Aula 01: Introdução ao MatLab

1. Introdução

Em Processamento Digital de Sinais, técnicas de amostragem de sinais, transformadas discretas, filtros digitais e, principalmente, algoritmos que realizam estas técnicas são estudados.

O MatLab é uma ferramenta muito útil neste sentido, possibilitando tanto o desenvolvimento, quanto a simulação de projetos em engenharia em geral.

O principal motivo deste sucesso é a utilização maciça de vetores e matrizes (MatLab = MATrix LABoratory) para representação de dados, que praticamente elimina a necessidade de utilização de laços FOR e WHILE simplificando e acelerando os programas.

O objetivo desta primeira aula é rever alguns conceitos de programação em MatLab.

2. Conhecendo a Área de Trabalho:

A área de trabalho do MatLab é dividida em várias partes, cujas posições variam de versão para versão. As principais são:

- ***Current Directory*** - apresenta os documentos e arquivos existentes na pasta em que se está trabalhando
- ***Workspace -*** apresenta as variáveis criadas pelo algoritmo.
- ***Command History*** - apresenta o histórico dos comandos utilizados.
- ***Command Window*** - executa comandos rápidos e apresenta os resultados e valores das variáveis.

O objetivo destas janelas é analisar o funcionamento dos algoritmos como evolução dos valores das variáveis, verificação da sintaxe dos comandos e utilização do comando HELP. Ou seja, são janelas voltadas para testes, "rascunhos" e pequenos algoritmos.

A elaboração e desenvolvimento de programas mais complexos deve ser realizada a partir da geração de arquivos de extensão *.m*.

3. Arquivos-m

Arquivos-m são arquivos de texto que contém comandos do MatLab e cujo nome apresenta a seguinte forma: $\boldsymbol{nome\_do\_arquivo.m}$.

Eles podem ser criados e modificados através de qualquer editor de texto capaz de salvar arquivos ASCII.

Porém, mais convenientemente, eles podem ser criados e editados através do editor contido no próprio MatLab, que pode ser aberto através do menu, opção *file*

ou dos dois botões à esquerda na barra de ferramentas utilizados para iniciar o editor de texto, criar ou abrir um arquivo já existente.

Outra opção para abrir um arquivo já existente é clicar duas vezes sobre seu nome no *current directory*.

Um erro muito comum, por parte dos usuários do MatLab, é tentar abrir um arquivo que não aparece nesta janela. Isso ocorre quando o MatLab está apontando para outra pasta.

A consequência deste erro é a não compilação e a não execução do arquivo. Para corrigir este problema, basta clicar nos três pontinhos do lado direito da barra de ferramenta e procurar a pasta desejada.

Para esta disciplina, é conveniente que cada aluno crie uma pasta em sua área de trabalho (não no computador) com o nome de EL9720 e salve todos os arquivos criados nesta disciplina nesta pasta.

Existem dois tipos de arquivos-m: programas e funções.

4. Programas

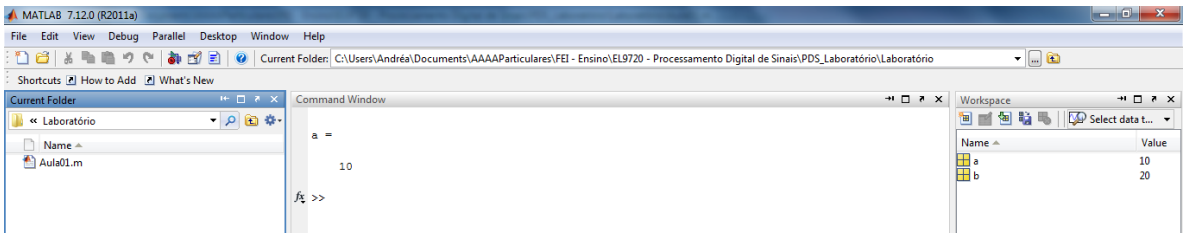
Programa é uma sequência de funções e comandos, que os executa como se tivessem sido digitados diretamente na *command window*.

