Mathematica techniques used in notebooks 12-22-16

Purpose

Mathematica/Wolfram Language is an extraordinarily flexible computer language and programming environment. There are different ways to accomplish almost any mathematical task and practitioners develop quite different styles. For example, notice the range of approaches taken by different experts when answering questions posed at *mathematica.stackexchange.com/questions*.

I have a personal style developed over the years and some of the methods I use will appear unusual, especially to new users of *Mathematica*. My style has evolved — partly because I gained experience and facility, partly because of advances in *Mathematica* technology. When I embarked on this effort to resurrect old technical works, I thought it would mainly be a matter of adding annotations to existing notebooks. However, I found myself cringing at the code I generated years ago and could not resist tweaks and rewrites. You will be disappointed if you expect rigid consistency between the notebooks presented on this website.

In this particular notebook, my intent is to isolate and discuss some of the techniques I use.

Initializing a subject matter notebook
Getting Information and help
Labeling sequential operations
Shortcuts for Using Functions

Initializing a subject matter notebook

On opening a subject matter notebook, you will see the following

Initialization: Be sure the files *NTGStylesheet2.nb* and *NTGUtilityFunctions.m* is are in the same directory as that from which this notebook was loaded. Then execute the cell immediately below by mousing left on the cell bar to the right of that cell and then typing “shift” + “enter”. Respond “Yes” in response to the query to evaluate initialization cells.

In[3]:=
```
SetDirectory[NotebookDirectory[]];
(* set directory where source files are located *)
SetOptions[EvaluationNotebook[], (* load the StyleSheet *)
 StyleDefinitions → Get["NTGStylesheet2.nb"]];
Get["NTGUtilityFunctions.m"]; (* Load utilities package *)
```

Executing this code loads a package of convenience functions that are used in most notebooks. It also establishes a "style" for the notebook that I find pleasing. See the *Mathematica* documentation on **Stylesheet**.

SO - EXECUTE THIS CODE RIGHT NOW. Mouse click left in some part of the gray area and type shift-enter.

My personal convenience functions are contained in the notebook ***Utility Functions.nb*** (source code with annotations) and **Utility Functions.ma** (loadable "package" notebook).

Getting information and help

Mathematica (aka *Wolfram Language*) has extraordinarily good documentation available with a simple mouse click on ***Help***. This documentation is rich with information, search capabilities, cross references, tutorials and collections of examples. Even with 20 years of experience, I typically use the ***Help*** facility several times during every programming project. There is much to be learned and you should explore.

Also, the ***Wolfram.com*** site has many videos that introduce *Mathematica* and how it can be used. It is a good idea to view these videos. Many of the operational aspects of using *Mathematica* make much more sense when you can see somebody performing those operations. There are also scores of YouTube videos by Mathematica users.

One of the best sources of information for experienced Mathematica users is http://mathematica.stackexchange.com/questions. At that site there exists a invaluable collection of Mathematica resources — http://mathematica.stackexchange.com/questions/18/where-can-i-find-examples-of-good-mathematica-programming-practice/259#259.

In my experience, the best way to learn guru level coding techniques in *Mathematica* is to study the masterful books by Michael Trott. http://www.mathematicaguidebooks.org/

Suppose you working with a subject matter notebook and encounter the function ***Sol,*** **a** function not found in *Mathematica* ***Help***. In fact, ***Sol*** is one of the functions defined in **Utility Functions.ma.** Information on ***Sol*** can be obtained by typing

```
Information[Sol]
```

```
Global`Sol
```

```
Sol[eqn_, var_] := Solve[eqn, var][[1, 1]]
```

or its short hand form

```
?? Sol
```

```
Global`Sol

Sol[eqn_, var_] := Solve[eqn, var][[1, 1]]
```

This provides the definition of Sol in terms of the built-in Mathematica function ***Solve***. Want to know about ***Solve***?

```
?? Solve
```

```
Solve[expr, vars] attempts to solve the system expr of equations or inequalities for the variables vars.
Solve[expr, vars, dom] solves over the domain dom. Common choices of dom are Reals, Integers, and Complexes. >>

Attributes[Solve] = {Protected}

Options[Solve] =
 {Cubics → True, GeneratedParameters → C, InverseFunctions → Automatic, MaxExtraConditions → 0,
  Method → Automatic, Modulus → 0, Quartics → True, VerifySolutions → Automatic, WorkingPrecision → ∞}
```

Want to know still more? Mouse click left on the >> on the second line and you will be taken to the full Mathematica help section on ***Solve***.

