

1 INTRODUÇÃO AO MATLAB

1.1 Introdução

Ao se projetar equipamentos eletrônicos, máquinas, processos químicos, sistemas de controle, etc., é fundamental avaliar o comportamento dinâmico do sistema. No caso de sistemas simples, muitas vezes a experiência do projetista é suficiente para tomar boas decisões durante o projeto, as quais influenciarão o desempenho e a operação do sistema. Entretanto, quando se trata de um sistema complexo, deve-se lançar mão de “ferramentas” que permitam antecipar problemas e auxiliar nas decisões.

O MATLAB é um *software* de simulação digital importante para análise e projeto de sistemas de controle. O objetivo principal desta experiência consiste em familiarizar o aluno com algumas de suas funções básicas, úteis para a análise de sistemas lineares e que serão usadas nas experiências subsequentes.

1.2 Fundamentos do MATLAB

Matrix Laboratory - MATLAB®

O MATLAB é um poderoso programa interativo voltado à solução de problemas científicos e de engenharia. Como o nome já diz, o MATLAB é um *software* específico para manipulação e processamento de matrizes. Assim, a matriz é o elemento básico de armazenamento de informações.

Além do módulo principal, o MATLAB possui módulos auxiliares (*toolboxes*) voltados para diversas áreas, tais como: Telecomunicações, Eletrônica Digital, Eletrônica Analógica, Sistemas Não Lineares, Processamento de Sinais, etc. Novas rotinas e até mesmo *toolboxes* podem ser criadas pelo usuário para resolver os seus problemas específicos.

No decorrer das explicações, são apresentados exemplos que o aluno deve executar. Para facilitar sua identificação e compreensão, estão colocados dentro de molduras (caixas) e cada comando é acrescido de um comentário, explicando sua função.

1.3 Descrição de Comandos Básicos

1.3.1 Operações Matemáticas Básicas

O MATLAB segue regras semelhantes as da matemática formal para as expressões. Por exemplo,

```
>> a=3/4
      a = 0.75
```

Caso não se queira mostrar o valor da resposta, o caractere ‘;’ deve ser incluído ao final da expressão. É permitido, também, inserir comentário, digitando antes do texto o caractere ‘%’.

```
>> b=2           % sem o caractere ‘;’ no final da sentença o resultado é apresentado.
      b = 2
>> c=3;         % com o caractere ‘;’ no final da sentença o resultado não é apresentado.
>> d=b+c        % o resultado é armazenado na variável ‘d’ e é apresentado.
      d = 5
>> b+c         % se nenhum nome é atribuído a uma variável ela é armazenada em “ans”.
      ans = 5
```

Os nomes das variáveis no MATLAB devem começar sempre com uma letra, que pode ser seguida de letras e/ou números. *Maiúsculas* e *minúsculas* são consideradas diferentes. Portanto, uma variável definida com *letra minúscula* deve ser referenciada sempre desta forma, para ser reconhecida pelo MATLAB.

As operações matemáticas no MATLAB são realizadas da esquerda para direita na seguinte ordem:

^	potenciação			
*	multiplicação	e	/	divisão
+	adição	e	-	subtração

Os parênteses podem afetar esta ordem das operações.

```
>> 1+2^3/4*2           % 1+{(2^3)/4}*2
ans = 5
>> 1+2^3/(4*2)        % 1+[(2^3)/(4*2)]
ans = 2
>> (1+2)^3/(4*2)      % [(1+2)^3]/(4*2)
ans = 3.3750
```

1.3.2 Polinômios, Vetores e Matrizes

Um polinômio pode ser formado, digitando-se seus coeficientes, conforme mostrado abaixo. Normalmente, utilizam-se letras maiúsculas para identificar polinômios, vetores e matrizes.

```
>> P=[1 6 8];          % declaração do polinômio P = s2 + 6s + 8
```

Para obter as raízes de um polinômio qualquer, procede-se da seguinte forma:

```
>> R=roots(P)
R = -4
    -2
```

A operação inversa também é possível, ou seja, a partir das raízes de um polinômio, pode-se determinar seus coeficientes. Para tanto, utiliza-se o comando *poly*, como segue.

```
>> S=poly([-2 -4])     % raízes do polinômio
S = 1 6 8
```

Para multiplicar dois ou mais polinômios, usa-se o comando *conv*, conforme abaixo:

```
>> P1=[1 4 8];        % polinômio P1 = s2 + 4s + 8
>> P2=[1 2];          % polinômio P2 = s + 2
>> PM=conv(P1,P2)     % PM = (s2 + 4s + 8)*(s + 2)
PM = 1 6 16 16       % PM = s3 + 6s2 + 16s + 16
```

Para dividir dois polinômios, é utilizado o comando *deconv*. Esta operação nem sempre é exata, podendo existir um resto; por isso, indica-se usá-la no seguinte formato:

```
>> [Q,R]=deconv(P1,P2) % equivale a operação (s2 + 4s + 8)/(s + 2)
Q = 1 2                % Q = s + 2
R = 0 0 4              % R = 4
```

A variável *Q* armazena o quociente da divisão, ao passo que o resto é armazenado na variável *R*.