4.1 Definindo variáveis

A definição de variáveis é feita de modo muito simples, através da atribuição de um valor ao nome da variável que se deseja criar, conforme é apresentado no exemplo da criação da variável $\boldsymbol{a}$ e $\boldsymbol{b}$:

```
a=10
b=20;
```

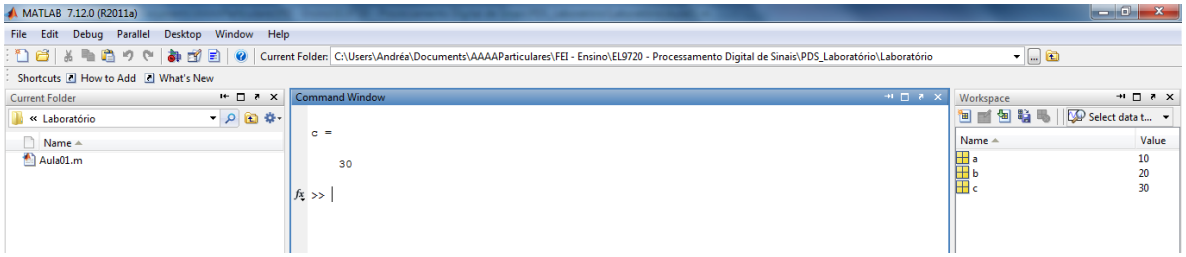
Pressionando a tecla F5, compilamos e rodamos este programa de apenas dois comandos e obtemos as seguintes mudanças na área de trabalho:



Note que as variáveis foram criadas no workspace e a variável a foi impressa também na área de trabalho. Isso ocorreu pela falta do ; no final do comando. Esta opção evita que os valores das variáveis sejam exibidos na tela.

Além disso, outras variáveis podem ser criadas a partir do resultado de novas operações conforme apresentado no exemplo:

```
a=10;
b=20;
c=a+b
```



Note que as variáveis são todas criadas no workspace, entretanto a única que aparece na janela de comando é a variável c devido a ausência do ponto-e-vírgula no final do comando.

Além das variáveis criadas e definidas pelo usuário**,** outras variáveis pré-definidas estão disponíveis pelo MatLab. Destas variáveis, as mais utilizadas são o ***pi*** (π) e as letras ***i*** e ***j*** ($\sqrt{-1}$).

Desse modo, uma variável complexa pode ser definida especificando-se a parte real e a parte imaginária da seguinte maneira:

```
d=1+2i;
e=3+3j;
```

É importante ressaltar que estas variáveis pré-definidas podem, também, serem sobrescritas e usadas para outros valores, permitindo que sejam usadas como índice de vetores, matrizes e para comandos ***for*****,** como é costume, principalmente, na linguagem C.

Por esse motivo, no MatLab, é sempre importante dar preferência para as letras m ou n para serem utilizadas como índices de laços e vetores.

No MatLab, todas as variáveis, sejam elas escalares, vetores ou matrizes são tratados da mesma maneira, como matrizes. O que significa que para o MatLab, uma variável escalar é uma matriz de ordem $[1\ x\ 1]$, um vetor linha é uma matriz de ordem $[1\ x\ n]$ e um vetor coluna é uma matriz de ordem $[n\ x\ 1]$.

Por este motivo, a definição de vetores e matrizes é bastante similar à definição de variáveis escalares, através da inserção dos valores, um a um, entre colchetes, do seguinte modo:

```
d= [2 4 6 8 10]

e=[2;3;5;7;11]

f= [1 2 3 4; 5 6 7 8; 9 10 11 12]
```

Valores separados ponto-e-vírgula definem uma mudança de linha.

Existem, entretanto, outras maneiras de definição de vetores e matrizes.