What is *Sol* for? It's just a short hand way of getting rid of the annoying brackets in

```
Solve[y == m x + b, x]
```

$$\left\{\left\{x \to \frac{-b+y}{m}\right\}\right\}$$

For example,

```
Sol[y == m x + b, x]
```

$$x \to \frac{-b+y}{m}$$

Sol is just some personal syntactic sugar *https://en.wikipedia.org/wiki/Syntactic_sugar*

Life is too short to spend time guessing what syntax was used for what operation in computer languages. Particular syntaxes were often chosen according the personal preferences of the language designer, the preferences of some standards committee, or for obscure historical reasons. Be quick to use ***Help***. Be quick to look at examples of working code. Be quick to ask questions.

Labeling sequential operations

Consider a typical physics derivation. One starts with some general principle that has a mathematical representation and then systematically specializes that principle with some objective in mind. Assumptions are introduced, side calculations are performed, simplifications are made — with the goal of generating a different mathematical representation that has utility in some specialized context. Typically, there are many steps in this process and some mechanism is needed to keep track of intermediate results. Mathematica notebooks have ***In*** and ***Out*** labeling and shortcuts for referring to previous expressions - ***%***, ***%%,*** etc. that are useful in various contexts.

However, I am never smart enough to proceed in a linear fashion from the beginning to the end of a calculation. As I make grudging progress toward some typically ill-defined goal, I am constantly refining, revising, getting confused, going off on tangents or just going down blind alleys.

So I have a developed a simple labeling system. Rather than discuss the labeling as an abstract concept, I illustrate its application in the context of a simple Freshman physics derivation — deriving the two-dimensional equations of motion for a projectile with mass m under gravity.

Since this example is intended for beginners I will be very basic in terms of notation and technique and proceed step by step. In the regular subject matter notebooks, I take shortcuts, and if you become an experience user of *Mathematica*, you will too. However, I note that one of the advantages of developing a topic in *Mathematica* is that the intermediate details can be shown in excruciating detail if that is desired. Pixels are free and there are no journal editors or referees. All of the algorithmic steps can be generated if desired. On the other hand, details can be easily hidden for aesthetic purposes.

To separate physics and *Mathematica* motives, I will use a smaller font when I am discussing purely Mathematica related issues.

I start with Newton's F = m a in component form for two dimensional motion in the x-y plane.

```
w[1] = {Fx == m ax, Fy == m ay}
```

```
{Fx == ax m, Fy == ay m}
```

Here w is a basic Mathematica entity known as a ***Symbol***. Writing
w[1] = something
means "assign the ***DownValues*** 1 of the ***Symbol*** w to the expression something." Look at Help for ***DownValues.***

Next, I specialize Newton's law to apply to motion under a gravitational force in the -y direction and assume there are zero forces in the x direction.

```
w[2] = w[1] /. {Fx → 0, Fy → -g} /. {ax → D[x[t], {t, 2}], ay → D[y[t], {t, 2}]}
```

$$\{0 == m\, x''[t],\ -g == m\, y''[t]\}$$

This last operation involved applying replacement rules via a postfix operator -- but the result is rather obvious. I discuss these techniques in detail below.

I solve these equations

```
w[3] = DSolve[w[2], {x[t], y[t]}, t]
```

$$\left\{\left\{x[t] \to C[1] + t\, C[2],\ y[t] \to -\frac{g\, t^2}{2\, m} + C[3] + t\, C[4]\right\}\right\}$$

You get the picture. I simply label each step of the calculation with w[1], w[2], ... so I can easily to refer to previous steps.