O comando $[r, p, k] = \text{residue}(\text{num}, \text{den})$ encontra os resíduos (*r*), os pólos (*p*) e o valor direto (*k*) associados à razão entre dois polinômios. O uso deste comando é ilustrado a seguir na obtenção da expansão em frações parciais da razão de polinômios abaixo:

$$\frac{N(s)}{D(s)} = \frac{3s^3 + 9s^2 - 11s - 57}{s^3 + s^2 - 9s - 9}$$

Inicialmente, são definidos os polinômios do numerador (*N*) e do denominador (*D*); em seguida, é chamada a função *residue*, obtendo-se os seus resíduos, pólos e termo direto, como ilustra o próximo quadro. >> N=[3 9 -11 -57]; % polinômio N = 3s³ + 9s² - 11s - 57

```
>> D=[1 1 -9 -9];           % polinômio D = s3 + s2 - 9s - 9
>> [r,p,k]=residue(N,D)    % Encontra os resíduos da razão polinomial N(s)/D(s)
    r
    -2
    3
    5
    % os resíduos são -2, 3 e 5
    p =
    -3
    3
    -1
    % O pólo -3 corresponde à constante -2, o pólo 3 ao resíduo 3 e o -1 ao 5
    k =
    3
    % valor direto da expansão em frações parciais
```

Com isso, é facilmente obtida a expansão em frações parciais da referida razão de polinômios:

$$\frac{3s^3 + 9s^2 - 11s - 57}{s^3 + s^2 - 9s - 9} = \frac{-2}{s+3} + \frac{3}{s-3} + \frac{5}{s+1} + 3$$

O comando `[num,den]=residue(r,p,k)` converte a expressão em frações parciais de volta para a razão de polinômios num/den. No exemplo, obtém-se de volta N(s)/D(s).

Um vetor é declarado da mesma forma que um polinômio, isto é, especificando seus componentes dentro de colchetes, conforme mostrado abaixo.

```
>> A=[1 6 8]                % declaração do vetor A = [ 1 6 8]
    A = 1 6 8
```

Alternativamente, um vetor pode ser criado usando o operador ":", na forma:

```
>> B=0:1:5                  % declaração do vetor contendo elementos de 0 à 5 com intervalos de 1 em 1
    B = 0 1 2 3 4 5
```

A declaração de uma matriz é feita de modo semelhante, isto é:

```
>> C=[1 2; 4 8]            % declaração da matriz C = [ 1 2 ]
    C = 1 2
        4 8
```

As propriedades matemáticas de cada operação envolvendo polinômios, vetores ou matrizes são válidas no MATLAB.

1.3.3 Variáveis e Funções Predefinidas

O MATLAB possui variáveis predefinidas, apresentadas a seguir:

- i e $j = \sqrt{-1}$ (se usar i e j como variáveis, limpar após o uso com o comando `clear`)
- $\pi = \pi$ (3,1416)
- $\text{Inf} = \infty$
- $\text{NaN} = \text{não número}$ (ex.: 0/0)

```
>> 2*pi
    ans = 6.2832
>> d=4/Inf
    d = 0
>> z=2+2i
    z = 2.0000 + 2.0000i
```

Além disso, diversas funções predefinidas, tais como as funções trigonométricas, as funções exponencial, raiz quadrada, etc, estão implementadas no MATLAB. Alguns exemplos são fornecidos no próximo quadro.

```

>> u=sin(3*pi/4)      % o MATLAB usa como default a medida de ângulo em radianos
    u = 0.7071
>> v=sqrt(4)         % o comando sqrt (square root) executa a operação raiz quadrada
    v = 2
>> abs(z)            % valor absoluto (módulo) da variável z definida anteriormente
    ans = 2.8284
>> angle(z)          % ângulo (fase) do número variável z definida anteriormente
    ans = 0.7854
>> exp(-1)           % exponencial
    ans = 0.3679
>> log10(100)        % logaritmo na base 10
    ans = 2

```

1.4 Gráficos no MATLAB

O MATLAB é um *software* capaz de traçar vários tipos de gráficos, sejam eles em duas ou mais dimensões, em escalas lineares, logarítmicas ou semilogarítmicas. Nesta experiência, serão estudados apenas os gráficos bidimensionais com escalas lineares.

1.4.1 Criando Gráficos

Para criar um gráfico linear da forma (x,y) , onde x é uma variável e y é uma função, ou seja, $y=f(x)$, digita-se o comando **plot(x,y)**. É importante, no entanto, fornecer uma faixa de variação da variável x , o que deve ser feito conforme mostrado abaixo.

```

>> x=-15:0.05:15;    % x assumirá valores entre -15 e +15 com intervalos de 0,05
>> y=sin(x);
>> plot(x,y)

```

No caso de se desejar mais curvas em um mesmo gráfico, pode-se usar o comando **hold** (ou **hold on**) de forma a fazer com que a escala do próximo gráfico se adapte à escala do gráfico atual. Continuando o exemplo anterior:

```

>> y2=cos(x);
>> hold                % pode-se utilizar também o comando hold on
current plot held      % gráfico corrente "travado"
>> plot(x,y2);

```

Observe que todas as curvas são traçadas com linha contínua azul, pois este é o estilo de linha padrão. Note que a semelhança entre as curvas pode ser um problema para sua identificação.

