Keeping paper current

Ben Klemens

9 February 2010

I had ambitious goals for the textbook—so ambitious, they were impossible. Here, I'll talk about the sort of things that keep the typical textbook author (i.e. me) up at night. It is a sort of apology for the fact that one of these three goals eventually had to give.

Goal 1: Write about merging statistical technique with simulation technique. I've written about merging modeling paradigms[1] before, and I still think it's something that's truly novel about the book. It surprises people. Folks who come from a simulation background typically sense the paradigm-merging intent quickly, and work out how the stats-oriented parts of the book could readily be applied to their computation-oriented models. I've found that people coming from the stats side have more trouble seeing the connection, mostly if they haven't put much thought into computational modeling. But even a number of people from that side eventually came back to me with comments about how that thing they didn't see the point of in Chapter Four turned out to be really useful.

OK, so far so good. Although it's not something everybody particularly cares about, I am happy with the design decision of writing a textbook on merging stats and simulation technique.

Goal 2: No pseudocode Pseudocode is good for exposition, but it typically doesn't solve the problem the reader has. First, there are often real problems in handling the data that are hidden under the algorithm—and yes, I've tried it in your favorite language; the quirks were different but the overall effort was not.

For example, say that $f(\cdot)$ and $g(\cdot)$ are both statistical models, implemented in code of arbitrary complexity (assumed away by the pseudocode). A line of math-oriented pseudocode may quip, *calculate* $f \circ g$. If your distributions are well-behaved and designed to make this work, then this is trivial; if the outputs and inputs are sometimes incompatible—¿Is the range of g exactly identical to the domain of f? ¿Does the dimension of f's input change with the size of an auxiliary data set not mentioned in the pseudocode?—then you'll spend quality time turning the three characters $f \circ g$ into reliable working code.

Second, if you are doing computationally-intensive work, then technique matters. If your simulation runs in two seconds, then that's not an indication that computing

[1] `http://modelingwithdata.org/arch/00000014.htm`

technique is irrelevant and you shouldn't care, but that you should re-run the thing with a hundred times as many elements for a hundred times as long. Your results will be better, and good technique can reduce your run time back down to minutes from hours. Of course, pseudocode assumes away technique.

Third, you can cut and paste real code, but not pseudocode. The first draft of the first chapter had a lot of routines for the reader to cut and paste from the planned online appendix. I'd explain that the GNU Scientific Library, perfect for using in the guts of simulations, doesn't provide anything at the level of a regression, so Figure Eight gives you regression code that you can cut and paste into your project.

And that's where the Apophenia library came from: I eventually got sick of telling the reader to cut and paste snippets, and just bundled the snippets into a package. That freed up the book to include more high-level sample code for cutting and pasting.

I've seen people pick up a copy of my book and immediately feel intimidation that there's code on every page, but I'm still happy with this design decision as well. I take some pride in writing a book that users can directly execute, and that doesn't pretend toward false generality. You know I'm not lying to you or hiding facts from you, because then the code wouldn't run.

Goal 3: Publish the book on paper, bound There are people who write in their books, and there are people who will prop up short table legs with books, but I ain't one of 'em. I don't think of books as newspapers, or as just another prop to be used for whatever practical value it has and then chucked out. Books are a part of the permanent record. If I want my information impermanent and disposable, that's what I've got computer screens for.

Modeling with Data came out in late 2008. It's early 2010 now, and the book, bound and immutable, has remained (almost) entirely in line with the code base. That's taken a lot of work, because the code keeps evolving. Many changes are just additions and new functions or features: means of doing things more easily, which make the code in the book look klunky in comparison but not false, *per se*.

But you can see how my three goals set me up for failure here:

- Write about novel things that are still being actively explored.
- Be concrete, and help the reader with the details.
- Publish something that will be correct in the details as far into the future as possible.

I think these three goals are, as a practical matter, impossible to reconcile, so every author has to give up on (at least) one. It's an interesting exercise to pick up the typical technical text and see what the author chose. The manuals about a specific language, with a spine measured in decimeters, throw the goal of permanence out the window. These book producers see the book as a newspaper or a parts manual, and if this manual takes off, then there'll be a second edition and you'll be expected to use the first edition for mulch. As above, I feel that this is disrespectful of the reader. Others throw out goal 2 and go straight to pseudocode, because by remaining vague they don't have to worry about the vagaries of coding language grammars. This may not even work: I have seen

many pages of pseudocode about how to invert a matrix, but it is effectively obsolete if users have half a line of code in their preferred language that does the whole thing. Finally, there's the option of just playing it safe, and using code that has already been established, tested, and written up in decimeter-thick manuals. Even that eventually fails; maybe have a look at the comments from the author of *Dive into Python*[2], where he confess that his plans for a Book for the Ages didn't mean much as Python 2 evolved into Python 3.

So, there you have my apology, as Apophenia evolves but the pages remain bound, and Goal 3 slowly gives way to Goals 1 and 2. I've set up an Updates to Apophenia[3] page explaining changes to Apophenia that would affect readers of the book. Their effect is of course centered on the sections of the book that make heavy use of the details of Apophenia's implementation (Sections 4.3 and 4.7 on shunting data, Sections 8.2–8.4 when reading model output, Chapter 10 when setting up new models). Much of the book is concrete but covers the not-novel: C, SQL, the GSL, and mathematics comprise most of the book's pages, and have not undergone much revision. Gosh, most readers won't even notice the changes, but this really is the sort of thing that the typical textbook author (i.e. still me) frets about endlessly.

[2] `http://diveintomark.org/archives/2009/03/27/dive-into-history-2009-edition`
[3] `http://modelingwithdata.org/updates.html`