In preparation for applying initial conditions I write

```
w[4] = {w[3], Map[Function[rule, D[rule, t]], w[3]]} // Flatten
```

$$\left\{x[t] \to C[1] + t\, C[2],\ y[t] \to -\frac{g\, t^2}{2\, m} + C[3] + t\, C[4],\ x'[t] \to C[2],\ y'[t] \to -\frac{g\, t}{m} + C[4]\right\}$$

```
w[5] = w[4] /. t → 0 /.
   {x[0] → x0, y[0] → y0, x'[0] → v0x, y'[0] → v0y} /. Rule → Equal
```

$$\{x0 == C[1],\ y0 == C[3],\ v0x == C[2],\ v0y == C[4]\}$$

where ***Flatten*** gets rid of extraneous brackets. Also, one typically changes expressions from equation form (required for DSolve, etc) to rule form (required for replacements) and vice versa. That is accomplished here with the postfix operations /. Rule → Equal, or /. Equal → Rule.

Calculate the constants of integration

```
w[6] = Solve[w[5], {C[1], C[2], C[3], C[4]}] // Flatten
```

$$\{C[1] \to x0,\ C[2] \to v0x,\ C[3] \to y0,\ C[4] \to v0y\}$$

and write the final expressions for the explicit x and y motion.

```
w[7] = w[3] /. w[6] /. Rule → Equal // Flatten
```

$$\left\{x[t] == t\, v0x + x0,\ y[t] == -\frac{g\, t^2}{2\, m} + t\, v0y + y0\right\}$$

Note: all of the intermediate results have been saved. Use ***Information*** to see them.

```
?? w
```

```
Global`w
```

$$w[1] = \{Fx == ax\, m,\ Fy == ay\, m\}$$

$$w[2] = \{0 == m\, x''[t],\ -g == m\, y''[t]\}$$

$$w[3] = \left\{\left\{x[t] \to C[1] + t\, C[2],\ y[t] \to -\frac{g\, t^2}{2\, m} + C[3] + t\, C[4]\right\}\right\}$$

$$w[4] = \left\{x[t] \to C[1] + t\, C[2],\ y[t] \to -\frac{g\, t^2}{2\, m} + C[3] + t\, C[4],\ x'[t] \to C[2],\ y'[t] \to -\frac{g\, t}{m} + C[4]\right\}$$

$$w[5] = \{x0 == C[1],\ y0 == C[3],\ v0x == C[2],\ v0y == C[4]\}$$

$$w[6] = \{C[1] \to x0,\ C[2] \to v0x,\ C[3] \to y0,\ C[4] \to v0y\}$$

$$w[7] = \left\{x[t] == t\, v0x + x0,\ y[t] == -\frac{g\, t^2}{2\, m} + t\, v0y + y0\right\}$$

Note: I didn't have to use numbers for the ***DownValues***. I could have written

```
w["final result"] = w[3] /. w[6] /. Rule → Equal // Flatten
```

$$\left\{x[t] == t\, v0x + x0,\ y[t] == -\frac{g\, t^2}{2\, m} + t\, v0y + y0\right\}$$

Note: Long complex derivations are naturally separated into parts, I will often use w1[1], w1[2],... to label results in Section 1, so as to distinguish from w2[1], w2[2],... in Section 2. If there is an Appendix A, I would write wA[1], wA[2],... There are few restrictions here, you can label long sequential calculations as you wish.

For the *Mathematica* cognoscenti — I know, I know. I could have derived this result in only a line or two of terse, cryptic, arguably elegant code. I could have used subscripts to resonate with traditional mathematical notation. I could have been more efficient or satisfied some other aesthetic. This discussion is intended for beginners.

Shortcuts for Using Functions

Functions are ubiquitous in *Mathematica.* There are thousands of built-in functions in the latest version *Mathematica 11*. Even a casual user of *Mathematica* quickly begins programming and developing their own functions. In the notebooks presented on this website, the most common operation will consists of applying some mathematical transformation to a data structure by operating on that structure with a function. Someone new to *Mathematica* should read the documentation on ***Function***, follow the pointers to other documents, and read tutorials like ***tutorial/FunctionalOperationsOverview***, ***guide/FunctionCompositionAndOperatorForms***, ***tutorial/SomeGeneralNotationsAndConventions.***

In this section, I will limit myself to those methods that I frequently use that might be confusing to a *Mathematica* beginner.