Digitando **hold** novamente (ou **hold off**), o gráfico é liberado, dando lugar a um novo.

```

>> hold
current plot released  % gráfico corrente "liberado"

```

Outra forma de traçar mais de uma curva no mesmo gráfico é mostrada a seguir.

```

>> y3 = sin(2*x);
>> plot(x,y,x,y2,x,y3);

```

Observe que, deste modo, as curvas são traçadas em cores distintas, facilitando sua identificação. A ordem das cores apresentadas é a seguinte: *azul, verde, vermelho, ciano, magenta, amarelo e preto*.

1.4.2 Melhorando a Apresentação dos Gráficos

Usando o comando **plot**, pode-se identificar as curvas pelas suas cores. No entanto, se a impressora utilizada for monocromática, é importante que as curvas sejam traçadas em estilos diferentes, especificando o *tipo de linha* e, se necessário, utilizando *símbolos* para marcar cada valor calculado pelo MATLAB, de acordo com a convenção estabelecida na Tabela 1.1.

Tabela 1.1 – Especificações extras para o traçado de gráficos

Tipos de Linha	Símbolos	Cores de Linha
- contínua	o com círculos	y amarelo (yellow)
-- tracejada	. com pontos	m magenta (magenta)
: pontilhada	x com caracteres x	c ciano (cyan)
-. traço e ponto	+ com caracteres +	r vermelho (red)
	* com caracteres *	g verde (green)
	s quadrado	b azul (blue)
	d diamante	w branco (white)
	v triângulo	k preto (black)
	p estrela com cinco arestas	
	h estrela com seis arestas	

Na versão 6 do MATLAB, estas melhorias e outras mais são conseguidas de uma forma mais fácil, acessando os menus na própria janela do gráfico (Ambiente *Figure*). Por exemplo, ativando o modo **Edit Plot** no menu **Tools** e clicando duas vezes, rapidamente, sobre a curva, abre-se uma janela, que permite ajustar diversos parâmetros como os exibidos na tabela acima.

A forma mais adequada para traçar gráficos com mais de uma curva é a seguinte:

```
>> plot(x,y,'-r')           % o gráfico y será traçado com linha contínua vermelha.
>> hold on                  % retém a execução do gráfico.
>> plot(x,y2,'--b')        % o gráfico y2 será traçado com linha tracejada azul.
```

Ou então, usando apenas um comando, conforme exemplificado abaixo:

```
>> plot(x,y,'-r',x,y2,'--b') % o gráfico obtido será idêntico ao anterior.
```

Para adicionar linhas de grade no gráfico, dar nome aos eixos, além de inserir título no gráfico e mudar o valor das escalas dos eixos, são usados os seguintes comandos:

grid	insere linhas de grade no gráfico
xlabel 'Nome do eixo x';	insere o nome do eixo x
ylabel 'Nome do eixo y';	insere o nome do eixo y
title 'título do gráfico';	insere o título do gráfico
axis([xmin,xmax,ymin,ymax]);	muda as escalas dos eixos x e y
text(posição no eixo x,posição no eixo y,'Texto');	insere um texto dentro do gráfico
legend('Texto 1', 'Texto 2', n);	insere a legenda no gráfico

Dica: Estes comandos são facilmente acionados através do menu **Insert**, que faz parte do ambiente *Figure*.

O comando **text** possui o inconveniente de, uma vez colocado o texto, não poder ser movido com facilidade. Existe, porém, um comando **gtext** com o qual a escolha da localização do texto é feita com auxílio do mouse. Segue um exemplo:

```
>> grid                    % insere linhas de grade no gráfico
>> xlabel 'x'              % insere o nome do eixo x
>> ylabel 'y = f(x)'      % insere o nome do eixo y
>> title 'Exercício'      % insere o título do gráfico
>> axis([-10,10,-1.1,1.1]) % muda as escalas dos eixos x e y
>> legend('sen(x)', 'cos(x)') % insere legenda no gráfico
```

Obs.: A legenda pode ser movida com o auxílio do mouse, arrastando-a para a posição desejada.

1.4.4 Imprimindo e Copiando Gráficos

A impressão dos diagramas pode ser feita de duas maneiras: impressão direta do gráfico para a impressora, ou transportando o gráfico para um editor de texto (Word, LaTeX, etc.)

Para imprimir diretamente o gráfico, clique no menu **"File" – "Print"**. Desta forma, o gráfico será impresso usando toda a página. Caso a impressão não esteja da forma desejada, deve-se configurar a impressora no menu **"File" – "Print Setup"** e a página no menu **"File" – "Page Setup"**.

