Computer Algebra and Technical Computing (MTH1006)

B. Vorselaars
bvorselaars@lincoln.ac.uk

School of Mathematics and Physics, University of Lincoln

Today

- Recap
- Plotting
- Vector indexing and vector operations
- Characters and strings
- Logical operators

Matlab recap

- Continue using logbook. Note: better to copy-paste instead of screen dump for commands (only screen dump for figures and images). Then the font size can be adjusted, etc. demo
- Vectors demo

```
>> x=0:0.01:10
>> y=x.^2+1
```

- Scripts demo
- Complex numbers

```
>> 3.6056*exp(0.9828*i)
ans =
   2.0000 + 3.0001i
```

Plotting

One of the great strengths of Matlab is the ease with which you can plot results of calculations. We consider just the plot command. The most basic format of this command is

```
>> plot(x,y)
```

where x and y are vectors. We can also omit the vector x, i.e.

```
>> plot(y)
```

In this case the values of x are taken to be $1, \ldots, N$ where N is the length of vector y. Example:

```
>> x=0:0.01:10; plot(x, sin(x))
```

will plot the sine of x for $0 \leq x \leq 10$

We now demonstrate some of the main features of the plot command, including

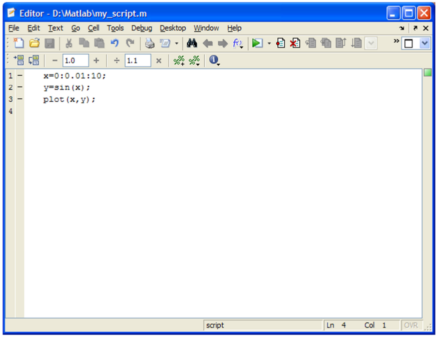
- 2 or more plots on one graph: `hold`
- Adding a legend: `legend`
- Changing linestyles: `plot(...,...,'+--')` will give a dashed line with the + symbol as markers
- Adding titles and labels: `xlabel`, `ylabel`, `title`. Example: `xlabel('this is the label for the x-axis')`
- Changing range of axes: `axis`
- Editing a plot.

demo

Extra windows

Matlab will open extra windows when necessary

- editor


Editor - D:\Matlab\my_script.m
File Edit Text Go Cell Tools Debug Desktop Window Help
script
Ln 4 Col 1 OVR

```
x=0:0.01:10;
y=sin(x);
plot(x,y);
```

Figure: Editor

- figure

Extra windows

Matlab will open extra windows when necessary

- editor
- figure

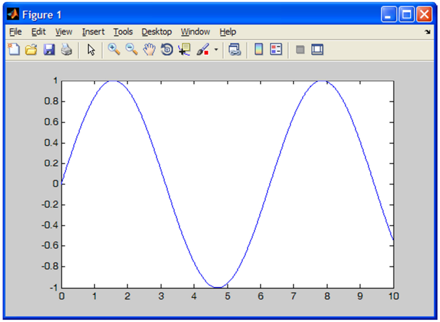


Figure: Figure window

Vector operations – length

- Number of elements, length, of a vector: `length`

  ```
  >> x=[0,pi/2,pi]
  x =
           0    1.5708    3.1416
  >> length(x)
  ans =
       3
  ```

  Note: this is not the physical length of a vector, which is given by `norm`

Vector operations – max and min

- Maximum of a vector: `max`

```
>> x=[0,pi/2,pi]
x =
         0    1.5708    3.1416
>> y=sin(x)
y =
         0    1.0000    0.0000
>> max(y)
ans =
     1
```

- Likewise: minimum of a vector: `min`

```
>> min(y)
ans =
     0
```

Vector operations – change row to column vector and vice versa

To change a row vector in a column vector or the other way around is called *transpose*.

```
>> a = [1, 2, 3]
a =
     1     2     3

>> transpose(a)
ans =
     1
     2
     3
```

Vector operations – transpose

Twice transposing gives the same vector back

```
>> transpose(transpose(a))
ans =
       1        2        3
```

Shortcut for `transpose` is `.'`

```
>> a.'
ans =
       1
       2
       3
```

Vector operations

- Sum of all the elements of a vector

```
>> x=0:pi/2:pi;
>> sum(x)
ans =
    4.7124
>> sum(x)/pi
ans =
    1.5000
```

Vector operations

- Sum of all integers ranging from 1 to 5

```
>> sum(1:5)
ans =
     15
```

- Sum of the inverse of all integers ranging from 1 to 5, i.e. $1 + 1/2 + 1/3 + 1/4 + 1/5$.

```
>> x=1:5
>> y=1./x;
>> sum(y)
ans =
     2.2833
```

Vector elements

We can refer to different elements of a vector as follows:

