Introdução à Ciência da Computação

Unidade III

Programação de Computadores com MATLAB/Octave

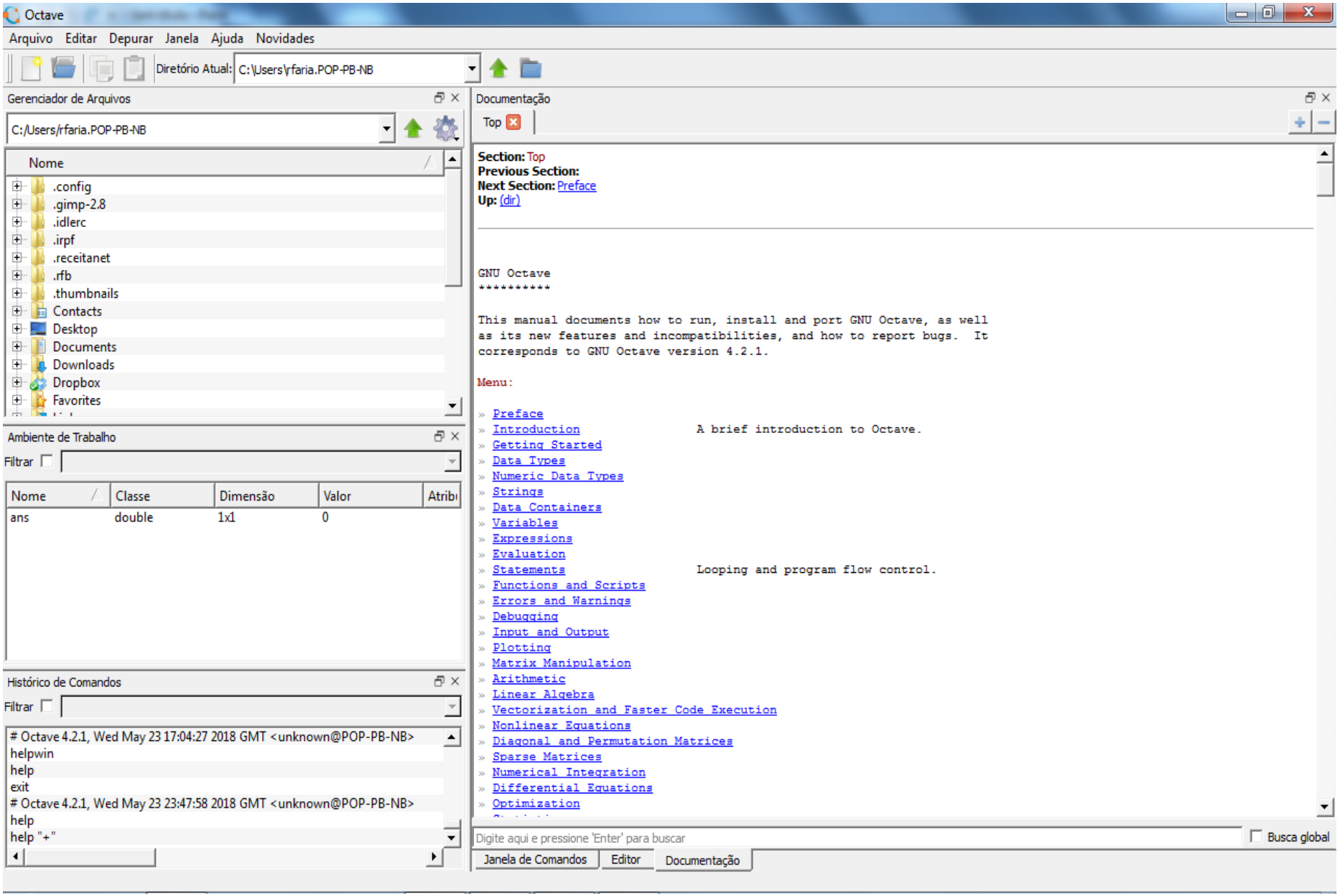
Uso do Octave no Modo Interativo

Prof. Roberto M. de Faria/DSC/UFCG

Ajuda para o Octave

- Para ter acesso à documentação ***on line*** do Octave, clique na aba “Documentação”, abaixo da Janela de Comandos, para ter acesso à Janela de Documentação
- Digite na Área de Comandos o comando **`help`**, seguido de um nome de comando, nome de função ou de um “operador” (o operador deve estar entre aspas), seguido da tecla **`<Enter>`**, para obter ajuda sobre um item específico destes