Para copiar um gráfico para um editor de textos, escolhe-se antes a opção de cópia. Esta escolha é feita no menu **"Edit" – "Copy Options"**, como mostrado na *Figura 1.1*. Nesta janela, escolhe-se inicialmente o tipo de figura: *Metafile* ou *Bitmap*.

Cada uma das opções tem as suas vantagens e desvantagens, sendo que a principal vantagem de uma figura tipo *Metafile* é que ela ocupa menos espaço que uma do tipo *Bitmap* (quase a metade). Por outro lado, uma figura do tipo *Metafile* pode ser distorcida quando, após colada no editor de textos, seu tamanho for modificado somente num dos sentidos (horizontal ou vertical). Já uma figura do tipo *Bitmap* preserva suas características mesmo quando suas dimensões são completamente alteradas dentro do editor de textos.

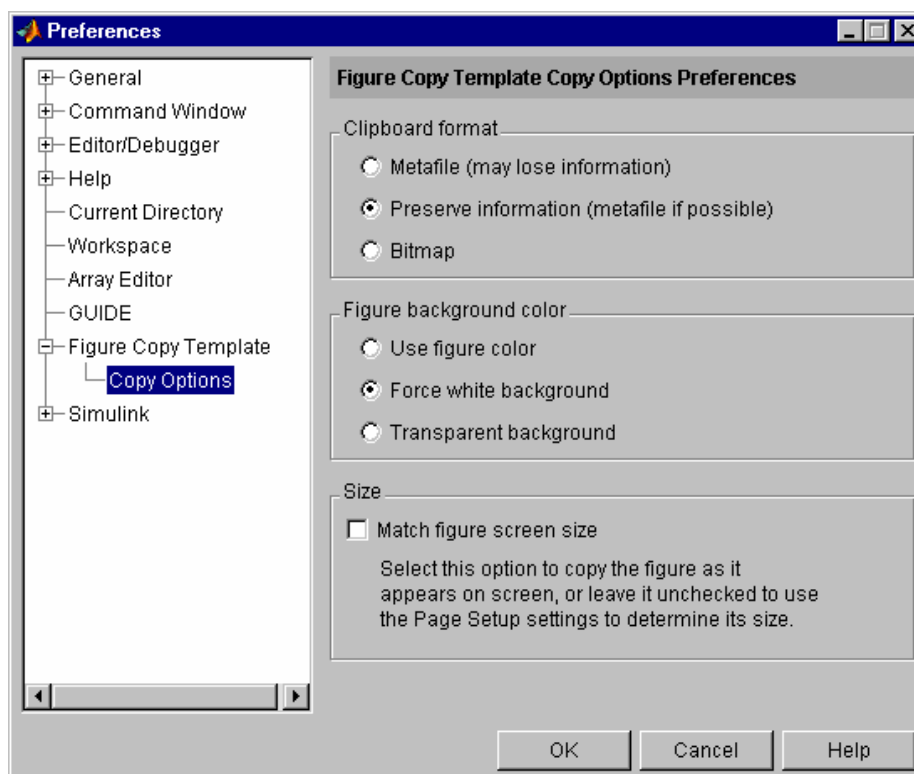


Figura 1.1: Janela de opções para copiar figuras

Pode-se escolher também a cor de fundo entre três opções: usar a cor da figura, a cor branca ou fundo transparente. Finalmente, é possível ajustar o tamanho da figura e sua posição na página.

Após selecionar as opções de cópia, escolha a opção **"Copy Figure"** no menu **"Edit"** do gráfico para copiar o gráfico no MATLAB e depois use o comando **"Editar" – "Colar especial"** no MS-Word. É importante salientar aqui que, dependendo das opções de cópia, surgirão opções diferentes no momento de colar a figura no MS-Word. Aconselha-se não usar a opção "flutuar sobre texto".

Uma vez colado no MS-Word, é possível ajustar o tamanho do gráfico clicando uma vez em cima deste e redefinindo o tamanho com o auxílio do *mouse*.

1.5 Usando o Help

O *help on line* do MATLAB deve ser usado para obter informações sobre os comandos, funções operadores e *toolboxes* que compõem o MATLAB. Existem duas formas de obter *help on line*.

