Using Wolfram's Mathematica

Johannes Grassberger[1]

[1]Information and Communications Technology Section
International Centre for Theoretica Physics

September 2012

Outline of Topics

General

Quick Tour

Lists

User Defined Functions

Patterns

Conclusion

Mathematica versus other software

+ Mathematica has lots of high-level mathematical knowledge
+ It can handle algebraic expressions, not only numbers
+ It can be used inter-actively
- It is not **freeware**

License

Mathematica is **commercial software**. Don't expect that you can install it on your computer or find it everywhere.
At the ICTP, only 15 copies of Mathematica can run at the same time.
Don't keep Mathematica open when not using it, and don't use it for simple calculations other programs can do.

First Steps

To start the command line interface, open a terminal window and type the command

```
math
```

Try the following inputs:

```
2+3
2*3
5!
2/3
N[2/3]
Sin[Pi/4]
Quit[]
```

Syntax elements

- Use () like in standard mathematical notation
- Square brackets are used for function arguments
- Curly braces are used for lists (which can be used as vectors, matrices etc.)
- The percentage sign refers to the last result, %3 to result number 3, etc.
- Finish your input with a semicolon if you don't want to see the output
- Use a single equal sign to assign a value to a variable
- The double equal sign tests for equality

Graphical frontend

Let's start the graphical frontend by typing

`mathematica`

Select `Create a new notebook`
Here, the normal Enter key will only add a line break, but not evaluate your input.
Use `Shift Enter` or the Enter key of the numerical key pad.

More examples

Try these commands:

```
N[E,10]
Factor[x^5-1]
Solve[x^2 - 5x + 6 == 0,x]
Solve[{x-y == 7,x+2y == 13},{x,y}]
Eigenvalues[{{3,1},{2,6}}]
Inverse[{{3,1},{2,6}}]
Integrate[Sqrt[x] ArcTan[x],x]
Integrate[Sqrt[x] ArcTan[x],{x,0,1}]
```

Plotting

```
Plot[Sin[x]+Cos[1.5x],{x,0,40}]
Plot[Eigenvalues[{{3,1},{2,c}}],{c,-1,10}]
Plot3D[Sin[x y],{x,0,4},{y,0,4}]
```

Internal form of expressions

Mathematica classifies the objects it operates on into types. These can be checked via the Head function:

```
Head[7]
Head[1/7]
Head[.707]
Head[7+2I]
Head[a+b]
Head[{1,2,3,4}]
```

FullForm shows Mathematica's internal representation of an expression:

```
FullForm[a+b]
FullForm[{1,2,3,4}]
```

Forms of Functions

- Infix: $3 + 4$
- The classic prefix form: `Sin[Pi/4],Plus[3,4]`
- Postfix: `Pi/4  // N`

Getting Information

- To get information about a function, type the question mark, followed by the function name
- The same works for variables and functions you have defined:

```
a = 5
?a
```

- You can always use your favourite Internet search engine to find out what function you need and how it works

Creating Lists

```
list1 = {1, 5, 3}
list2 = Range[13]
list3 = Table[i^2,{i,3,20}]
ListPlot[list3]
```

Using lists as vectors and matrices

A simple list can be seen as a row vector. To represent a matrix, use a list whose elements are lists itself (namely the rows of the matrix. E.g. `{{1,2},{3,4}}` would represent the matrix $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$.
Try the following:

```
mymatrix = {{1,2},{3,4}}
MatrixForm[mymatrix]
mymatrix . {{a},{b}}
Transpose[mymatrix]
```

Basic List Operations

```
Length[list2]
Position[list3,25]
Part[list3,4]
list3[[4]]
Take[list3,2]
Drop[list3,2]
Select[list3,EvenQ]
Join[list1,list2]
Intersection[list2,list3]
```

Predicate functions

Predicate functions return either True or False, depending upon whether its argument passes a test.

```
PrimeQ[91]
OddQ[5]
EvenQ[4]
IntegerQ[5/9]
3 == 7 - 4
```

Advanced List Operations

`Map` applies a function to each element in a list.
E.g. `Map[f,{3,5,6,7}]` gives `{f[3],f[5],f[6],f[7]}`
`Apply` uses the elements of a given list as function arguments:
`Apply[f,{3,5,6,7}]` gives `f[3,5,6,7]`
Try: `Apply[Plus,Range[10]]`
`Outer` applies a function to all possible combinations of elements from several lists. Try: `Outer[f,{1,2},{3,4}]`

Defining a simple function

You can define a function yourself using the following form:

```
f[x_] := x^2
```

Note:

- On the left-hand side, the variable names are followed by underscores. The reason for this will be explained later.
- As assignment operator := used. This tells Mathematica to evaluate the expression on the right-hand side only when a value for x is given.

Simple function (continued)

You can now use the function like any of Mathematica:

```
f[5]
Plot[f[x],{x,0,10}]
?f
```

To erase the definition, use `Clear[f]`

Patterns Elsewhere

We have already seen patterns when working with Linux. For example, `ls -d publ*` will list all files and directories the names of which start with the letters `publ` and are followed by any numbers of characters.

Patterns in Mathematica

In Mathematica, patterns are designed to work well with mathematical expressions. To see whether an expression matches a certain pattern, we can use the function `MatchQ`:

```
MatchQ[x^2, x^_Integer]
MatchQ[x^2, _^_]
MatchQ[x^2, _List^_Integer]
MatchQ[x^2, _^_?EvenQ]
MatchQ[{1,2}^3, _List^_Integer]
```

Patterns in function definitions

We can provide more than one definition for a single function:

```
h[n_?EvenQ] := n/2
h[n_?OddQ] := 3n + 1
?h
h[5]
h[6]
```

Recursive functions

We could define the Fibonacci series like this:

```
fib[1] := 1
fib[2] := 1
fib[n_] := fib[n-1] + fib[n-2]
fib[5]
?fib
Trace[fib[5]]
```

In order to store and re-use results already calculated, do this:

```
fib[n_] := fib[n] = fib[n-1] + fib[n-2]
Trace[fib[5]]
```

Functions in Mathematica

As opposed to most programming languages, Mathematica sees function definitions as rewrite rules that it may apply whenever the given pattern matches.
It's like adding some additional knowledge to its database that knows how to factor polynomials, solve systems of linear equations etc.

Further Reading and Thanks

See shelf 519.688 of the ICTP Library.
In particular, I owe my thanks to the authors of *Introduction to Programming with Mathematica*, **519.688 GAY**.