Notes on *Mathematica*

Mathematica is a general purpose mathematics software system. You can think of it as a powerful graphing calculator. When it originated, *Mathematica* was often to referred to as a *computer algebra system (CAS)*. CAS programs are characterized by allowing for *symbolic* calculations in contrast to *numeric* calculations. The phrase CAS does not fully describe the capabilities of *Mathematica* or other similar programs.

Other similar commercial programs/packages include *Matlab* and *Maple*. *Sage* is an open source system that is being developed as a freely available alternative to commercial products. You can learn more about *Sage* at `sagemath.org`

These systems have different origins but have converged to include roughly the same capabilities. Different points of origin mean each has different relative strengths and weaknesses. I have not carefully evaluated those strengths and weaknesses. I use *Mathematica* primarily because it is the system I learned first and now know best.

Here's some quick notes on getting started:

- basic operation: type input, hit `enter` or `shift-return` for output
- two pieces: front end and kernel
- note In[] and Out[] labels added
- can refer to previous output using Out[x] or %x
- % refers to the most recent output
- note cell structure
- can edit, copy and paste
- order of execution is what matters, not order of appearance in the notebook
- multiple notations: keyboard, palettes, Esc
- multiplication with * or space
- role of brackets: square, curly, rounded
- distinguish among = and := and ==
- built-in values and commands use upper case (Pi, E, I, Factor)
- built-in functions have required arguments and optional arguments
- assigments are global and persistant unless you say otherwise
- assignments are not automatically initialized unless you say otherwise
- documentation is available

The accompanying handout lists a variety of inputs for you to try. For each, you should type in the given input and then have *Mathematica* produce the corresponding output. Study the output to see how it relates to the input. Experiment with variations on the input to see what each piece of the input does.