Introduction to Octave (Part I)
IT WS I - Lecture 14

Saurabh Barjatiya

International Institute Of Information Technology, Hyderabad

02 November, 2009

Contents

Introduction to octave

- Octave can be used to solve mathematical problems.
- Octave syntax is very similar to syntax of matlab so that matlab users / programs can be migrated to octave without much difficulty.
- Octave is free and open-source
- Octave can be used to solve linear equations. It supports operations over matrices and complex numbers.
- Octave can also be used for drawing plots
- Advanced sound / image processing is often done using octave due to its extensive mathematical capabilities
- Octave is proper language with loops, variables, conditional statements, etc.

Matrices - 1

To create/declare a matrix in octave, we have to use '[', ']', ',' and ';' operators.

Creating matrix - 1

```
 > x = [1,2;3,4]
x =
   1   2
   3   4
```

Matrices - 2

The column elements can also be separated by space instead of comman

Creating matrix - 2

```
 > x = [1 2;3 4]
x =
   1   2
   3   4
```

But this is very bad practice. It is recommended to always separate column values by ',' to avoid any confusion.

Matrices - 3

To access some element of matrix we can use '(' and ')' operators

Accessing element of matrix - 1

```
> x = [1 2;3 4];
> x(1,1)
ans =  1
> x(2,2)
ans =  4
> x(1,2)
ans =  2
```

Matrices - 4

We can access more then one element of matrix at same time

Accessing elements of matrix - 2

```
> x = [1 2;3 4];
> x(1,[1,2])
ans =
   1   2
> x([1,2],[1,2])
ans =
   1   2
   3   4
```

Matrices - 5

Few important points:

- Octave variable names cannot start with number and are case-sensitive.
- If you end a line with ';' it means you do not want to print the final value. If any expression is not terminated with semi-colon then its value is printed.
- We do not explicity define type of variable in octave. Variables type depends upon value stored in the object. Same variable may belong to different types based on values stored in it.
- Dimensions are specified in row by column syntax.

Range notation - 1

Octave supports range notation for specifing linear sequences of numbers

Syntax of range notation

```
<start>:[<increment>:]<end>
```

Here, increment is optinal. If we do not specify increment it is taken to be '1'.

Range notation - 2

Example of range notation

```
> 1:2:7
ans =
   1   3   5   7
> 1:7
ans =
   1   2   3   4   5   6   7
```

Contents

Basic operations on matrices - 1

Matrices of equal dimensions can be added using '+' operator

Adding matrices

```
> x=[1,2;3,4];
> y=[4,3;2,1];
> x+y
ans =
   5   5
   5   5
```

Basic operations on matrices - 2

Matrices of equal dimensions can be subtracted from each other

Subtracting matrices

```
> x=[1,2;3,4];
> y=[4,3;2,1];
> x-y
ans =
  -3  -1
   1   3
```

Basic operations on matrices - 3

Matrices can be multiplies when number of columns of first matrix is same as number of rows of second matrix

Matrix multiplication

```
> x=[1,2];
> y=[3;4];
> x*y
ans =  11
```

Basic operations on matrices - 4

Matrix divison

Matrix divison

```
> x=[1,2;3,4];
> y=[4,3;2,1];
> x/y
ans =
    1.5000   -2.5000
    2.5000   -3.5000
```

Basic operations on matrices - 5

Scalar addition

Scalar addition

```
> x=[1,2;3,4];
> x .+ 4
ans =
   5   6
   7   8
```

Basic operations on matrices - 6

Scalar subtraction

Scalar subtraction

```
> x=[1,2;3,4];
> x .- 4
ans =
  -3  -2
  -1   0
```

Basic operations on matrices - 7

Scalar multiplication

Scalar multiplication

```
> x=[1,2;3,4];
> x .* 4
ans =
    4    8
   12   16
```

Basic operations on matrices - 8

Scalar divison

Scalar divison

```
> x=[1,2;3,4];
> x ./ 4
ans =
   0.25000   0.50000
   0.75000   1.00000
```

Contents

Creating strings

Row of characters in matrix is treated as string. Or string is stored as 1 dimensional matrix of characters.

Declaring strings

```
> x="abc"
x = abc
> y=['a','b','c']
y = abc
> strcmp(x,y)
ans =  1
```

ASCII values

- Characters in strings are stored as numbers inside computer memory.
- Before storing characters get converted to appropriate ASCII decimal value
- Before displaying the decimal value is converted to corresponding character
- ASCII value of char '0' is 48
- ASCII value of char 'a' is 97
- ASCII value of char 'A' is 65

Basic operations on strings - 1

Extracting portions of string

```
> x="abcdefgh";
> x(3)
ans = c
> x([1,5,6])
ans = aef
> x([1:2:7])
ans = aceg
```

Basic operations on strings - 2

Changing portions of strings

```
> x="abcdefgh";
> x(3)='z'
x = abzdefgh
> x([5:6])='pq'
x = abzdpqgh
```

Basic operations on strings - 3

Adding numeric values to strings

```
> x="abcdefgh";
> x+1
ans =
    98    99   100   101   102   103   104   105
> char(x+1)
ans = bcdefghi
> uint8(x)
ans =
   97   98   99  100  101  102  103  104
```