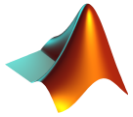


MathWorks©

MINICURSO DE MATLAB

por Renan Machado

DO INÍCIO

- IDE (*Integrated Development Environment*) com linguagem própria
- MATLAB vem de *Matrix Laboratory*
- Baseia-se em cálculos matriciais
- Escrito em C, C++ e Java
- Última release R2019a em 20 de Março de 2019

COM ELE, VOCÊ PODE...

Análise de Dados

Visão Computacional

Simulações de Comunicação *Wireless*

Processamento de Sinais

Deep Learning

Robótica

INTRODUÇÃO A LINGUAGEM

TIPOS DE DADOS FUNDAMENTAIS

- Matrizes de números reais ou complexos
- Escalares são matrizes com uma dimensão
- Definição

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

```
>> A=[1, 2, 3;4, 5, 6];
>> A=[1, 2, 3
4, 5, 6]

A =
        1        2        3
        4        5        6
```

TIPOS DE DADOS FUNDAMENTAIS

› Podemos definir matrizes através de uma técnica chamada *linespace*

```
>> a=70:-5:0

a =
       70   65   60   55   50   45   40   35   30   25   20   15   10   5   0
```

› Através de uma outra matriz

```
>> n=1:3;
>> pot=[n n.^2 n.^3]

pot =
       1   2   3   1   4   9   1   8   27
```

TIPOS DE DADOS FUNDAMENTAIS

- *Strings* são sequências ordenadas de caracteres
- Delimitadas por aspas simples (‘)
- Podemos acessar cada letra com um índice do vetor

```
>> str=‘Isto é uma string’;
>> str(1)

ans = I
>> str(3)

ans = t
```

TIPOS DE DADOS FUNDAMENTAIS

› Podemos armazenar um conjunto de *strings* através da seguinte sintaxe

```
>> semana={'Domingo'; 'Segunda'; 'Terça'; 'Quarta'; 'Quinta'; 'Sexta';
'Sábado'};
>> semana(1)

ans = 'Domingo'
```

› Da mesma forma que concatenar duas *strings*

```
>> strcat('Hoje é ', 'sexta-feira')

ans = 'Hoje é sexta-feira'
```

OPERADORES

› Aritméticos

Operador	Descrição
+	Adição
-	Subtração
*	Multiplicação
/	Divisão à direita
\	Divisão à esquerda
^	Potência
'	Transposta

› Relacionais

Operador	Descrição
<	Menor que
<=	Menor ou igual a
>	Maior que
>=	Maior ou igual a
==	Igual
~=	Diferente

› && (and) e || (or)

MANIPULAÇÃO DE ARQUIVOS

LEITURA E ESCRITA

- Forma mais simples de escrita: *disp*
- Forma mais simples de leitura: *input*

```
>> disp('Imprime uma mensagem na tela')
Imprime uma mensagem na tela

>> n1=input('Digite um número: ');
```

- Para imprimir variáveis, use *fprintf*

```
>> fprintf('O número digitado foi %f\n', n1)
O número digitado foi 42
```

LEITURA E ESCRITA DE ARQUIVOS

- Para abrir um arquivo utilize *fopen*
- Para ler os dados utilize *fscanf*

```
>> fid = fopen('arquivo.dat', 'r');
>> x = fscanf(fid, '%f');
>> fclose(fid);
```

- Para escrever um arquivo utilize *fprintf*

```
>> fid = fopen('arquivo-gravação.dat', 'w');
>> fprintf(fid, '%f', x);
```

CONDICIONAIS

IF E ELSE

› *if* avalia um conjunto de expressões com base em uma condição

› A execução do condicional é delimitado por *end*

```
n=5;
if n==5
  disp('A variável n é exatamente 5');
end
```

› Caso o condicional falhe, é possível conduzir ao *else*

```
n=5;
if mod(n,2) == 0
  disp('A variável n é par');
else
  disp('A variável n é ímpar');
end
```