Janela da Documentação



Comandos Básicos do Octave

- Operadores Aritméticos (por precedência)
  - “^” potenciação ou exponenciação
  - “*” e “/” multiplicação e divisão
  - “+” e “–” soma e subtração
- Numa expressão aritmética, usa-se parênteses para alterar a precedência dos operadores
- No Octave, usa-se ponto decimal “.” para separar as casas decimais de números reais

Comandos Básicos do Octave

- Comandos para cálculos básicos
  - Encerra-se um comando com a tecla `<Enter>`
  - Ex:
    - **`>> 2 + 5`**
    - **`>> 7.8 + 3.5`**
    - **`>> 6 * 5`**
    - **`>> 2.5 / 5`**
    - **`>> 2 ^ 7`**
    - **`>> (2 + 5) * 7`**
  - O resultado da execução de um comando, não atribuído a uma variável, fica armazenado na variável "**`ans`**" (answer – resposta) do Octave

Reedição dos Comandos já Submetidos

- Utilizando-se as teclas “**`seta-para-cima`**” e “**`seta-para-baixo`**”, pode-se acessar comandos já submetidos à execução anterior
- Um comando já executado pode ser reeditado com modificações e re-submetido à execução ou simplesmente ser reexecutado sem modificações
- Clicar duas vezes sobre um comando na “Área de Histórico”, faz com que este comando seja reexecutado

Exercícios

- Calcule a energia cinética de uma partícula que tem a massa de 0,0023 kg e a velocidade de 3235 m/s
- Calcule o IMC de uma pessoa de peso 84,5 kg e altura de 1,72 m
- Calcule a área de um círculo e sua circunferência, sabendo que seu raio é 60 cm
- Calcule o valor final, após um mês, de um investimento de R$ 1.000,00 numa caderneta de poupança, sabendo que a inflação do período foi de 2,37% e o juro mensal é de 0,5%

Relembrando Variáveis

- Uma variável é simplesmente uma porção (parte) da memória do computador usada para armazenar algum dado.
- Uma variável possui um nome (um único caractere ou um conjunto de caracteres) que é por meio dele que se referencia aquela parte da memória. Um nome de variável pode conter letras, números e/ou um sublinhado, mas deve começar com uma letra. Além disso, os nomes de variáveis diferenciam maiúsculas de minúsculas. Ex.: a variável "X" é diferente da variável "x".

Relembrando Variáveis

- Em geral, uma variável pode armazenar:
  - um número (inteiro, real, complexo,...)
  - uma cadeia de caracteres (texto)
  - uma matriz de números, sequências de caracteres e/ou outras matrizes
  - um apontador para uma função anônima.

Ainda sobre Variáveis

- Armazenamento numa Variável
  - Quando atribui-se um valor a uma variável, ela passa a existir e é mostrada na Área de Variáveis
    - Ex:
    - **>> a = 2**
    - **>> b = 9**
    - **>> c = 7.5**
    - **>> delta = b ^ 2 - 4 * a * c**
    - **>> X1 = (-b + sqrt(delta)) / (2 * a)**
    - **>> X2 = (-b - sqrt(delta)) / (2 * a)**
  - Obs: "**sqrt()**" é uma função pré-definida do Octave

Ainda sobre Variáveis

- Uma variável pode armazenar também um texto
  - Ex: **`>> titulo = 'Raízes de uma Equação do Segundo Grau'`**
- O nome de uma variável, como comando, faz com que seu valor seja apresentado
  - Ex: **`>> delta`**
- O comando "**`who`**" mostra as variáveis já definidas
  - Ex: **`>> who`**
- O comando "**`clear`**" seguido do nome da variável remove a variável da memória
  - Ex: **`>> clear delta`**
- O comando "**`clear all`**" remove todas as variáveis definidas da memória
  - Ex: **`>> clear all`**

Variáveis do Octave

- O Octave possui algumas variáveis com valores pré-definidos:
  - **`nan`** → not-a-number
  - **`inf`** → infinito
  - **`pi`** → 3.1416
  - **`e`** → 2.7183 – base do logaritmo natural (constante de Euler)
  - **`i`** → `0.0000 + 1.0000i` → `raiz quadrada de -1`
  - **`j`** → `0.0000 + 1.0000i` → `raiz quadrada de -1`
  - **`eps`** → 2.2204e-016 – epsilon – constante da máquina de dupla precisão em ponto flutuante
  - **`true`** → valor lógico verdadeiro
  - **`false`** → valor lógico falso

