Introduction to MATHEMATICA

Spiridon Penev

Abstract.

This is a light introduction into MATHEMATICA. It is a set of recipes and observations I have come into when running MATHEMATICA which I would like to share with you if you for a quick start. There is no need of thorough teaching material in MATHEMATICA since the help and the "Get Started" materials are comprehensive enough.

1 Introduction.

The best way to learn MATHEMATICA is to first get a feeling about the system by typing in some of the examples in the notebook window. After having got enough of a feeling, you can start modifying the examples and getting into more complicated ones. The next step is about **learning** to go through what is needed to solve a complete problem with MATHEMATICA. The MATHEMATICA system has a logical overall structure but one is not advised to go into it too early before having got enough experience with solving particular simple problems. To really understand the overall structure of MATHEMATICA, one needs a broad experience with an advanced computer language. We will not try to do this here.

While almost any MATHEMATICA program can be written in a procedural way like in languages like Fortran, C, etc. this is rarely the best approach to utilize the power of the MATHEMATICA system. In a symbolic system like MATHEMATICA, functional and rule-based programming help to design programs that are more efficient and easy-to-understand.

2 MATHEMATICA as a calculator

- Arithmetic operations are grouped according to the standard mathematical conventions.
- You can enter numbers in scientific notation
- You can tell MATHEMATICA to give you an approximate numerical result using //N
- Whenever you give a number with an explicit decimal point, MATHEMATICA produces an approximate numerical result. If there is a rational number and a dot is not used, the result is again a rational number reduced to its lowest terms
- Arguments of all MATHEMATICA functions are enclosed in *square brackets*, not in parentheses. Parentheses are reserved in MATHEMATICA for indicating the grouping of terms. This distinction has several advantages.
- Multiplications are represented by a space but could be represented by using * as well.
- The names of built-in functions begin with *capital letters*: `Abs, Sqrt, Exp, Sin, Round` etc.
- There are some standard values like `Pi, E, I, Infinity`
- You can perform basically *arbitrary precision calculations* using `N[expr,n]` (it gives you numerical value of `expr` with `n` digit precision)
- Complex numbers can be handled freely: `Sqrt[-4]` gives you 2I. Functions to be used in complex arithemtics are (amongst others: `Re[z], Im[z], Conjugate[z], Abs[z], Arg[z]`)
- When you use *Notebook based interface* you can utilise the many palettes provided which operate like extensions to your keyboard. This really alleviates your programming a lot. You access these palettes by clicking on the `Palettes` submenu.

3 Building up calculations

- Defining variables: `x=value`, `x=y=value`, `x=.` or `Clear[x]` to remove any assigned value to `x` (It is advisable to remove assigned values once the variable is not used anymore).
- There are four kinds of bracketing in MATHEMATICA: *(term)* for grouping, `f[x]` square brackets for functions, {a,b,c} for lists, v[[i]] for indexing Part[v,i] in the list v.
- Each step in your program can be put on a separate line. If several steps are put on the same line you should separate them by semicolons.
- Using the notebook interface is not compulsory. For example, you can integrate $\int \frac{1}{x^3-1} dx$ by applying the text command

  ```
  Integrate[1/(x ^3-1),x]
  ```

  (try it) but it is much simpler to use the palette!
- The notebooks are structured into a sequence of `cells`, each cell containing material of any type- be it text, graphics, sounds etc. The extent of each cell is indicated by a bracket on the right. These brackets can also be utilized to easily delete unnecessary cells. SHIFT-ENTER sends a cell of input to be executed by the MATHEMATICA kernel (the part of the system that actually performs the calculations).

4 Algebraic Calculations

This is a very significant strength of the MATHEMATICA system. The simplest thing it does is to automatically perform algebraic simplifications. When you type an algebraic expression using the standard operators and press SHIFT-ENTER, it gets simplified in the output if any simplification is possible. *Do not forget* that if you want to indicate the product of `x` and `y` you should write `x y` (with a space between them), otherwise `xy` will be interpreted as a single name, not as a product of two symbols.

Additional features:

- The function `Expand[]` multiplies out products and powers; `Factor[]` does the inverse to `Expand[]`.

- You can **replace** values using the pattern *expr* /. *rule* Try for example

  ```
  1+x+x^2 /. x -> 2-y
  ```

  and explain to yourself the result. You can perform several replacements simultaneously:

  expr /. $\{x- > xval, y- > yval\}$

- If you want to define (assign) a value for `x` that will *always* be used, write `x=value`, for example `x=3`. **There is a crucial difference:** replacement only operates **locally** on the specific expression whereas `x=3`, for example, works **globally**.

