

Apresentação do Matlab (interfaces, help)

Exercícios: matrizes/vetores, operações aritméticas, comandos básicos, células.

Funções: “zeros”, “ones”, “length”, “clear”, “help”, “find”, “sum”, “clc”, “cell”, “celldisp”, “exit”, “quit”.

O Matlab. Como funciona?

Plataforma (padrão/arquitetura de sistema operacional) que gerencia algoritmos (funções) matemáticos/estatísticos.

Toolbox: conjunto de funções agrupadas por tema.

Apresentar: *Command window*, *command history* e *workspace*

Apresentar *Help* (a janela, não o comando)

Index: procura por assuntos

Search: procura por funções

Entrar no campo *search*, do *Help*, e digitar *arithmetic operators* no campo *search for*.

Ler o campo *description* (os operadores aritméticos básicos são os mesmos utilizados para escalares – números).

Aritmética

Calcule: $2+3$ 5×9 $(15+35) \div 5$ $8-2^3$

Álgebra

Verifique a sua *workspace*

Digite: $a = 10$ <enter>

$b = 5$ <enter>

Verifique a sua *workspace* novamente. As variáveis **a** e **b** devem estar lá agora.

Calcule: **a + b** **Dica:** Ao digitar ↑ você instrui o matlab a reproduzir sua última linha de comando. Nesse caso então, para calcular **a x b** basta digitar ↑ e trocar o sinal.

a x b (experimente as setas ↑ e ↓)

((a - b)^(a + b) + 3a - b) ÷ a - b ... isso tem que dar zero!

c = 3a + 4b

Verifique a sua *workspace* novamente. Confira se **c** está lá.

Digite: **clear c** <enter> (obs: funções serão aqui destacadas com negrito, mas no matlab isso não é necessário!)

Verifique a sua *workspace* novamente. Confira novamente se **c** está lá.

Estude a função **clear**, ou seja, digite: **help clear** <enter>

Dica: As funções **help** e **clear** são importantíssimas. Use **help** sempre que quiser saber o que uma determinada função faz e como deve ser usada, e use **clear** sempre que precisar limpar sua *workspace* (você deve fazer isso frequentemente para evitar que o matlab use valores antigos de variáveis indexadas, por você, com o mesmo nome!!!).

Digite: **clear** <enter>

Verifique a sua *workspace* novamente.

Entre agora no campo *search*, do *Help*, e digite **elementary math** no campo *search for*. Você receberá uma lista de funções matemáticas que serão úteis mais adiante.

Calcule:

- o logarítimo neperiano (natural) de 10
- o logaritimo decimal (comum) de 10
- a raiz quadrada de 121

Estude as funções correspondentes a esses cálculos

Vetores e Matrizes

Vetores e matrizes são definidos por colchetes (ou seja: [...]). “Espaço” e “vírgula” separam colunas enquanto que “ponto e vírgula” e “enter” separam linhas.

Construa o vetor linha **a** cujas colunas recebam os valores 1 2 3 4 5. Faça isso duas vezes, usando os dois tipos de separadores de colunas.

Construa o vetor coluna **aa** cujas linhas recebam os valores 6 7 8 9 10. Faça isso duas vezes, usando os dois tipos de separadores de linhas.

Digite: `b = [1:5] <enter>`

Digite: `bb = [1:0.1:5] <enter>` (Obs: O matlab usa originalmente o “ponto” (.) como separador de decimais.)

O valor intermediário em `bb` (0.1) é lido pelo matlab como o intervalo entre os valores da seqüência linear de início em 1 e término em 5. Quando nenhum valor intermediário é dado, o matlab assume como intervalo o valor “1”, assim como ocorreu em `b`.

Para transpor vetores ou matrizes, utiliza-se o símbolo “ ‘ ”.

Digite: `a <enter>`

Digite: `a' <enter>`

Compondo vetores/matrizes

Digite: `a <enter>`

Digite: `b <enter>`

Digite: `ab = [a b] <enter>`

Estude agora as funções **zeros** e **ones** (use para isso a função **help**)

Usando **ones**, crie uma matriz `c5x5`, onde $c_{ij} = 5$ para todos i e j .

Usando **zeros**, crie uma matriz `d4x3`, onde $d_{1j} = 3$ para todo j .

Dica: para indicar todas as colunas j de uma mesma linha i , ou vice versa, use o símbolo “ : ” e use parênteses para indicar linhas e colunas.

Estude a função **find**.

Usando **find**, ache, em b :

- valores maiores que 3
- valores idênticos a 3 (use “idêntico” (==) e não “igual” (=)).
- valores menores ou iguais a 3.
- valores maiores do que 5.

Digite **b** e confira.

Estude a função **sum**.

Crie uma matriz $e_{3 \times 5}$, onde $e_{1j} = j$ e $e_{2j} = 5$, para todo j , e $e_{3j} = e_{1j} + e_{2j}$, para cada j

Digite: **clear** e <enter>

Recrie a matriz e , com a mesma seqüência de comandos, mas adicione “ponto e vírgula” (;) ao final de cada linha de comando.

Ao final, digite **e** <enter>

O comando “;”, ao final de cada linha de comando, faz com que o matlab retenha a resposta ao comando dado. Isso é útil quando tais comandos geram resultados intermediários que não são relevantes para a sua análise. Acostume-se a usar sempre esse comando, pois ele permite que você receba apenas as informações de interesse e não se perca numa seqüência de valores e matrizes intermediários.

Digite: **clear** <enter>

Digite: **clc** <enter>

O comando **clc** limpa sua *command window*.

Células

Você deve ter percebido que o matlab opta por descrever o conteúdo de uma célula ao invés de mostrá-lo, quando esse conteúdo é considerado grande.

Como visualizar o conteúdo de uma célula:

1 – Indexe o conteúdo que você quer enxergar. Ex:

```
Digite:      a(2,2) <enter>
           b(2,2) <enter>
```

2 – Função **celldisp**. Estude-a.

```
Digite:      celldisp(a) <enter>
           celldisp(b) <enter>
           clear <enter>
```

O uso de células é muito útil, por exemplo, quando você quer transformar uma escalar (número) em um vetor. Esse pode ser o caso se você for fazer bootstrap nas histórias de vida das plantas/bichos da população que você estuda.

Exemplo (IMPORTANTE !). Estude bem o que foi feito. Isso será útil a você em breve.

Digite:

```
a = [2 5 0 10] <enter>
for j = 1:length(a) <enter>
    A{j} = ones(1,a(j))*j <enter>
end <enter>
b = [A{1,1:length(a)}] <enter>
```

Finalizando a sessão: digite **exit** ou **quit**