- Element-wise

```
>> x=[2 3 4 5 6 7];
>> x(2)
ans = 3
>> x(end)
ans = 7
```

- Multiple elements at the same time

```
>> 1:2:5
ans =
     1     3     5
>> x(1:2:5)
ans =
     2     4     6
```

Vector elements

- Accessing multiple elements using a variable

```
>> indices=[1 4 5];
>> x(indices)
ans =
     2     5     6
```

- Setting multiple elements

```
>> x=[2,3,4,5]
x =
     2     3     4     5
>> x(1:2)=[0,1]
x =
     0     1     4     5
```

Vector elements

- Combining two vectors into one longer

```
>> x = [1, 2]
x =
        1         2
>> y = [3, 4]
y =
        3         4
>> z = [x, y]
z =
        1         2         3         4
```

Characters

- Numbers. Just type in a number

```
>> 6
ans =
     6
```

- Characters. Any literal character (e.g., a letter) should be enclosed in quotation marks:

```
>> 'a'
ans =
a
```

Strings

- A string (i.e., a word or a whole sentence) is a *vector* of characters

```
>> 'Hello!'
ans =
Hello!
```

- We used this before:

```
>> title('This is a plot title')
```

Strings - new format

In 2016 Matlab introduced strings with double quotation marks, such as `"Hello"` instead of `'Hello'`. They have different capabilities. In the current module the latter will be mostly used.

Variables with strings

Variables can store numbers, but also characters or strings

- Character

```
>> x = 'a'
x =
a
```

- String

```
>> x = 'ab'
x =
ab
```

- Using a string variable

```
>> x = 'This is a plot title'
x =
This is a plot title
>> title(x)
```

Modifying string variables

- Combining strings: same as with vector of numbers

```
>> txt1 = 'Hi', txt2 = 'ya'
txt1 = Hi
txt2 = ya

>> txt3=[txt1, txt2]
txt3 = Hiya
```

- Changing part of a string

```
>> txt3(2:5)='ello'
txt3 = Hello
```

Relational expressions

Matlab can determine the validity of some relations

- Test if one number is larger than another

```
>> 10 > 5
ans =
  logical
     1
```

This implies the relation is indeed valid: 1 for yes, or *true*, since 10 is larger than 5

```
>> 5 > 10
ans =
  logical
     0
```

This implies the relation is invalid: 0 for no, or *false*, since 5 is not larger than 10

Relational expressions

- Comparison of the number with itself

  ```
  >> 5 > 5

  ans =
    logical
       0
  ```

  A number is not larger than itself.
- Equal or larger: use >=

  ```
  >> 5>=5
  ans =
    logical
       1
  ```

Comparisons

- Equality: use ==:

```
>> 3==1
ans =
  logical
     0

>> 3==3
ans =
  logical
     1
```

Comparison

- Don't use =, since this is reserved for variable assignments!

```
>> 3=3
 3=3
Error: Incorrect use of '=' operator. To
    assign a value to a variable, use '='.
    To compare
values for equality, use '=='.
```

Relational expressions with variables

-
  ```
  >> x=1; y=3;
  >> x^2+2>y

  ans =
    logical
       0
  ```

Relational expressions with characters

-
  ```
  >> result='y';
  result =
  y
  ```

- Compare characters

  ```
  >> result =='y'

  ans =
    logical
        1
  ```

Relational expressions with characters

-
  ```
  >> result='y';
  result =
  y
  ```
- Case sensitive

  ```
  >> result =='Y'

  ans =
    logical
       0
  ```

  'Y' is not equal to 'y'

Logics

True and false values can also be directly entered

- True:

```
>> true
ans =
  logical
     1
```

- False:

```
>> false
ans =
  logical
     0
```

Combined logical operations

- $5 > 3$ and $2 > 1$? Both are true, so result should be true. In Matlab AND can be established using &&

```
>> (5>3) && (2>1)
ans =
  logical
     1
```

- $2 > 1$ or $10 < 3$. The first one is true, and the second one is false, but since it is either one, the total is true. In Matlab OR can be established using ||

```
>> (2>1) || (10<3)
ans =
  logical
     1
```

Logical operations

- Not $10 < 3$? $10 < 3$ is false, so not $10 < 3$ is equal to not false, which should be true. In Matlab NOT can be established using $\sim$.

```
>> ~(10<3)
ans =
  logical
     1
```

Summary relational and logical operators

Relational

- Larger than: $>$
- Smaller than: $<$
- Smaller than or equal to: `<=`
- Larger than or equal to: `>=`
- Equal to: `==`
- Unequal to: `~=`

Logical

- not: $\sim$
- and: `&&`
- or: `||`