Uma delas é através da especificação de valores iniciais, finais e incrementos entre os valores, apresentados nos exemplos abaixo, onde o vetor g varia de 1 a 5 com espaçamento de 1 em 1 e o vetor h varia de 1 a 11 com espaçamento de 2 em 2.

```
g=1:5;
h=1:2:11;
```

A outra é através do comando $\boldsymbol{linspace(i,f,n)}$ que fornece um vetor linha de n elementos entre os valores i e f. O exemplo mostra os dois vetores g e h apresentados no exemplo acima, sendo criados a partir deste comando:

```
g=linspace (1,5,5);
h=linspace(1,11,6)
```

É importante ressaltar que os vetores criados destas últimas maneiras são, obrigatoriamente, vetores linha. Deste modo, se for desejado um vetor coluna, deve-se usar o operador de transposição indicado por uma apóstrofe após o nome da variável, conforme o exemplo:

```
k=g'
```

Assim como vetores podem ser criados a partir de valores escalares, as matrizes também podem ser criadas a partir de vetores. Alguns exemplos são apresentados abaixo:

```
L= [1:5; 2:6]
m= [linspace(1,5,6); linspace(10,12,6)];
```

4.2. Acessando Variáveis

Uma das grandes vantagens de se trabalhar com vetores e matrizes é que este tipo de variável pode ser acessado de maneira muito fácil.

Por exemplo, seja a um vetor e m uma matriz. Suponha que se deseje modificar o valor da terceira posição de do vetor $\boldsymbol{a}$ para 9, e o valor da linha 2, coluna 3 da matriz $\boldsymbol{m}$ para o valor 10, então:

```
a(3)=9;
m(2,3)=10;
```

ou então, suponha que se deseje somar o último valor do vetor a com o primeiro valor da última linha da coluna $\boldsymbol{m}$ e então armazenar o resultado em uma nova variável, então

```
n=a(end)+m(end,1);
```

Além disso, é possível delimitar, ou selecionar, um intervalo de linhas ou colunas, utilizando o operador dois pontos

```
o=m(1:2,2:3)
```

ou ainda, se o que se deseja é selecionar todas as linhas de uma determinada coluna, pode-se usar os dois pontos conforme o exemplo abaixo, que remove a terceira coluna de uma matriz e atribui a esta coluna a matriz nula [].

```
» m(:,3)=[]
m =
    8    1
    3    5
    4    9
```

4.3. Operações aritméticas

As operações aritméticas simples (adição, subtração, divisão e multiplicação) envolvendo números escalares seguem a interpretação natural e estão listadas abaixo:

Operação	Símbolo
adição, a+b	+
subtração, a-b	-
multiplicação, a·b	*
Divisão, a÷b	/
potenciação	^

Quando se trata de operações entre uma variável escalar e uma variável matricial (vetores ou matrizes) a operação é realizada entre a variável escalar e cada um dos elementos, por exemplo:

$$a * \begin{bmatrix} b & c \\ d & e \end{bmatrix} = \begin{bmatrix} a*b & a*c \\ a*d & a*e \end{bmatrix}$$

$$a + \begin{bmatrix} b & c \\ d & e \end{bmatrix} = \begin{bmatrix} a+b & a+c \\ a+d & a+e \end{bmatrix}$$

Já nas operações entre duas variáveis matriciais ou vetoriais, as operações de soma e subtração sempre são realizadas elemento por elemento, enquanto as operações de multiplicação podem ser realizadas tanto elemento por elemento como pela matriz.

A multiplicação de um vetor de (5X1) por um vetor (1X5) resulta numa matriz de (5X5). Já a operação de multiplicação elemento por elemento produz um terceiro vetor de mesma dimensão cujos elementos são resultado da multiplicação entre os dois vetores iniciais. Este último tipo de operação apresenta uma sintaxe diferenciada com um ponto antes do operador matemático.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \pm \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a \pm e & b \pm f \\ c \pm g & d \pm h \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a*e+b*g & a*f+b*h \\ c*e+d*h & c*f+d*h \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} .* \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a.e & b.f \\ c.g & d.h \end{bmatrix}$$

4.4. Gráficos

Outra característica muito interessante do MatLab é a facilidade de se construir qualquer tipo de gráfico de uma maneira muito simples. Os dois comandos mais utilizados são:

- $\boldsymbol{plot}(\boldsymbol{vetor.abscissa}, \boldsymbol{vetor.ordenada}, \text{'}\boldsymbol{modo}\text{'})$ e
- $\boldsymbol{stem}\ (\boldsymbol{vetor.abscissa}, \boldsymbol{vetor.ordentada}, \text{'}\boldsymbol{modo}\text{'})$**.**

O comando ***plot*** traça um gráfico contínuo com valores interpolados linearmente.