ELSEIF

› É possível verificar mais de um condicional por bloco utilizando *elseif*

```
n=-1;
if n > 0
  disp('n é maior do que zero');
elseif n < 0
  disp('n é menor do que zero');
else
  disp('n é zero')
end
```

SWITCH

› Se é necessário verificar muitas condições de uma única vez, utilize *switch*

```
n = input('Digite um número: ');

switch n
  case 42
    disp('Você é um nerd!');
  otherwise
    disp('Você é uma pessoa comum.');
end
```

LAÇOS DE REPETIÇÃO

FOR

- Geralmente executado por um número determinado de vezes
- Possível interromper a execução com o comando *break*

```
a = [3 4 5 6];
for i = 1:length(a)
  if mod(a(i),2) == 0
    disp('%f é par', a(i));
  else
    dips('%f é ímpar', a(i));
  end
end
```

WHILE

- Pode executar infinitamente se a condição sempre for satisfeita
- Para parar a execução, a variável de controle deve ser alterada dentro das instruções do laço

```
n = 0

while n <= 0
  disp(n);
  n = n + 2;
end
```

WHILE

- Pode executar infinitamente se a condição sempre for satisfeita
- Para parar a execução, a variável de controle deve ser alterada dentro das instruções do laço

```
n = 0

while n <= 0
  disp(n);
  n = n + 2;
end
```

FUNÇÕES

FUNÇÕES

- Um arquivo .m de mesmo nome
- *function [saida1, saida2, ...] = nome_função(arg1, arg2, ...)*

```
function [soma, produto] = soma_produto(x, y, z)
  % Esta função pega 3 parâmetros e retorna a soma e o produto deles

  soma = x+y+z;
  produto = x*y*z;
end
```

UM DESAFIO...

O PROBLEMA

Você está encarregado(a) de cuidar do bolo para o aniversário da sua sobrinha e você decidiu que o seu bolo vai ter uma velinha para cada ano da idade total dela. Quando ele for assoprar as velinhas, ela só vai poder assoprar as velas mais altas. Sua tarefa é descobrir quantas velinhas ela apagou.

Por exemplo, se a sua sobrinha está fazendo 4 anos e o bolo vai ter 4 velinhas de alturas 4, 4, 1, 3, ela só vai poder assoprar 2 velinhas, já que as maiores velinhas têm 4 de altura e há 2 dessas velinhas.

DESCRIÇÃO DA FUNÇÃO

- Clone o repositório ***github.com/tesla-engenharia/matlab***
- Complete a função *birthdayCakeCandles*. Ela precisa retornar um inteiro representando o número de velinhas que ela pode assoprar
- birthdayCakeCandles tem os seguintes parâmetros:
  - ar: um *array* de inteiros representando a altura das velinhas

FORMATO DA ENTRADA

- A primeira linha contém um único inteiro, **n**, denotando o número de velinhas no bolo
- A segunda linha contém **n** inteiros, onde cada inteiro **i** representa a altura da vela **i**

CONSTANTES

- $1 \leq n \leq 10^5$
- $1 \leq ar(i) \leq 10^7$

INPUT

```
4                       < número de velinhas no bolo
3 2 1 3                 < altura das velinhas
```