Tipos de Variáveis

- Toda a variável tem um tipo a ela associado
- Os tipos para as variáveis do Octave são:
  - **`'cell'`** → arrays de células
  - **`'struct'`** → arrays de estruturas
  - **`'logical'`** → arrays lógicos
  - **`'int8'`** → inteiros com 1 byte
  - **`'int16'`** → inteiros com 2 bytes
  - **`'int32'`** → inteiros com 4 bytes
  - **`'int64'`** → inteiros com 8 bytes
  - **`'float'`** ou **`'single'`** → real de ponto flutuante com 4 bytes (4 decimais)
  - **`'double'`** → real de ponto flutuante com 8 bytes (14 decimais)

Mais sobre Tipos de Variáveis

- Outros tipos para as variáveis do Octave são:
  - **`'uint8'`** → inteiros sem sinal com 1 byte
  - **`'uint16'`** → inteiros sem sinal com 2 bytes
  - **`'uint32'`** → inteiros sem sinal com 4 bytes
  - **`'uint64'`** → inteiros sem sinal com 8 bytes
  - **`'char'`** → para arrays de string
- Criação de variável usando tipo:
  - **`>> x = int32(123)`**
- Mudança de tipo com a função "**`cast`**"
  - **`>> y = cast(x, 'int64')`**

Múltiplos Comandos

- Octave permite que se coloque múltiplos comandos numa mesma linha de comando separados por “;”
  - Ex: **`>> x = 7; y = x * 5; z = x + y`**
- O “**`;`**” no final de um comando, faz com que o Octave não mostre a resposta do cálculo
  - Ex:
    - **`>> raio = 30;`**
    - **`>> pi * raio ^ 2`**

Comentários e Limpeza da Área de Comandos

- Uma linha de comando que começa com “`%`” é um comentário
- Um comentário é colocado para acrescentar alguma informação sobre os comandos que estão sendo executados
- Um comentário não é executado, ou seja não interfere na execução de outros comandos nem no armazenamento de dados
- O comando “**`clc`**” executa a limpeza da “Área de Comandos”

Funções Pré-definidas

- Uma função é um conjunto de comandos que possui um nome e que, quando executados, produzem um resultado
- O Octave possui muitas funções pré-definidas
  - Exemplos de algumas funções:
    - **`abs()`** → valor absoluto
    - **`max()`** → máximo
    - **`min()`** → mínimo
    - **`sum()`** → soma
    - **`imag()`** → parte imaginária de um número complexo
    - **`round()`** → arredonda para o inteiro mais próximo

Mais Funções Pré-definidas

- Exemplos de algumas funções trigonométricas (usa radianos):
  - **`sin()`** → seno
  - **`cos()`** → cosseno
  - **`tan()`** → tangente
  - **`sec()`** → secante
  - **`csc()`** → cossecante
  - **`cot()`** → cotangente

Mais Funções Pré-definidas

- Exemplos de mais algumas funções trigonométricas (retorna radianos):
  - **`asin()`** → arco seno
  - **`acos()`** → arco cosseno
  - **`atan()`** → arco tangente
  - **`asec()`** → arco secante
  - **`acsc()`** → arco cossecante
  - **`acot()`** → arco cotangente

Mais Funções Pré-definidas

- Exemplos de mais algumas funções trigonométricas (usa graus):
  - **`sind()`** → seno
  - **`cosd()`** → cosseno
  - **`tand()`** → tangente
  - **`secd()`** → secante
  - **`cscd()`** → cossecante
  - **`cotd()`** → cotangente
- Existem outras funções tais como: logaritmo, exponencial, raiz quadrada, etc., que podem ser pesquisadas na documentação do Octave

Formatos

- O Octave pode trabalhar com formatos de precisão (número de dígitos após o ponto decimal) diferentes
- A precisão pode ser alterada com os comandos "`format long`" e "`format short`"
  - Ex:

```
>> format short
>> pi

ans =
    3.1416

>> format long
>> pi

ans =
   3.14159265358979
```

Formato de Notação Científica

- O Octave pode utilizar o formato de notação científica com diferentes precisões
  - Ex:

```
>> format short e
>> pi
ans =
 3.1416e+000

>> format long e
>> pi
ans =
 3.14159265358979e+000
```