O comando stem traça um gráfico composto por uma sequência de raias com círculos na extremidade superior.

Ambos os comandos colocam o vetor relacionado como primeiro argumento do comando no eixo horizontal e o vetor do segundo argumento no eixo vertical. O 'modo' indica o formato das linhas, conforme a tabela seguinte.

Símbolo	Cor	Símbolo	Marcador	Símbolo	Tipo de Linha
b	azul	.	ponto	-	linha contínua
g	verde	o	círculo	:	linha pontilhada
r	vermelho	x	x	-.	traços e pontos
c	ciano	+	+	--	linha tracejada
m	magenta	*	estrela		
y	amarelo	s	quadrado		
k	black	d	losango		
w	branco	v	triângulo p/ baixo		
		^	triângulo p/ cima		
		<	triângulo p/ esquerda		
		>	triângulo p/ direita		
		p	pentagrama		
		h	hexagrama		

Além destes comandos para traçar o gráfico, existem outros comandos auxiliares, onde os mais importantes estão listados na abaixo:

- $grid\ on$ – coloca linhas de grade no gráfico
- $title(\text{'}\ldots\text{'})$ – acrescenta um título ao gráfico
- $xlabel(\text{'}\ldots\text{'})$ – acrescenta um título no eixo das abscissas
- $ylabel(\text{'}\ldots\text{'})$ – acrescenta um título no eixo das ordenadas
- $hold\ on$ – permite que dois gráficos sejam plotados na mesma figura sem que o anterior seja apagado.

5. Funções

As funções são arquivos-m que mediante a uma variável de entrada executam um conjunto de comandos e, então, retornam, ao usuário, outra variável. Elas podem ser criadas e geradas pelo usuário ou podem ser originais do MatLab.

Algumas funções originais são apresentadas no quadro abaixo e a listagem completas de função é fornecida no *help* do MatLab

Em todos os casos, $\boldsymbol{x}$ é a variável de entrada, e caso não seja definida pelo usuário, o MatLab apresenta a variável $\boldsymbol{ans}$ como saída.

abs(x)	Valor absoluto ou módulo de um número complexo
acos(x)	Arco co-seno
acosh(x)	Arco co-seno hiperbólico
angle(x)	Ângulo de um número complexo
asin(x)	Arco seno
asinh(x)	Arco seno hiperbólico
atan(x)	Arco tangente
atanh(x)	Arco tangente hiperbólico
cos(x)	Co-seno
cosh(x)	Co-seno hiperbólico
cross(a,b)	Produto vetorial dos vetores a e b
exp(x)	Exponencial
format	*format long* mostra 15 dígitos significativos
inv(x)	Matriz inversa da matriz x
log(x)	Logaritmo natural
log10(x)	Logaritmo na base 10
max(x)	Maior elemento em x
mean(x)	Média de x
min(x)	Menor elemento em x
sin(x)	Seno
sinh(x)	Seno hiperbólico

A função $\boldsymbol{help}$, é a função mais útil do MatLab. Sempre que surgir alguma dúvida sobre como utilizar uma determinada função, basta digitar na *command window* $\boldsymbol{help\ nome_da_fun}$ção que o MatLab irá apresentar uma breve explicação sobre a função, indicará o caminho dos documentos sobre esta função e ainda, apresentará funções relacionadas que podem ser, em alguns casos, mais úteis do que a função que se desejava saber.

