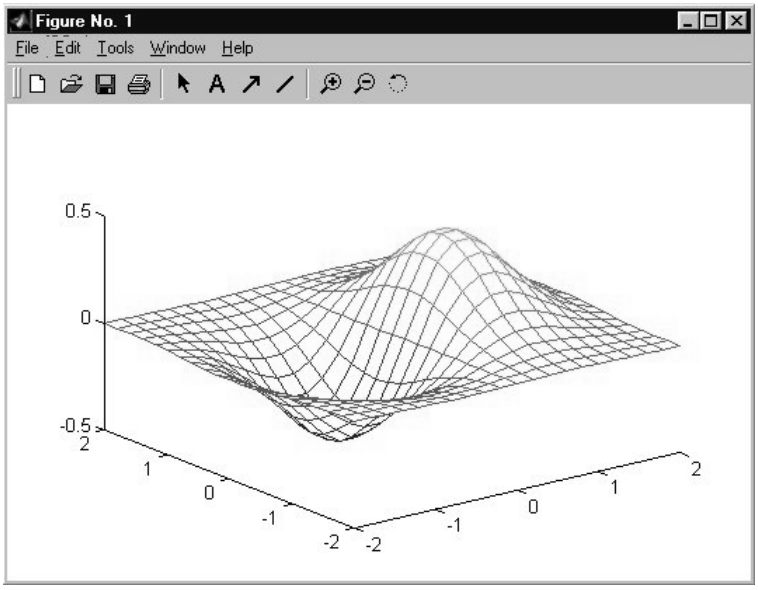

Fig. 6.14 – Resultado gráfico do comando *mesh*.

e o comando **contour(Z,10)** mostra a projeção da superfície acima no plano xy com 10 iso-linhas:

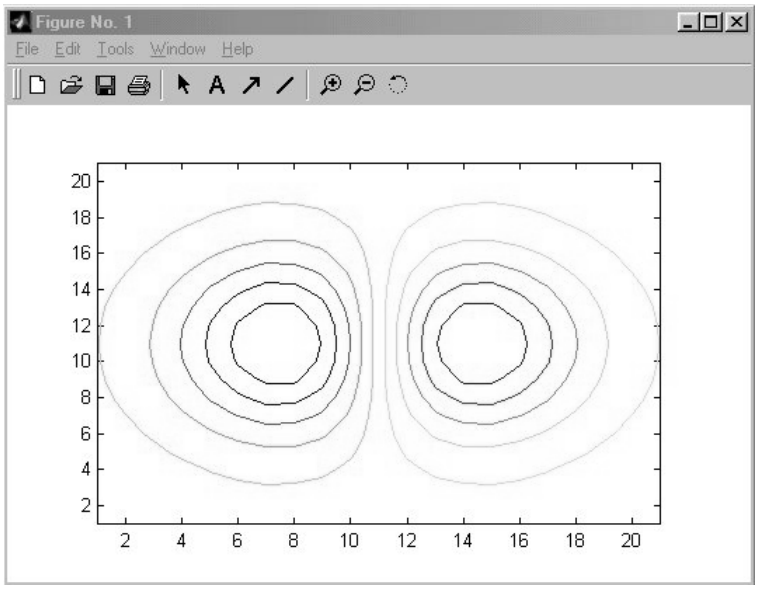

Fig. 6.15 – Resultado gráfico do comando *contour*.

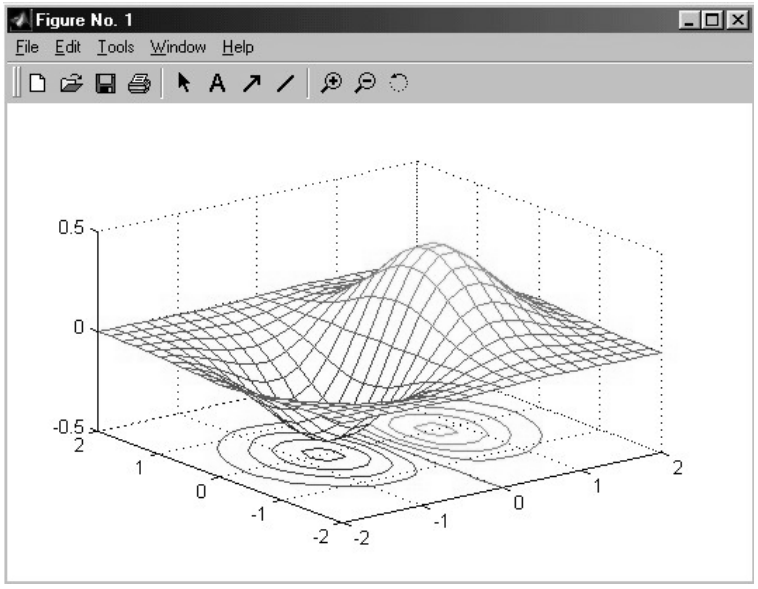

Fig. 6.16 – Resultado gráfico do comando *meshc*.