- O Octave pode retornar à formatação padrão digitando-se:

```
>> format short
```

Vetores e Matrizes

- Um vetor ou uma matriz é uma coleção de valores (elementos) de mesmo tipo que possui um nome e um conjunto de índices associados
- Os elementos de um vetor ou matriz são acessíveis por meio de um conjunto de índices
- Criação de um vetor:

```
>> vet = [-5, 20, -33, 15, 0]
```

- Criação de uma matriz:

```
>> mat = [-5, 7, -12, 0; 20, 100, -3,
0; 88, -33, 15, 0]
```

Mais sobre Vetores e Matrizes

- O acesso a um elemento de um vetor é realizado por meio do nome do vetor ou matriz seguido do conjunto de índices do elemento entre parênteses
- Acesso a um elemento de um vetor:

```
>> vet(2) * 10
```

- Acesso a um elemento de uma matriz:

```
>> mat(2, 3) = 10
```

Mais sobre Vetores e Matrizes

- Criando vetores sequenciais:
  - Para criar vetores sequenciais, utiliza-se o operador “:”
    - Informando o valor inicial e o final:

```
>> vetor = 1 : 10
```

    - Informando o valor inicial, o incremento e o final:

```
>> sequencia = 1 : 3 : 10
```

    - Informando o valor inicial, o incremento e o final:

```
>> radianos = 0 : pi/4 : 2 * pi
```

Mais sobre Vetores e Matrizes

- Criando vetores randômicos:
  - Para criar vetores e matrizes randômicas, utiliza-se a função "`rand()`"
    - Matriz com uma linha e dez colunas (vetor):

      ```
      >> vetor = rand(1, 10)
      ```

    - Matriz 10x10:

      ```
      >> matriz_1 = rand(10, 10)
      ```

    - Matriz 10x10:

      ```
      >> matriz_2 = rand(10)
      ```

    - Matriz de 3 dimensões (2x3x2):

      ```
      >> matriz_3 = rand(2, 3, 2)
      ```

Algumas Operações com Matrizes

- Criação de matrizes:

```
>> A = [10, 22, -3, 14; -1, 12, 37,
4; 9, -8, 61, 4]
```

```
>> B = rand(3, 4)
```

```
>> C = [12, 27; -22, 14; -1, 0; 7, 4]
```

- Soma de matrizes:

```
>> D = A + B
```

- Subtração de matrizes:

```
>> E = A - B
```

Algumas Operações com Matrizes

- Multiplicação de matrizes:

```
>> F = A * C
```

- Multiplicação de matriz por escalar:

```
>> G = A * 2
```

- Cálculo do determinante de uma matriz:

```
>> D = [22, -3, 14; -1, 37, 4; 9, -8, 61]
```

```
>> determinante = det(D)
```

Algumas Operações com Matrizes

- Maior elemento de uma matriz:

```
>> maior = max(max(A))
```

- Maior elemento de uma matriz:

```
>> menor = min(min(A))
```

- Soma dos elementos de uma matriz:

```
>> soma = sum(sum(A))
```

- Soma acumulativa elementos das colunas de uma matriz:

```
>> somaAc = cumsum(D)
```

- Diferença entre elementos das colunas de uma matriz:

```
>> diff([2, 4, 8; 15, 31, 22])
```

Mais Algumas Operações com Vetores

- Remover um elementos de um vetor:
- `>> vetor = [22, -3, 14, -1, 37, 4, 9, -8, 61]`

  `>> vetor(4) = []`
- Remover vários elementos de um vetor:

  `>> vetor(2:4) = []`
- Acrescentar um elemento a um vetor:

  `>> vetor(5) = 10`
- Acrescentar vários elementos a um vetor:

  `>> vetor(6:10) = 1:5`

Mais Algumas Operações com Matrizes

- Remover uma linha de uma matriz:

```
>> matriz = [6, 9, 0; 7, -4, 10; 22, -3, 12]

>> matriz(1, :) = []
```

- Acrescentar várias colunas a uma matriz:

```
>> matriz(3:5, 1:3) = 0
```

- Remover várias colunas de uma matriz:

```
>> matriz(:, 2:3) = []
```

- Acrescentar várias linhas a uma matriz:

```
>> matriz(6:10, 1) = -1
```