SAÍDA

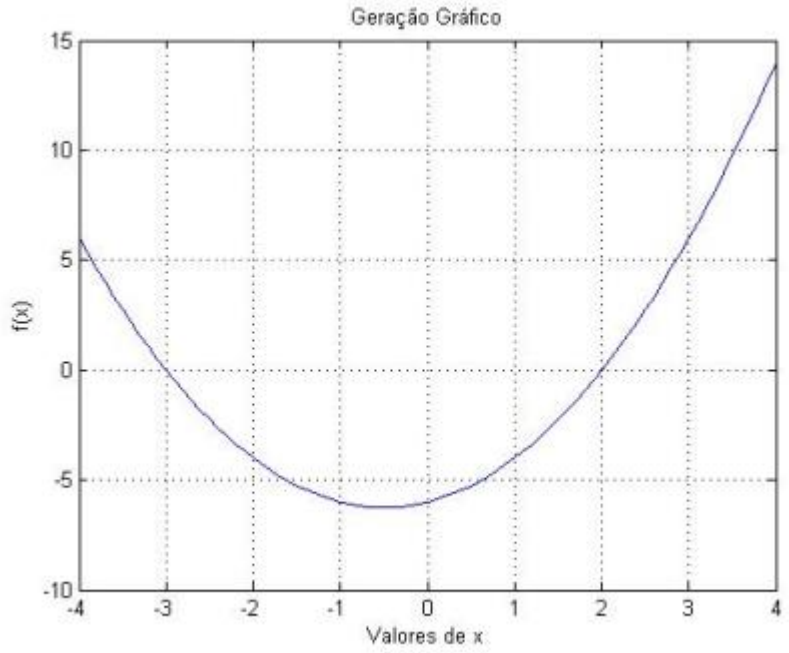
```
2                       < ela poder assoprar 2 velinhas de altura 3
```

PLOTAGEM DE GRÁFICOS

GRÁFICOS 2D

› Utilizar um conjunto de pares ordenados de pontos

```
X = -4:0.1:4;
Y = x.^2+x-6;
plot(X, Y)
xlabel('f(x)')
Ylabel('Valores de x')
title('Geração de Gráficos')
grid
```

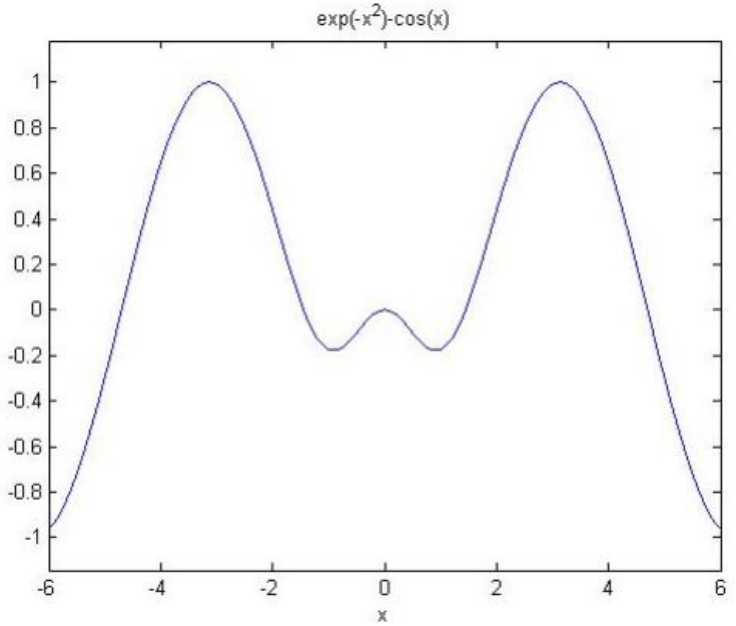


EZPLOT

› Variáveis simbólicas não possuem valores numéricos

› Com elas é possível realizar cálculos como derivação e integração

```
syms x;
ezplot('exp(-x^2)-cos(x)',
       [-6, 6]);
```



ALGO MAIS REAL...

MOTORES ELÉTRICOS

Na disciplina Motores Elétricos II, em um dos trabalhos, pede-se para que você plote o gráfico que relaciona o torque com o escorregamento de um motor de indução trifásico.

Uma contextualização básica:

- O torque é a medida da aceleração angular do rotor
- O escorregamento é uma medida do quanto o rotor está atrasado em relação ao campo magnético girante do estator

MOTORES ELÉTRICOS

- Se o torque desenvolvido é:

$$T_d = 3I_2^2 \frac{R_2}{s\omega_s}$$

- Se I_2, R_2 e ω_s forem mantidos constantes, plote o gráfico que relaciona T_d com s, sabendo que:
  - $0 \leq s \leq 1$

OBRIGADO PELA ATENÇÃO!

(63) 98461-0014
renanmav@uft.edu.br
www.linkedin.com/in/renanmav