6.7 Anotações no Gráfico

O MATLAB possui comandos de fácil utilização para adicionar informações em um gráfico:

title	Título do gráfico.
xlabel	Título do eixo-X.
ylabel	Título do eixo-Y.
zlabel	Título do eixo-Z.
text	Inserir anotação no gráfico.
gtext	Inserir anotação com o “mouse”.
grid	Linhas de grade.

Por exemplo:

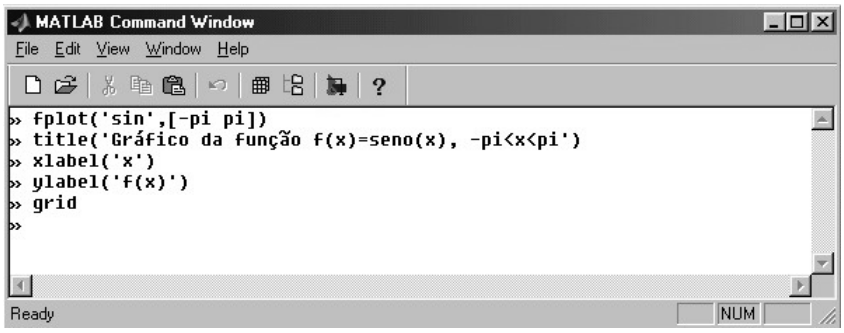

Fig. 6.17 – Comandos de anotações no gráfico.

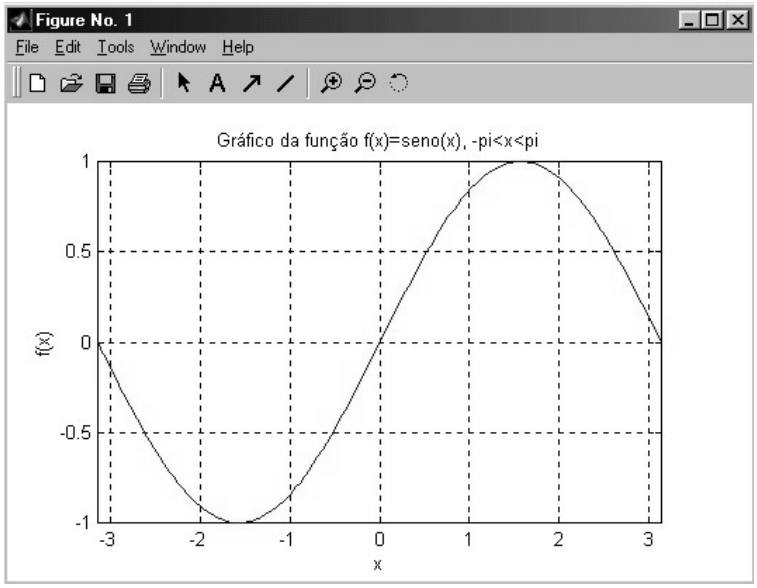

Fig. 6.18 – Exemplo de anotações na janela gráfica.

7 CONTROLE DE FLUXO

Os comandos que controlam o fluxo especificam a ordem em que a computação é feita. No MATLAB estes comandos são semelhantes aos usados na linguagem C, mas com uma estrutura diferente.

7.1 Comando de repetição ou laço

A repetição é realizada através do laço ***for***, o qual é o controlador de fluxo mais simples usado na programação MATLAB. Analisando a expressão:

```
>>for i=1:5,
```

```
    X(i)=i^2;
end
```

O laço **for** é dividido em três partes:

- A primeira parte (**i=1**) é realizada uma vez, antes do laço ser inicializado.
- A segunda parte é o teste ou condição que controla o laço, (**i<=5**). Esta condição é avaliada; se verdadeira, o corpo do laço (**X(i)=i^2**) é executado.
- A terceira parte acontece quando a condição se torna falsa e o laço termina.

O comando ***end*** é usado como limite inferior do corpo do laço.