- It is a good idea to clear the value of a variable you are going to use so that you are sure it does not contain unwanted content. Use `Clear[x]` for example or just `x=.`

- You can use commands `Simplify[]` or `FullSimplify[]` to find the simplest form of an expression by applying a wide range of transformations. The former applies *algebraic* transformations only whereas the latter can perform even more difficult tasks. For example,

  `Simplify[x Gamma[x]]` does not do anything but try

  `FullSimplify[x Gamma[x]]`.

  Of course, FullSimplify might be too slow and should not be used if the transformation required is a simple algebraic transformation.

- Other options to try: `Expand, Factor, Together, Apart, Cancel, TrigExpand, TrigFactor, TrigReduce, TrigToExp, ExpToTrig` etc.

- Some operations are not always executable and because of this, you have an option to execute a simplification *with assumption*. For example,

  ```
  Simplify[Sqrt[x^2]]
  ```

just leaves you with $\sqrt{x^2}$ (which is the result that *always* holds but if you write

```
Simplify[Sqrt[x^2], x > 0]
```

the result will be `x` since under the assumption of positive `x` this further simplification is possible. The general form of the command is `Simplify[expr, assum]`

5 Lists and Tables, vectors and matrices

Lists are a very general and important concept in MATHEMATICA. Like in S-PLUS, lists are very general structures that can contain **any** kind of Mathematica objects, including numbers, variables, functions, and equations. The for of the list is $\{x,y,z\}$ where *x,y,* etc. are the elements of the list. In many cases, you can treat the list like a single object by executing operations on its elements component-wise. For example if

`w=1,2,5,7` then when you type

```
w^2
```

you get 1,4,25,49 as a result. Extracting the `n`th element of the list happens through `w[[n]]` or equivalently, `Part[w,n]`. Specifying a list of positions, you can extract a sublist, e.g

`w[[2,4]]` would give you the list $\{2, 7\}$.

Two lists can be concatenated in a bigger list via `Join[list1,list2]`

A very useful and powerful command is the `Table` command. It really can be used to generate tables which are sorts of lists. It can be used also as a substitute to computing intensive `For` loops. A simple illustration:

```
f[x_]:=x^3+5x-7

Table[f[x],{x,0,10,2}]
```

calculates the function's value for $x = 0, 2, 4, 6, 8, 10$ and concatenates the outcomes in a list. The result of the execution of the Table command is the list $\{-7, 11, 77, 239, 545, 1043\}$. Here the increment of 2 had to be specified but if it was one, it could have been omitted. Tabulated form of the outcome in a form of argument, function value on each row can be obtained via the command

```
Table[{x,f[x]},{x,0,5}] //TableForm
```

Try it.
Vectors in MATHEMATICA are represented by lists; matrices are represented as lists of lists. The sublists are the **rows** of the matrix. For example:

```
mat={{1,8,3},{2,4,6},{8,1,-2}}
```

is a matrix and its first row has the elements 1,8 and 3. Try now `MatrixForm[mat]` to display the same matrix in the usual matrix form. Many commands are available to manipulate matrices in MATHEMATICA. You can consult the help about them.

6 Symbolic Mathematics

Differentiation, Integration, Sums and Products, Algebraic (Systems of) Equation Solving, Solving Differential Equations, Power Series Expansions, Limits, Integral Transforms etc.: all these topics can be dealt with and are described thoroughly in the help. Besides, there are several packages for advanced symbolic mathematics specifically tailored for Vector Analysis, Solving Inequalities, Solving Recurrence Relations etc.

7 Most common mistakes when using MATHEMATICA as a novice

I can summarize them as follows:

- Forgetting to clear values
- Not keeping track of In/Out numbering when executing commands within a given notebook
- Improper use of built-in functions
- Improper use of arithmetic symbols using a command from a MATHEMATICA package before the package has been loaded
- Using (typing) the wrong type of equal sign

- Omitting the `Evaluate` command inside a plotting command
- Using lower case letters instead of upper case letters in Mathematica commands or misspelling (especially lengthy) commands.

8 Using MathStatica

MathStatica is an add-on to MATHEMATICA. It provides a specialised methodology for doing Statistics. It exploits the power of MATHEMATICA especially in the area of symbolic calculations. There is an online book that comes with the add-on.

MathStatica is intended to help students and researchers. No specific knowledge of MATHEMATICA is indeed required in order to be able to use MathStatica. The many patterns and examples provided in the online book can be easily understood by the customer who then can easily modify/extend them according to his/her needs.