Digitando-se *help*, o MATLAB apresenta uma lista de tópicos de ajuda dentro da janela de comando. Para obter informações sobre um tópico ou comando em especial digite *help + nome do tópico* que se deseja aprender.

```
>> help plot
```

Para obter ajuda sobre uma *toolbox* em especial, digite *help + nome da toolbox*. Por exemplo, para conhecer os comandos existentes na área de controle, ou seja, sobre a "toolbox control", digite:

```
>> help control
```

Helpwin apresenta os tópicos de ajuda numa janela fora da janela de comando do MATLAB. Este comando é útil quando não se quer misturar o trabalho feito na janela de comando com o texto do tópico de ajuda. Assim, para obter o mesmo texto de ajuda do comando *plot* em uma janela separada digite:

```
>> helpwin plot
```

1.6 Salvando e Recuperando Variáveis

Para salvar os valores das variáveis usadas numa sessão do MATLAB, a fim de que possam ser utilizadas em outro momento, utiliza-se o comando *Save*. Este comando salva todas as variáveis geradas pelo usuário num arquivo MATLAB.mat no diretório de trabalho do MATLAB. É conveniente, entretanto, que o nome deste arquivo e o diretório no qual ele será colocado, sejam pessoais, como exemplificado a seguir.

```
>> save c:\expl\teste % o nome do arquivo será teste.mat e ele será gravado no diretório c:\expl
```

Obs.: Este comando salva apenas os valores das variáveis definidas numa sessão, não salvando gráficos, ou qualquer outro tipo de procedimento adotado no MATLAB.

Tendo salvado as variáveis neste arquivo, o MATLAB pode ser fechado para, numa próxima sessão, reutilizá-las, recuperando-as através do comando *load*, isto é:


```
>> load c:\expl\teste % é necessário informar todo o caminho onde se encontra o arquivo
```

Ob.: Para maiores detalhes a respeito destes comandos, consulte o help do MATLAB.

Existem ainda outros comandos que permitem trabalhar com as variáveis utilizadas. Os comandos *who* e *whos* fornecem uma lista das variáveis correntes. O comando *clear* limpa as variáveis.

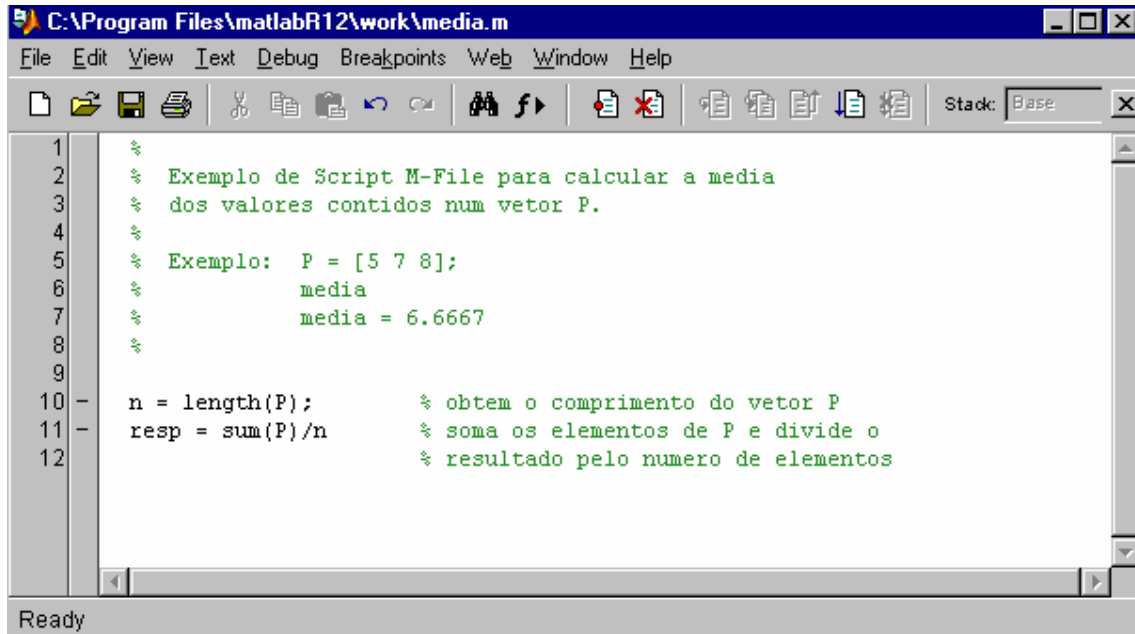
1.7 Programando no MATLAB – Uma introdução

Arquivos que contêm código de linguagem MATLAB são chamados de *M-files*. M-files podem ser *funções*, que aceitam argumentos e produzem um resultado, ou podem ser *scripts*, que executam uma série de declarações do MATLAB. O nome de um arquivo M-file deve ser seguido por “.m”.

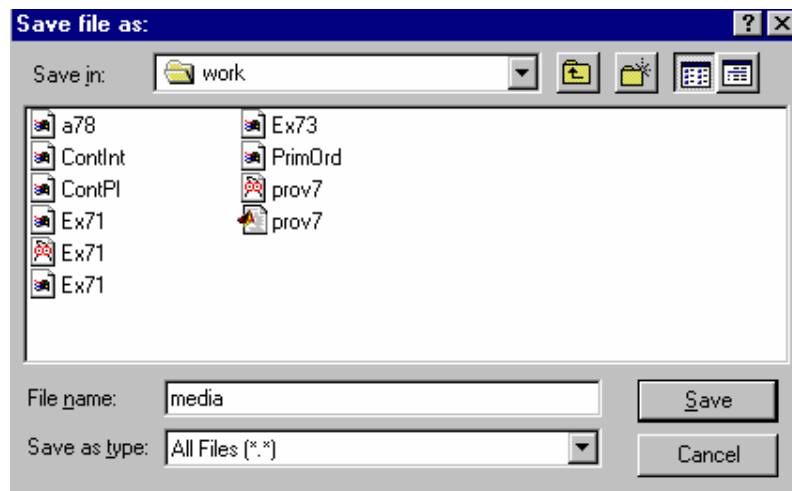
Para escrever um *M-file* deve-se selecionar "**File+New+M-File**" ou clicar no botão de atalho . Com isso, o Editor/Debugger é aberto e um programa *M-file* pode ser criado.