É comum construções em que conjuntos de laços ***for*** são usados principalmente com matrizes:

```
for i=1:8
for j=1:8,
        A(i,j)=i+j;
        B(i,j)=i-j;
end
end
C=A+B;
```

7.2 Comando de repetição condicional

No laço ***while*** apenas a condição é testada. Por exemplo na expressão

```
a = l; b = 15;
while a<b,
            clc
            a = a+l
            b = b-l
            pause(l)
end
disp('fim do loop')
```

a condição **a<b** é testada inicialmente. Se ela for verdadeira o corpo do laço, será executado.

Na próxima iteração a condição é testada novamente, e se verdadeira o corpo será executado. Quando o teste se tornar falso o laço terminará, e a execução continuará no comando que segue o laço após o ***end***.

7.3 Comando condicional e parada

A seguir, é apresentado um exemplo do uso da declaração ***if*** no MATLAB.

```
for i = l:5,
    for j = l:5,
        if i = = j
            A(i,j) = 2;
        elseif abs(i-j) = = 1
            A(i,j) = -1;
        else
            A(i,j) = 0;
        end
    end
end
```

Os valores de **i e j** variam de 1 a 5, varrendo toda a matriz **A.** Se **(*if*) i** for igual a **j**, **A(i,j)=2**, ou se **(*elseif*)** o valor absoluto de **i-j** for igual a 1, **A(i,j)=-1**, ou **(*else*) A(i,j)=0**, se nenhuma das condições anteriores forem satisfeitas.

Muitas vezes é conveniente controlar a saída de um laço de outro modo além do teste, no início ou no fim do mesmo. O comando ***break*** permite uma saída antecipada de um ***for*** ou ***while*.** Um comando ***break*** faz com que o laço mais interno seja terminado imediatamente. Por exemplo,

```
%modifica a matriz A
clc
x = 's';
for i = l:5,
    if x = = 'q',
        break
    end
    j = l;
    while j<=5,
       ['A(' num2str(i) ',' num2str(j) ') = ' num2str(A(i,j))]
```

```
            x = input('Modifica? (s-sim, n-não, p-próxima linha, q-sair)
        =>');
            if x = = 's',
                A(i,j) = input('Entre com o novo valor de A(i,j) = = >');
                j=j+l;
                clc
            end
            if x = = 'n',
                j=j+l;
                clc
        end
        if x = = 'p',
            clc
            break
        end
        if x = = 'q',
            clc
            break
        end
    end
end
```

8 ARQUIVOS ".m"

Os comandos do MATLAB são normalmente digitados na Janela de Comando, onde uma única linha de comando é introduzida e processada imediatamente. O MATLAB é também capaz de executar seqüências de comandos armazenadas em arquivos.

Os arquivos que contêm as declarações do MATLAB são chamadas arquivos ".m", e consistem de uma seqüências de comandos normais do MATLAB, possibilitando incluir outros arquivos ".m" escritos no formato texto (ASCII).

Para editar um arquivo texto na Janela de Comando do MATLAB selecione **New M-File** para criar um novo arquivo ou **Open M-File** para editar um arquivo já existente, a partir do menu **File**. Os arquivos podem, também, ser editados fora do MATLAB utilizando qualquer editor de texto.

Existem alguns comandos e declarações especiais para serem usados nos arquivos, por exemplo

```
%Plota uma função y=ax^2 + bx + c no intervalo -5<x<5
clear
aux='s';
```

```
while aux= = ‘s’,
        clc
        a=input(‘a =’);
        b=input(‘b =’);
        c=input(‘c =’);
        x=-5:0.1:5;
        y=a*x.^2+b*x+c;
        plot(y)
        figure(1)
        pause
        clc
        close
        aux=input(‘Plotar outro ? (s/n) = => ‘,’s’);
end
```

O caracter **%** é usado para inserir um comentário no texto, o comando ***clear*** apaga todos os dados da memória, o comando ***input*** é usado quando se deseja entrar com um dado a partir da Janela de Comando, ***pause*** provoca uma pausa na execução do arquivo até que qualquer tecla seja digitada, ***clc*** limpa a Janela de Comando, ***figure(1)*** abre a Janela Gráfica número 1 e ***close*** fecha todas as Janelas Gráficas.

9 OPERAÇÕES COM O DISCO

Os comandos ***load*** e ***save*** são usados, respectivamente, para importar dados do disco (rígido ou flexível) para a área de trabalho do MATLAB e exportar dados da área de trabalho para o disco. Outras operações com o disco podem ser efetuadas, como executar programas externos, trocar o diretório de trabalho, listagem do diretório, e serão detalhadas a seguir.