Suppose we have some function F that is applied to argument x. I could write

```
F[x]
```

```
F[x]
```

where, for the time being, I leave F unspecified

```
?? F
```

```
Global`F
```

I can apply function F to argument x in several different ways —

```
{F[x], F@x, x // F}
```

```
{F[x], F[x], F[x]}
```

Why choose one over the other? For operational convenience! — it's just another example of syntactic sugar.

Suppose you are carrying out a calculation you apply some operation and arrive at some expression whose form you don't like

```
w[1] = Expand[(Sin[x] + Exp[-x] Cos[x])^3]
```

$$e^{-3x}\,\text{Cos}[x]^3 + 3\,e^{-2x}\,\text{Cos}[x]^2\,\text{Sin}[x] + 3\,e^{-x}\,\text{Cos}[x]\,\text{Sin}[x]^2 + \text{Sin}[x]^3$$

You could type a new line

```
w[2] = Simplify[w[1]]
```

$$e^{-3x}\,\left(\text{Cos}[x] + e^{x}\,\text{Sin}[x]\right)^3$$

However, its simpler to just edit the existing command by adding a postfix operator

```
w[1] = Expand[(Sin[x] + Exp[-x] Cos[x])^3] // Simplify
```

$$e^{-3x}\,\left(\text{Cos}[x] + e^{x}\,\text{Sin}[x]\right)^3$$

Equivalently, with a bit more labor you could edit the existing command to read

```
w[1] = Simplify@Expand[(Sin[x] + Exp[-x] Cos[x])^3]
```

$$e^{-3x}\,\left(\text{Cos}[x] + e^{x}\,\text{Sin}[x]\right)^3$$

I prefer the postfix form and will quite often make a succession of transformations to an expression to achieve some objective

```
F[x] // G // H
```

```
H[G[F[x]]]
```

Another important technique is the use of “pure functions.” The syntax for pure functions is not intuitive and you should definitely read ***tutorial/PureFunctions*** and look at examples.

but, for example, suppose you want to cube each term in a list of expressions

```
w[1] = {x, x + y, Sin[x] + Exp[-x]}
```

$$\left\{x,\ x+y,\ e^{-x}+\text{Sin}[x]\right\}$$

You don’t want to retype this list of expressions

```
{x^3, (x + y)^3, (Sin[x] + Exp[-x])^3} // Expand
```

$$\left\{x^3,\ x^3+3\,x^2\,y+3\,x\,y^2+y^3,\ e^{-3\,x}+3\,e^{-2\,x}\,\text{Sin}[x]+3\,e^{-x}\,\text{Sin}[x]^2+\text{Sin}[x]^3\right\}$$

You suspect Mathematica has a function called ***Cube*** and try

```
Map[Cube, w[1]]
```

$$\left\{\text{Cube}[x],\ \text{Cube}[x+y],\ \text{Cube}\left[e^{-x}+\text{Sin}[x]\right]\right\}$$

but it doesn’t. Instead this operation is easily accomplished with

```
(#^3) & /@ w[1] // Expand
```

$$\left\{x^3,\ x^3+3\,x^2\,y+3\,x\,y^2+y^3,\ e^{-3\,x}+3\,e^{-2\,x}\,\text{Sin}[x]+3\,e^{-x}\,\text{Sin}[x]^2+\text{Sin}[x]^3\right\}$$

where $(\#^3)\&$ is a pure function and /@ is shorthand for Map -- examples of syntactic sugar. A longer was of accomplishing this would be

```
Map[Function[{x}, x^3], w[1]] // Expand
```

$$\left\{x^3,\ x^3+3\,x^2\,y+3\,x\,y^2+y^3,\ e^{-3\,x}+3\,e^{-2\,x}\,\text{Sin}[x]+3\,e^{-x}\,\text{Sin}[x]^2+\text{Sin}[x]^3\right\}$$

When reading *Mathematica* code written by experienced coders, you will see a LOT of pure functions, and a lot of short hand expressions like /@.

Depending on the response of readers, I may add more notebooks like this so check the Guideline tabs from time to time.