Script M-files	Function M-files
➤ Não aceitam argumentos de entrada.	➤ Podem aceitar argumentos de entrada.
➤ Operam sobre dados definidos no <i>workspace</i> do MATLAB.	➤ Por <i>default</i> , as variáveis internas são locais.
➤ Úteis para automatizar uma série de passos que necessitam ser repetidos várias vezes.	➤ Úteis para aumentar a linguagem do MATLAB.

Um exemplo de **Script M-file** é dado abaixo:



Para salvar o *M-File* do tipo *script* ou do tipo *function*, utilize a opção "Save" do menu "File" do MATLAB Editor/Debugger, o que abrirá a janela mostrada abaixo (Windows 98). Escolha então um diretório apropriado para salvar seu arquivo, o nome do arquivo seguido da extensão ".m".



Os comentários inseridos no arquivo servem para auxiliar os usuários que não sabem utilizar tal arquivo. Assim, pode-se utilizar o help do MATLAB conforme mostrado a seguir.

```
>> help media
```

Uma propriedade muito importante de um *Script M-File* é que suas variáveis são globais, o que implica na necessidade de declará-las no MATLAB. Note que se, no exemplo anterior, for usado um nome diferente para a variável P, por exemplo T = [1 3 5], não será possível obter o resultado, pois internamente a variável T não é utilizada e a variável P não existe.

```

>> T=[0 1 3];
>> media
    ??? Undefined function or variable P.

```

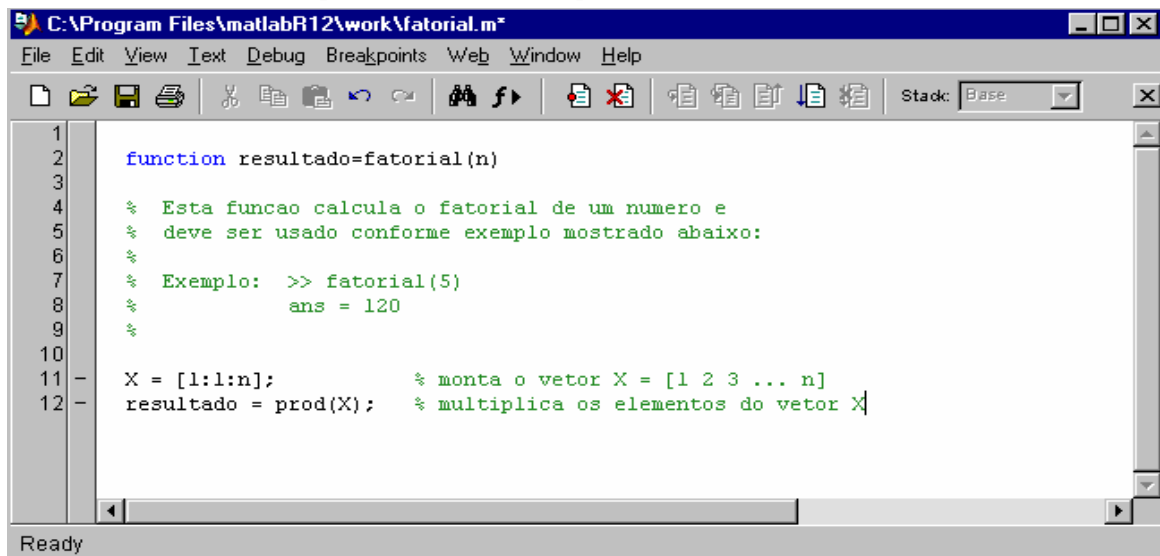
É necessário que, na declaração das variáveis, sejam utilizados os mesmos nomes internos ao arquivo. Desta forma, tem-se:


```
>> P=[0 1 3];
>> media
    resp = 1.3333
```

Utilize agora este arquivo para calcular a média entre alguns valores e testar seu funcionamento!!!

Obs.: Para executar um arquivo script *M-File*, basta digitar seu nome na janela de comandos do MATLAB.

