IN THE CURRENT INSTALLMENT OF OUR OCCASIONAL “AT ISSUE” DEPART-EMENT (SEE P. 5), GREG WILSON CONTENDS THAT WE AREN’T PREPARING STUDENTS TO BE “COMPUTATIONALISTS WHO WILL BE ABLE TO CALL THEIR WORK ‘SCIENCE.’”

He bases this contention on several observations he has made over the years and suggests that there’s sufficient blame for this situation to go around the science and engineering community. Significantly, he recommends that a practical step toward a future resolution of this problem might be to go back—not to the past, but to the basics.

I want to draw attention to his arguments in this space, but I won’t challenge them—I leave that task to you, by sending me one of those cards or letters I’m always requesting. The reason I take this issue so seriously has to do with CiSE’s mission. Let me quote here from our Publication Handbook (www.computer.org/portal/pages/cise/content/handbook.xml):

Computing in Science & Engineering’s mission is to support the development of computing tools and methods as well as their effective use in theoretical, computational, and experimental science, engineering, and education.

I regret that this statement lacks one word that Greg’s article has made absolutely clear to me ought to be there. He reminds us that scientific, engineering, and educational work involve practices, and that these can be as different as the cultures of the practitioners. I tip my hat to Wilson for arguing well that computational practices matter.

We all agree and make the effort to maintain scientific and engineering best practices in our laboratories, and our academies train and acculturate students to those practices. But Greg is calling attention to what he sees is a double standard—one applied to experimental work and another applied to computational work. No one would argue that all scientific work shouldn’t be reproducible or that the materials and instruments used shouldn’t have open and verifiable provenance. (Fortuitously, this special issue’s theme is all about provenance and reproducible research; see p. 9.)

So in what sense can I suggest that Greg is making an appeal to go back to basics? You can tell from my picture at the top of this page that I’m no spring chicken. I used punched cards when I started integrating computing in my work and teaching. When we got our first laboratory computers and gave up those cards for paper tape and then floppy disks (I mean really floppy disks—those big, 8-inch leaflets), I had my first programming epiphany with the Pascal programming language. It was highly structured and nearly self-documenting, which facilitated and encouraged good programming practices. I could now produce program codes that were actually understandable a few days later.

It seems to me that Greg is arguing for a return to this type of orderly programming practice, but in terms of today’s modern tools (version tracking, scriptable languages, and sophisticated debuggers) and in light of new workplace realities (collaborative, cross-disciplinary research groups). The abiding and essential reality, in Greg’s words, is that “computational scientists […] are almost always their own customers and […] modify their programs continuously as their questions are answered.”

The next logical question, then, is what are we do-
ing to acculturate our students and prepare them for this workplace? I’ll particularize my response to undergraduate physics curricula because it’s the area with which I’m most familiar. In 2002, the American Institute of Physics released the results of a survey it conducted a few years earlier that examined people who had earned a BS in physics and found themselves in the workplace five years after graduating. (Rachel Ivie and Katie Stowe, *The Early Careers of Physics Bachelors*, Am. Inst. of Physics, AIP Pub. R-433, 2002.) The upshot was that computation was an appreciable component of their work responsibilities but was the one area in which their BS preparation was most deficient.

You would think that this would have set off an alarm throughout the entire physics education community, but it didn’t. In 2005, CiSE sponsored a survey of its own, polling faculty at the 700+ institutions in the US that grant a BS in physics. The disappointing results gave us scant hope that the AIP’s findings would differ if it conducted its previous survey of physics BS graduates today; our full results appear as part of a special issue (R.G. Fuller, “Numerical Computations in US Undergraduate Physics Courses,” *Computing in Science & Eng.*, vol. 8, no. 5, 2006, pp. 16–21).

I draw two conclusions from these results that connect to Greg’s article:

- Essentially none of the extended, staged integration of computation into the full curriculum that Greg feels is essential to developing good computational thinking and practices exists. Most departments offer either a one-semester specialty course in computational methods or a limited computational experience at the introductory level (most frequently in connection with the introductory lab).
- Although many survey respondents expressed strong conviction that computation was a very important part of physics education, the distribution of colleagues in their departments who took this conviction seriously enough to assign computational exercises as part of a course grade was quite narrow. A large fraction of departments reported that only 20 percent of their faculty made such assignments.

So what’s to be done? Clearly, individual faculty members have failed to make serious commitments to include computation into the physics curriculum. This depresses me because my experiences with computing in the undergraduate physics realm—buoyed by all those earlier epiphanies—are 30 years old. When I look at these survey results, I think, with Greg, that we’ve failed to learn much from history. I hear the same old saws about the importance of computation at physics education meetings, but nothing seems to change—it’s déjà vu all over again. But institutions aren’t exempt from blame, either. My personal experience is that part of the malaise is generational. Many traditional physics faculty don’t use computing at all, let alone produce software. Moreover, incoming faculty who might be more disposed to include computation are frequently prisoners of institutional tenure processes that don’t credit instructional innovation.

Nevertheless, the stakes are important enough for physics departments to be involved in any institutional effort to integrate computation because virtually all engineers and non-physics science students take an introductory physics course. If good computational practices are to be taught over an extended period in parallel with the actual use of tools and methods, then we better offer good beginnings to those students we serve and better foundations for work practice to those we graduate as majors.

It’s time to practice what we preach and past the time to get back to the future.