

Getting started with *Mathematica* 6 or 7

This document is available in PDF and *Mathematica* notebook form at <http://facstaff.unca.edu/mcmclur/Mathematica/>.

■ Installation

You can get a disk from me or the ITS office. It should have the license number with it. Note that disks are specific to Mac/PC or faculty/student and need the appropriate license number. Installation is fairly routine. If it is being installed on a campus machine, you should use the license server: `macadmin1.unca.edu`. If you want to install it on a home machine or one that cannot access the university server, you will need to go through the registration process. Be sure to use your UNCA email when you register. After registering, you should receive an email from Bob Benites with your password.

■ The expanded first five

When you open up *Mathematica*, you are presented with a blank untitled screen. The pressure is on you to come up with something. Well, it's not quite so bad; there's also probably a small Startup or Welcome window depending on whether you're using V6 or V7. The V6 Startup window has a button labelled "First Five Minutes with *Mathematica*" document, which you can always open under the Help menu. The first five tutorial is an interactive notebook that introduces the very basics of using *Mathematica*. You might think of the document in your hands as an expansion of that tutorial; it contains all the first five commands and a few others, together with a bit of explanation. The V7 Welcome window has a button labelled "Learn with guided examples" that has many of the same items as the V6 version, plus a bit more.

The most extensive source of help with *Mathematica* 6.0, is the Documentation Center available under the Help menu.

Basic computations

The first thing the tutorial asks you to do is add 2 and 2. You can do so by typing the following command *exactly as shown* and hitting `ENTER`, the enter key.

```
2 + 2
```

Please take the phrase "exactly as shown" seriously. Details such as capitalization, brackets `[]` versus braces `{}`, and the like tend to be important. Also, `ENTER` and `↵` (return) are different. On many systems, `ENTER` is equivalent to `SHIFT-↵`, shift-return.

The next computation is a bit more impressive.

```
500!
```

The result has over 1000 digits. You can compute the exact number of digits like so.

```
Length[IntegerDigits[%]]
```

Note that `%` refers to the previous output. Previous means most recently generated and might or might not refer to the output directly above. If you re-execute the last command, you should get 4.

Speaking of 1000 digits, here's 1000 digits of π .

```
N[Pi, 1000]
```

Note that there is a big difference between **Pi** and 3.14; one is an *exact* number, the other is a decimal approximation. There is even a big difference between 3 and 3.0. Usually, computations with decimal approximations will run much faster than exact computation. The following command indicates that they are treated differently in computations.

```
{Pi - 3, Pi - 3.0}
```

2D Graphics and lists

Next, we learn how to plot a function.

```
Plot[Sin[x], {x, 0, 10}]
```

Note to version 5 users: A semi-colon at the end of line suppresses all output - including graphical output.

Most commands may be input in a variety of formats and have a variety of options. Next, we use a list to plot two functions and we use a couple of options to affect the appearance. Options are specified using rules with arrows: \rightarrow , which can simply be entered as \rightarrow .

```
Plot[{Sin[x], Cos[x]}, {x, 0, 3 Pi},
  Filling  $\rightarrow$  True, Ticks  $\rightarrow$  {
    Table[x, {x, 0, 3 Pi, Pi / 4}], {-1, 1}}]
```

Note that we've set up lists in two different ways. The list of functions is enclosed in braces: this is the sole purpose of braces in *Mathematica*. The ticks on the x -axis are also represented as a list, but that list is generated using the **Table** command.

```
Table[x, {x, 0, 3 Pi, Pi / 4}]
```

More generally, the syntax is

```
Table[expr, {x, xMin, xMax, xStep}]
```

Dynamic expressions and 3D

The previous syntax works for a variety of commands, including **Manipulate** which can be used to generate dynamic output.

```
Manipulate[x, {x, 0, 3 Pi, Pi / 4}]
```

This should generate a dynamic box with a slider that you can manipulate. If you delete the **xStep** parameter, the value varies nearly continuously.

```
Manipulate[x, {x, 0, 3 Pi}]
```

You can use **Manipulate** with arbitrary *Mathematica* code. This can be used to generate interactive presentations of basic concepts. This is the point behind the next command, which demonstrates the affect of a coefficient inside a trig function on frequency.

```
Manipulate[Plot[Sin[c * x], {x, 0, 10}], {c, 1, 5}]
```

The next command in the tutorial mixes this with 3D graphics.

```
Manipulate[Plot3D[Sin[x + y + c], {x, 0, 6}, {y, 0, 6}], {c, 1, 5}]
```

All the demonstrations from the Wolfram Demonstration site were generated in this way. You can see these here: <http://demonstrations.wolfram.com/>.

Three dimensional graphics in V6 are automatically rotatable and, generally, a huge improvement. Here's a somewhat interesting plot.

A bit more 3D

```
Plot3D[{Sin[x + Cos[y]] + 1}, {x, -3, 3}, {y, -3, 3}]
```

Here's a darker version of it, restricted to lie in a sphere and named so we can refer to it. Note that the **RegionFunction** is represented using a pure function.

```
restrictedPlot = Plot3D[{Sin[x + Cos[y]] + 1}, {x, -3, 3}, {y, -3, 3},
  RegionFunction -> (#1^2 + #2^2 + #3^2 < 9 &),
  PlotStyle -> Blue, BoundaryStyle -> Thick]
```

Here's a clear image of the containing sphere (represented using graphics primitives).

```
clearSphere = Graphics3D[{Opacity[0.5], Sphere[{0, 0, 0}, 3]}]
```

Here they are together.

```
Show[restrictedPlot, clearSphere,
  PlotRange -> {0, 3}]
```

Algebra and calculus

Mathematica can do algebra.

```
Expand[(x + h)^8 - x^8] / h
```

It can take limits.

```
Limit[%, h -> 0]
```

Thus, I guess it can do calculus.

```
D[x^8, x]
```

Even integration. Here is the tutorial's example:

```
Integrate[1 / (x^5 - 1), x]
```

They make it dynamic, too.

```
Manipulate[Integrate[1 / (x^n - 1), x], {n, 2, 10, 1}]
```

■ Solving equations

Mathematica can solve equations symbolically and numerically. Note that equations are represented using double equals signs.

```
Solve[x^2 + 2 x - 1 == 0, x]
```

Even simple polynomial equations can yield complicated symbolic results.

```
Solve[x^3 + 2 x - 1 == 0, x]
```

NSolve can be used as a numerical polynomial solver.

```
NSolve[x^3 + 2 x - 1 == 0, x]
```

Other simple equations are not amenable to the techniques of **Solve** or **NSolve**.

```
NSolve[Cos[x] == x, x]
```

We can see that there is a solution close to 0.8, however.

```
Plot[{Cos[x], x}, {x, 0, 1}]
```

In this case, the **FindRoot** command is the correct numerical tool to use to get an estimate. Note that the command accepts an initial guess, which we have from the plot.

```
FindRoot[Cos[x] == x, {x, 0.8}]
```

Integrated data sources

Here is the final example in the first five tutorial.

```
KnotData[{5, 2}]
```

Most functions of the form ***Data** access servers at Wolfram research, so they require internet access. The **KnotData** function is a bit abstract, but others are much more concrete. Here's a plot of the solar system.

```
Graphics3D[AstronomicalData[#, "OrbitPath"] & /@  
  AstronomicalData["Planet"],  
  ImageSize -> Large]
```

Using other curated data sources, it is just as easy to plot the Dow Jones average, the chemical structure of Caffeine, a map of Europe, or the Platonic solids. Here's why I don't drink Diet Coke.

```
ChemicalData["Nutrasweet", "NaturalProduct"]
```