Além da imensa lista de funções internas ao MatLab, em muitos casos específicos é necessário que o usuário crie suas próprias funções. Estas funções também se tratam de arquivos-m, e apresentam o seguinte formato:

$$\boldsymbol{function}\ [\boldsymbol{out\_arg1}, \boldsymbol{out\_arg2}, \ldots..] = \boldsymbol{nome\_do\_arquivo}(\boldsymbol{in\_arg1}, \boldsymbol{in\_arg\_2}\ \ldots.)$$

onde $\boldsymbol{out\_arg}$ são os argumentos de saída e $\boldsymbol{in\_arg}$ os argumentos de entrada.

O exemplo abaixo mostra uma função, que recebe dois valores como entrada e retorna a soma deles como saída.

```
function res=somateste(a,b)
% Função para somar dois numeros, a e b.
res=a+b;
```

No caso do exemplo, $\boldsymbol{res}$ é a variável de saída, e $\boldsymbol{a}$ e $\boldsymbol{b}$ são as variáveis de entrada. O símbolo % representa a adição de um comentário.

A função deve ser salva com o mesmo nome declarado no início da função.

Para chamar a função basta digitar nas linhas de comando do programa:

```
>>y=somateste (10,15)
y=
    25
```

Note que durante a execução da função, as variáveis internas à função, como o $\boldsymbol{res}$ para este exemplo, não são criadas no *workspace.*

6. Boas práticas de programação

Para que um programa possa ser executado, ele precisa que todas as variáveis necessárias estejam em sua estrutura, não podendo ser afetado por variáveis criadas previamente ou por gráficos de outros programas.

Além disso, é conveniente que a *command window* esteja limpa para facilitar a observação dos resultados e mensagens de erro.

Deste modo, ao se criar um novo programa, deve-se utilizar o comando $\boldsymbol{clc}$ para limpar a *command window*, o comando $\boldsymbol{clear\ all}$ para apagar todas as variáveis pré-existentes e o comando $\boldsymbol{close\ all}$ para encerrar todas as janelas de gráficos que estiverem abertas. Um exemplo é apresentado abaixo:

```
clc
clear all
close all
```

Além disso, é muito importante que comentários sejam sempre utilizados, uma vez que eles podem explicar o que está sendo feito durante a execução do problema e facilitar na interpretação dos resultados.

Qualquer linha de um arquivo-m que comece com o símbolo % é tratado como um comentário e por isso não é executado.

Em uma função, um comentário com um resumo desta função deve ser inserido logo abaixo da linha de definição, este comentário aparecerá sempre que o usuário utilizar o comando help.

```
function soma=Aulo02_02(a,b)
% Função para somar dois números, a e b

soma=a+b;
```

6. Atividades

IMPORTANTE:
É recomendado que cada aluno crie em seu ambiente de rede uma pasta que será utilizada apenas para os arquivos de MatLab desta disciplina.
Recomenda-se também, que todos os exercícios sejam salvos e mantidos nesta pasta durante o semestre.

I. Usando o editor de texto do MatLab, crie um script chamado $\boldsymbol{Aula01\_ex01.m}$ que execute as seguintes funções:
   a) Limpe a janela de comandos
   b) Limpe todas as variáveis pré-existentes
   c) Feche todos os gráficos e figuras
   d) Gere um vetor $\boldsymbol{a}$ com os números inteiros de 0 a 50
   e) Gere um vetor $\boldsymbol{b}$ com valores de 0 a 1 com passo de 0,1
   f) Mostre o segundo elemento do vetor $\boldsymbol{a}$
   g) Gere um vetor $\boldsymbol{c}$ de números pares de -50 a 50
   h) Gere um vetor $\boldsymbol{d}$ de 1000 pontos que vá de 0 a 1
   i) Gere um vetor $\boldsymbol{e}$ de 237 pontos que vá de 1 a 0
   j) Gere um vetor $\boldsymbol{f}$ de 5000 pontos com valores entre 0 e 2π
   k) Gere a seguinte matriz $\boldsymbol{g} = \begin{bmatrix} \mathbf{1} & \mathbf{3} & \mathbf{5} & \mathbf{7} & \mathbf{9} \\ \mathbf{2} & \mathbf{4} & \mathbf{6} & \mathbf{8} & \mathbf{10} \end{bmatrix}$