Vamos fazer um exemplo de *function M-file* para treinar.



```
C:\Program Files\matlabR12\work\fatorial.m
File Edit View Text Debug Breakpoints Web Window Help
[Icons] Stack: Base
1
2 function resultado=fatorial(n)
3
4 % Esta funcao calcula o fatorial de um numero e
5 % deve ser usado conforme exemplo mostrado abaixo:
6
7 % Exemplo: >> fatorial(5)
8 %          ans = 120
9
10
11 X = [1:1:n];           % monta o vetor X = [1 2 3 ... n]
12 resultado = prod(X);  % multiplica os elementos do vetor X
```

Salve esta function M-file com o nome de *fatorial.m*. Digitando *fatorial(n)* na janela de comando do MATLAB, onde *n* é um número inteiro positivo, o fatorial de tal número é obtido. Assim, por exemplo, ao digitar-se *fatorial(5)*, obtém-se a seguinte resposta:

```
>> fatorial (5)
    resultado = 120
```

A função do exemplo acima tem alguns elementos que são comuns a todas as funções M-file. A primeira linha define o nome da função, o número e a ordem dos argumentos de entrada e saída (*function*[argumentos de saída] = nome da função [argumentos de entrada]). Os termos entre colchetes são opcionais. O texto ajuda é aquele que se encontra depois do caractere %.

Digite *help fatorial* na janela de comando do MATLAB e observe o resultado.

```
>> help fatorial
```

1.8 Definindo os Caminhos de Procura de Arquivos do MATLAB

É importante ressaltar que os arquivos de usuários devem ser salvos num diretório que esteja no caminho de procura do MATLAB, ou seja, no seu *MATLAB Path*. Para ver todos os diretórios em que o MATLAB pesquisa arquivos do tipo M-file (ou outros), escolha **File+Set Path** na janela de comandos do MATLAB. Com isso, a janela da *Figura 1.2* é exibida.

É possível adicionar caminhos (*paths*), clicando no botão **Add Folder ...** (*Figura 1.2*). No laboratório de ASL, por exemplo, é conveniente incluir o caminho G:\ASL\ e colocar os arquivos criados neste diretório ou num subdiretório do usuário.

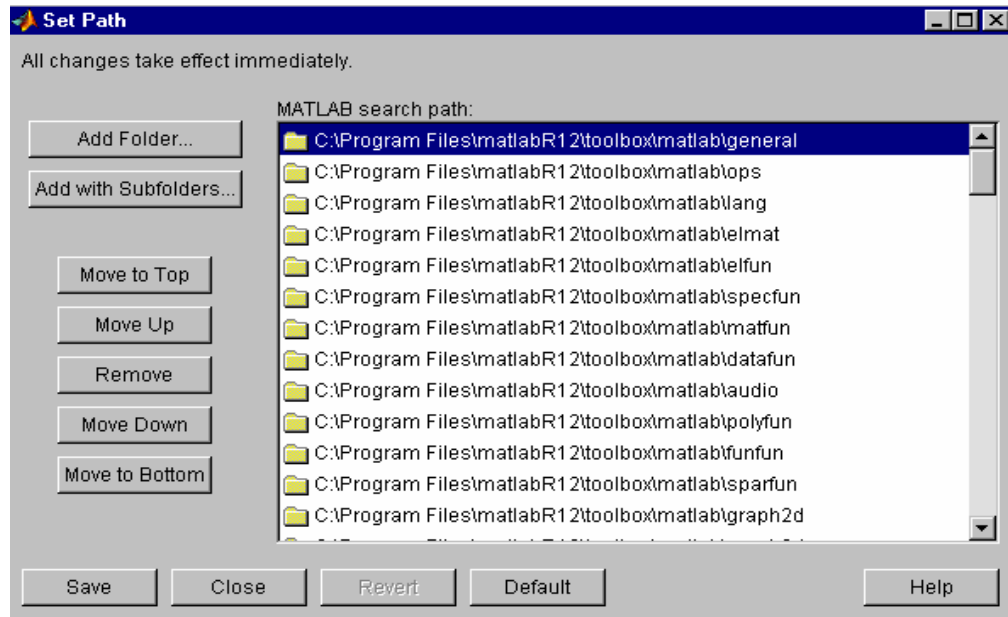


Figura 1.2: Janela do MATLAB Path.

Numa **function**, podem ser empregados várias das estruturas normalmente usadas em linguagens de programação, tais como **if - else - end**, **for - end**, **while - end**. A seguir, são listadas algumas das mais usadas para realização dos exercícios propostos nessa experiência:

Comando	Sintaxe
fprintf	fprintf('O valor medido é %d e o valor calculado é %d',x,y)
input	N = input('Digite o número de provas ')
if else elseif	if x >= 5 fprintf('Texto 1') else fprintf('Texto 2') end

1.9 Tarefa

Escreva arquivos M-files para responder os exercícios abaixo. Entregue as listagens dos M-files antes da próxima experiência.

Exercício 1 – Obtenha as raízes dos seguintes polinômios:

$$a) s^2 + 4s + 5 = 0$$

$$b) s^3 + 6s^2 + 25s + 8 = 0$$

Exercício 2 – Obtenha os polinômios que possuem as seguintes raízes:

$$a) s = -2 \pm j2$$

$$b) s_1 = -10 \quad e \quad s_2 = -1 \pm j2$$

Exercício 3 – Obtenha os polinômios resultantes das seguintes operações:

$$a) (s^2 + 5) * (s + 2) / (s + 3)$$

$$b) (s + 1) * (s + 2) * (s + 3) * (s + 4)$$

Obs.: Nas operações com divisão, mostre tanto o quociente como o resto obtido na operação.