9.1 Manipulação do Disco

Os comandos ***cd***, ***dir***, ***delete***, ***type*** e ***what*** do MATLAB são usados da mesma maneira que os comandos similares do sistema operacional.

cd	troca o diretório de trabalho atual
dir	lista o conteúdo do diretório atual
delete	exclui arquivo
type	mostra o conteúdo do arquivo texto
what	lista arquivos “.m”, “.mat” e “.mex”.

Para maiores detalhes sobre estes comandos utilize o ***help***.

9.2 Executando Programas Externos

O caracter ponto de exclamação, **!**, é um desvio e indica que o restante da linha será um comando a ser executado pelo sistema operacional. Este procedimento vem sendo historicamente utilizado em todos as versões do MATLAB como "***prompt***" para indicar a execução de um comando do DOS, sendo muito útil nas versões que usavam somente o DOS. No ambiente Windows, entretanto, este comando é desnecessário, mas foi mantido nas versões do MATLAB para Windows.

Para entrar com o caracter de desvio no "***prompt***" do MATLAB, deve-se coloca-lo no início do comando do DOS ou Windows que se deseja executar. Por exemplo, para carregar um aplicativo como o programa ***Notepad*** do Windows (Bloco de Notas), sem sair do MATLAB, entre com

```
>> ! Notepad
```

Uma nova janela é aberta, o ***Notepad*** é carregado, podendo ser utilizado da maneira usual.

Pode-se usar, também, qualquer comando implícito do DOS, por exemplo:

copy, ***fomat***, ***ren***, ***mkdir***, ***rmdir***, ...

9.3 Importando e Exportando Dados

Os dados contidos na Área de Trabalho do MATLAB podem ser armazenados em arquivos, no formato texto ou binário, utilizando o comando **save**. Existem diversas maneiras de utilizar este comando. Por exemplo. para armazenar as variáveis X, Y e Z pode-se fazer:

save	salva os dados no arquivos binário "matlab.mat".
save X	salva a matriz X no arquivo binário "x.mat".
save arql X Y Z	salva as matrizes X, Y e Z no arquivo binário "arq1.mat".
save arq2.sai X Y Z -ascii	salva as matrizes X., Y e Z no arquivo texto "arq2.sai" com 8 dígitos.
save arq3.sai X Y Z -ascii -double	salva as matrizes X., Y e Z no arquivo texto "arq3.sai" com 16 dígitos.

Os dados obtidos por outros programas podem ser importados pelo MATLAB, desde que estes dados sejam gravados em disco no formato apropriado. Se os

dados são armazenados no formato ASCII, e no caso de matrizes, com colunas separadas por espaços e cada linha da matriz em uma linha do texto, o comando ***load*** pode ser usado. Por exemplo suponha que um programa em linguagem C, depois de executado, monta o arquivo "teste.sai" (mostrado abaixo) que contém uma matriz.

```
1.0000    2.0000   3.0000

4.0000    5.0000   6.0000

7.0000    8.0000   9.0000
```

Executando o comando:

```
>> load teste.sai
```

o MATLAB importa a matriz, que passa a se chamar **teste**:

```
>> teste

teste =

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

O MATLAB também pode importar (através do comando ***load***) os dados que foram anteriormente exportados por ele. Por exemplo, para importar as variáveis X, Y e Z, anteriormente exportadas usando o comando ***save***, pode-se fazer:

save	load
save X	load x
save arq1 X Y Z	load arq1
save arq2.sai X Y Z -ascii	load arq2.sai
save arq3.sai X Y Z -ascii -double	load arq3.sai

Deve-se ressaltar que o comando ***save***, quando usado para exportar os dados do MATLAB em formato texto, exporta apenas um bloco contendo todas as variáveis. E quando importamos estes comandos através do comando ***load***, apenas uma variável com nome do arquivo é importada. Por exemplo

```
>> X=rand(3,3)

X =
    0.2190    0.6793    0.5194
    0.0470    0.9347    0.8310
    0.6789    0.3835    0.0346

>> Y = rand(3,3)

Y =
    0.0535    0.0077    0.4175
    0.5297    0.3835    0.6868
    0.6711    0.0668    0.5890

>> save arq2.sai X Y -ascii
>> clear
>> load arq2.sai
>> arq2


arq2 =

    0.2190    0.6793    0.5194
    0.0470    0.9347    0.8310
    0.6789    0.3835    0.0346
    0.0535    0.0077    0.4175
    0.5297    0.3835    0.6868
    0.6711    0.0668    0.5890
```

10 REFERÊNCIAS

[1] MATLAB for Windows User's Guide, The Math Works Inc., 1991.