II. Digite na janela de comandos "*help zeros*" e "*help ones*" e com o auxílio da informação fornecida crie um programa chamado $\boldsymbol{Aula01\_ex02.m}$ que:
   a) Gere um vetor $\boldsymbol{a}$ de 10 números zeros em linha
   b) Gere um vetor $\boldsymbol{b}$ de 10 números 2 em coluna

III. Sendo $\boldsymbol{a} = [\,\mathbf{2\ 3\ 7}\,]$ **e** $\boldsymbol{b} = [\mathbf{0} - \mathbf{1\ 3}]$**,** crie um programa chamado $\boldsymbol{Aula01\_ex03.m}$ que execute as seguintes funções:
   a) Crie os vetores
   b) Some ($\boldsymbol{a} + \boldsymbol{b}$)
   c) Subtraia ($\boldsymbol{a} - \boldsymbol{b}$)
   d) Multiplique de $\boldsymbol{a}$ e $\boldsymbol{b}$ (elemento por elemento)
   e) Divida a por $\boldsymbol{b}$ (elemento por elemento)
   f) Eleve $\boldsymbol{a}$ ao quadrado
   g) Eleve $\boldsymbol{a}$ a $\boldsymbol{b}$ (elemento por elemento)

IV. Crie uma função que receba a variável a e forneça o gráfico da função $y = \sin(at)$, com o comando $\boldsymbol{y} = \boldsymbol{Aula01\_ex04}(\boldsymbol{a})$. Teste com:
   a) $Aula01\_ex04(20)$
   b) $Aula01\_ex04(50)$

V. Escreva um programa, chamado $\boldsymbol{Aula01\_ex05.m}$**,** que forneça um vetor contendo 100 valores aleatórios uniformemente distribuídos no intervalo de 0 a 1 e que faça o gráfico deste sinal. (Dica: use a função $\boldsymbol{rand}$**)**.

Exercícios de fixação

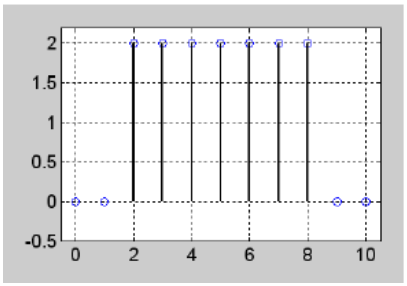
I. Sendo $x = 0{,}5 + t$ e $y = 0{,}5 + n$, onde t é um vetor de 1000 valores que vai de -1 a 0,5 e n um vetor que, também, vai de -1 a 0,5, porém com passo de 0,25, escreva uma sequencia de comandos, chamada $\boldsymbol{Aula01\_fix01.m}$, que

a) Plote, numa mesma figura, $x(t)$, em linha vermelha e contínua, e $y[n]$ em raias azuis
b) Insira título, linhas de grade, nomes nos eixos x e y e legenda. Tem dúvidas de como fazer isso? Utilize o comando help na *command window*.

II. Escreva uma sequencia de comandos, chamada $\boldsymbol{Aula01\_fix02.m}$, que gere um gráfico do sinal $x[n] = \cos\left(\frac{\pi}{8}n\right) + 0{,}2.r[n]$ onde $r[n]$ é um vetor de números aleatórios com distribuição uniforme com valores de -1 a 1. Faça 0≤n≤99. (Dica: use o comando $\boldsymbol{rand}$)

III. Escreva uma função MatLab chamada $\boldsymbol{Aula01\_fix03.m}$ cujas entradas sejam dois números inteiros **a** e **b** com **a** < **b**.
A função deverá fazer o gráfico de impulsos com amplitude 2 no intervalo **a** ≤n≤ **b.** O gráfico deve começar em a-2 e terminar em b+2. Por exemplo, ao digitar na *command window*:

>> Aula01_fix03(2,8)

Devemos obter a figura: 

Obtenha as figuras para os comandos:
a) Aula01_fix03 (1,9)
b) Aula01_fix03 (-5,15)