Exercício 4 – Expanda as seguintes B(s)/A(s) em frações parciais utilizando o MATLAB:

$$\text{a) } \frac{B(s)}{A(s)} = \frac{s^2 + 2s + 3}{(s + 1)^3}$$

$$\text{b) } \frac{B(s)}{A(s)} = \frac{2s^3 + 5s^2 + 3s + 6}{s^3 + 6s^2 + 11s + 6}$$

Exercício 5 – Trace em um mesmo gráfico as funções $y_1 = x^2$ e $y_2 = x^{0.5}$. Considere uma variação de x entre 0 e 10 com intervalos de 0,01. Utilize os recursos do *software* para diferenciar as curvas em cor e tipo de linha; acrescente, também, legenda, título e o nome dos eixos.

Dica: Note que x é um vetor linha e que não é possível obter a operação matemática x^2 para esse tipo de vetor. Sendo assim, deve-se utilizar o recurso mostrado abaixo, de forma que a operação matemática seja executada sobre cada elemento escalar do vetor, e não sobre o vetor propriamente dito.

```
>> y1 = x.^2;      % observe que foi utilizado um ponto logo após a variável x
```

Exercício 6 – Obtenha, num mesmo gráfico, as curvas de resposta das funções $y_1 = \text{sen}(t)$ e $y_2 = \text{sen}(2t + \pi/4)$ e verifique o período de cada um dos sinais. Considere que t varia de 0 a 4π . Não esqueça de usar os recursos do *software* para diferenciar as curvas e colocar título e nome nos eixos, além da legenda. É necessário mostrar apenas um gráfico, o qual deve conter as duas curvas traçadas, bem como uma indicação gráfica de como foi obtido o período de uma das senóides.

Exercício 7 – Crie um arquivo *script file* (M-File) que trace três curvas no mesmo gráfico. Este arquivo deve ainda fazer com que o gráfico tenha fundo branco, grid, e nome na escala x como sendo "Tempo (s)" e na escala y como sendo "Amplitude". **Note que a faixa de variação de valores para o eixo das abscissas, bem como as funções que serão usadas para obter os gráficos devem ser fornecidas externamente no momento de uso do M-File, não podendo ser definidas internamente no arquivo.**

Exercício 8 – Utilizando o *help* do MATLAB, explique, com suas próprias palavras, como funcionam os comandos *figure* e *printsys*. Dê exemplos da utilização dos comandos.

Exercício 9 – Crie uma função que calcule a média semestral, considerando que são fornecidas as notas das três provas e a média do laboratório. Esta função deve retornar a média do aluno juntamente com o resultado "Aprovado" ou "Exame". Caso o aluno não seja aprovado, deve fornecer ainda a nota necessária no exame para obter aprovação.

Exercício 10 – Faça um programa MATLAB, que peça para introduzir os coeficientes de dois polinômios e apresente os seguintes resultados:

- Raízes de cada um dos polinômios;
- Gráfico de cada um dos polinômios;
- Divisão dos polinômios;
- Expansão em frações parciais da relação dos polinômios.

Ë O programa deve ser repetitivo, parando somente sob a ação de um comando específico.

Ë A escolha das funções deve ser feita através de um menu.

Ë Será valorizado o uso de *functions* para algumas das tarefas envolvidas.

1.10. REFERÊNCIAS BIBLIOGRÁFICAS E DA INTERNET

- [1] **D'azzo**, John J. and Houpis, Constantine H., *Análise e Projeto de Sistemas de Controle Lineares*. Guanabara Dois, 1982.
- [2] **Distefano**, Joseph J., *Sistemas de Retroação e Controle*, Coleção Schaum. McGraw-Hill do Brasil, 1972.

- [3] **Kuo**, Benjamin C., *Automatic Control Systems*. Prentice-Hall International, 1995.
- [4] **Ogata**, Katsuhiko, *Engenharia de Controle Moderno*. Prentice-Hall do Brasil, 1993.
- [5] **Ogata**, Katsuhiko, *Solving Control Problems*. Prentice-Hall International, Englewood Cliffs, NJ 1997.
- [6] **Júnior**, A. P., **Neves** H. P., *Guia do Matlab - Versão do Estudante*. Makron Books Editora, São Paulo, SP 1997.
- [7] <http://www.mat.ufmg.br/~regi/topicos/intmatl.html>
- [8] <http://www.del.ufms.br/tutoriais/matlab/apresentacao.htm>
- [9] <http://www.mec.ita.br/~adade/Matlab/Web/indice.htm>
- [10] <http://www.math.mtu.edu/~msgocken/intro/intro.html>
- [11] <http://www.mathworks.com>