Aula básica de OCTAVE

28/03/2019

Lucas Nunes Sequeira

OCTAVE

linguagem open source desenvolvida para computação matemática, possui uma boa interface gráfica.

Sintaxe Básica

1. Variáveis, expressões e declarações
2. Execução condicional e loops
3. Funções
4. Operações matriciais
5. Gráficos

Variáveis, expressões e declarações

linguagem não tipada

```
>> x = 5;

>> y = “Bananas Assassinas”;

>> z = [1, 1, 2, 3, 5, 8];
```

Variáveis, expressões e declarações

dúvidas sobre funções built-in

```
>> help print

'print' is a function from the file /usr/share/octave/4.2.2/m/plot/util/print.m

-- print ()

-- print (OPTIONS)

-- print (FILENAME, OPTIONS...
```

Variáveis, expressões e declarações

operadores matemáticos intuitivos

```
>> x = 5;                % comentário

>> y = x**2.25           % sem ';' (ponto e vírgula) printa-se o valor da variável

y = 37.384

>> 27.54 + 28 * 36 / 4.16

ans = 269
```

Variáveis, expressões e declarações

leitura de variável pelo terminal [input()]

```
. . .

numero_de_ararinhas = input("Quantas ararinhas tem no céu? ");

. . .
```

Variáveis, expressões e declarações

printando um valor no terminal [printf()]

```
. . .
printf("No céu, Lipinho contou %d ararinhas.", numero_de_ararinhas)
. . .

>> line = "Adorei seu chapéu!\n";
>> printf(line)                    % equivale a simplesmente escrever: >> line
Adorei seu chapéu!
```

Execução condicional e loops

operadores lógicos e função condicional [if, elseif e else]

```
. . .
a = 25;
if (a == 10 || a > 15)
        a = 14;                 % indentação não necessária mas recomendável
elseif (a != 14 && a <= -5)
        a = 40;
else
        a = 0;
endif                           % aqui declaramos final do condicional if
. . .
```

Execução condicional e loops

funções de loop [while e for]

```
. . .
for i = 1:n                    % lê-se de i = 1 até i = n
        A(i) = i**i;           % IMPORTANTE: os arrays são indexados a partir do 1
endfor
. . .
```

Funções

praticidade e estética

```
...
while ehPrimo(n)                    % aqui chamamos a função ehPrimo()
        n = n + 51;
endwhile
...
```

Funções

arquivo separado com extensão .m e mesmo nome da função

```
function resp = ehPrimo(n)                  % 'resp' é a variável que a função retornará
        resp = true;
        if n < 2
                resp = false;
        else
                for i = 2 : sqrt(n)
                        if mod(n, i) == 0   % em C seria if (n%i == 0)
                                resp = false;
                        endif
                endfor
        endif
endfunction
```

ehPrimo.m

Operações matriciais

definição $R^{n \times m}$:= matriz de n linhas por m colunas

```
...
A_array = zeros(1, 5);          % ∈ R^(1x5) == R^5 vetor linha
A_matrix = zeros(5, 7);         % ∈ R^(5x7)

A_result = A_array * A_matrix;  % R^(1x5) * R^(5x7) ∈ R^(1x7)
...
```

Operações matriciais

declarando uma matriz com valores definidos

```
>> A = [1, 1, 2; 3, 4, 5; 7, 8, -5; 2, 1, 0]        % ∈ R^(4x3)
A =
     1     1     2
     3     4     5
     7     8    -5
     2     1     0

>> A(3,3)                                           % acessando a posição A_(3,3)
ans = -5
```

Operações matriciais

além dos operadores normais (+, *, -), podemos aplicar uma função à matriz e operador ‘elemento a elemento’ (.*, ./)

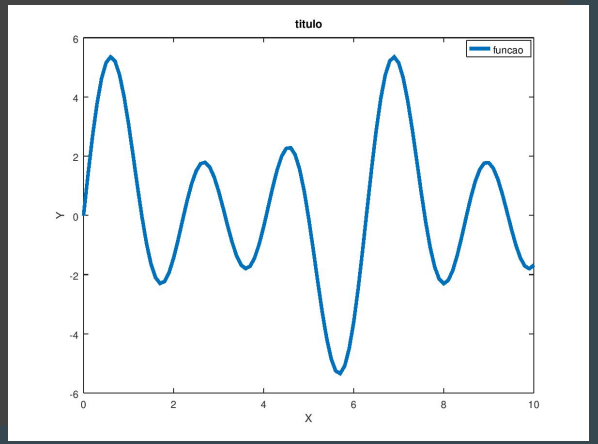
```
>> A = sin( [0, pi/6, pi/4, pi/2] )          % ∈ R^4, vírgula e espaço para colunas, ponto e vírgula, linhas
A =
        0.00000   0.50000   0.70711   1.00000
>> B = [1, 2, 3, 4];                         % ∈ R^4
>> C = A .* B                                % ∈ R^4
C =
        0.00000   1.00000   2.12132   4.00000
>> C(4)                                      % acessando a posição C_4
ans = 4.00000
```

Gráficos

flexível e de fácil uso (título, legenda, nome e tamanho dos eixos, estilo etc...)

```
>> X = [0 : 0.1 : 10];              % vetor de 0 a 10 com espaçamento 0.1
>> Y = sin(X) + 2*sin(2*X) + 3*sin(3*X);

>> hold on                          % para manter aberto um gráfico
>> plot(X, Y, "linewidth", 3)       % plotando (x, y) com expessura 3
>> legend("funcao");
>> title("título");
>> xlabel("X");
>> ylabel("Y");
```



Problema

SIR - modelo compartimental de transmissão direta, S(t) suscetíveis, I(t) infectados e R(t) recuperados.

Problema

SIR - modelo compartimental de transmissão direta, S(t) suscetíveis, I(t) infectados e R(t) recuperados.

avalie :

$$\begin{cases} dS/dt = (-\beta.S.I)/N + \varepsilon.R \\ dI/dt = (\beta.S.I)/N - \gamma.I \\ dR/dt = \gamma.I + \varepsilon.R \end{cases} \quad , \text{com } S + I + R = N$$