

How Much and What of Computing for Physics Education?

Hans-Joachim Bungartz

Dept. of Informatics, TUM, bungartz@in.tum.de

Abstract

In this contribution, we try to depict how the future undergraduate physics education should take into account the increasing computational flavor in science and engineering, observed in both academia and industry. This flavor, of course, has its roots in mathematical modeling and numerical simulation, but it has got a much broader scope in the last years, including issues of visualization, parallel and distributed processing (up to the Grid), data handling and exploration, and software engineering. Hence, this paper's core statement is that the modern physicist needs more than a mere programming course or some basic numerical knowledge, but a profound education and training in "Advanced Computing" – organized in a way similar to the typical Advanced or Higher Mathematics Courses covering calculus and linear algebra which are close to standard today.

Introduction and Background

Today, Computational Sciences are generally considered to be a – if not *the* – key technology of the future [see (Benioff & Lazowksa, 2005), e.g.]. Actually, on the research side, it is obvious that in many fields of science and engineering gaining new insight is closely related to the use of computer-based methods. Physics (which we will take as our example in the remainder) and mechanics are two prominent representatives for this development. However, on the education side, changes seem to be much slower. Although there are quite a lot of new specialized and tailored programs such as CSE or Computational Physics (especially on the graduate level), there are several evident problems. First, the computational aspect in such programs is often simply covered by focusing on modeling (which, actually, is just theoretical physics and not a really computational feature), by some mere introduction to programming, and by a small selection of extremely physics-oriented numerical topics, typically presented by (computational) physicists within their standard courses – mathematicians and computer scientists stay outside. Hence, second, topics such as distributed processing, visualization, data exploration, or software engineering are more or less ignored and left over to some later "training on the job". Finally, the classical bachelor's programs in physics are rather reluctant to modifications anyway – rigid regulations and the "What to omit instead?" argument are frequently preventing new courses from being introduced. Recently, however, voices also from inside physics [see (Post & Viotta, 2005), e.g.] have pointed out that changes are crucial, that producing reliable simulation software means much more than the mere act of

putting some mathematical model or the numerical algorithm derived from it into code.

What is said in the remainder of this contribution, to some extent, is an external or non-physicist point of view. However, it is based upon year-long experiences in the Computational Sciences education. Since it seems to be a general observation that Computational Physics programs are organized in a more mono-disciplinary (i.e. physics-based) way than CSE education is done (at least in Germany), it might be a good idea to look what lessons have been learned or still are to be learned there when thinking about “computational updates” of undergraduate programs in physics.

Experiences and Observations

In this section, we briefly summarize our experiences with the interdisciplinary education in Computational Sciences and mention some interesting observations often made there.

Classical physics programs

Our group is responsible for the 2-semester course “Introduction to Programming” for beginners in physics at TUM. The idea is to provide a solid basis (Maple plus one programming language such as C, C++, or Java) for later computational activities. Though the students are in their first year and, hence, do not yet have the mathematical background typically required for a numerical course, we have given our course some numerical flavor, i.e. providing more or less an introduction to *numerical* programming. At present, “Introduction to Programming” is the only mandatory part in the physics curriculum imported from the computer science department (as well as from the math department apart from “Higher Mathematics”). Physics students tell us that, after this introduction, they get a bit of numerical education here and there – always integrated into some topical course – but nothing systematic, and that’s why quite a lot of them, later, choose elective courses on scientific computing at our chair.

Computational X programs

Experiences on the graduate level stem from the interdisciplinary master’s programs “Computational Mechanics of Materials and Structures” at Universität Stuttgart, “Computational Mechanics” and “Computational Science and Engineering (CSE)” (CSE, 2001) at TUM, as well as the Bavaria-wide honors program “Bavarian Graduate School of Computational Engineering (BGCE)” (BGCE, 2005). The latter two, for which the author is responsible at present, are hosted by informatics departments, but they are, nevertheless, designed in a really multi-disciplinary way. TUM’s CSE program, e.g., is a joint venture of seven of TUM’s departments (informatics, mathematics, physics, chemistry, civil engineering, mechanical engineering, and electrical engineering). There, we try to focus on the *methodical* part of Computational Sciences or

simulation technology (i.e. numerical programming, parallel and supercomputing, as well as all of the “enabling technologies” such as algorithmics, software engineering, or visualization) without neglecting the *application* part. Actually, students get in contact with two classical fields of application of numerical simulation (physics, chemistry, biosciences, structural mechanics, fluid mechanics, microelectronics, ...), but these are not in the centre of interest. Our applicants typically have a bachelor’s degree in one of these fields of application, and they want to get this additional computational flavor in their education which this paper wants to promote to get also implemented in classical programs – at least to some extent. TUM’s CSE master’s program has been running since 2001 – hence, there is no relevant statistics available yet, but experiences and feedback (from teachers, students, and employers) so far are very positive.

Co-operations

The Garching campus north of Munich does not only host TUM’s departments of mathematics, informatics, physics, chemistry, and mechanical engineering, but also the neutron source research reactor, the physics department of Ludwig-Maximilians-Universität, four Max-Planck-Institutes for physics (astrophysics, extraterrestrial physics, quantum optics, and plasma physics), the European Space Observatory, the nation-wide computing centre of Max-Planck society, and the Leibniz Computing Centre of the Bavarian Academy of Sciences, hosting one of Germany’s three federal supercomputers. Hence, it is quite natural that in such an environment there are a lot of research activities in computational physics. Just to mention a few examples, there are several Grid projects concerning the Virtual Space Observatory, there are Grid activities for the Large Hadron Collider at CERN, and there is an abundant variety of numerical simulation projects (cosmology, accelerator technology, etc.). One of the central messages we receive and learn from these networks and projects is that modern physics needs expertise in fields such as data exploration or Grid technology – classical subdomains of informatics having got this enabling character for (Computational) Physics.

Innovative course modules

Introducing new topics in new or modified lectures is one part of the story, but we also have to think about new types of courses. Most strategic papers dealing with Computational Sciences [see (Post & Viotta, 2005) and (Benioff & Lazowksa, 2005), e.g.] emphasize that the software issue will probably be the bottleneck and, hence, the challenge of the future (replacing or, at least, joining hardware and algorithmics). How to get high-quality software, how to professionalize the design and implementation process, and how to educate people who are sensitive and well prepared for improving the situation? One possible course model we reported in (Bernreuther & Bungartz, 2006) and which turned out to be

both motivating for students and highly effective are the student projects we took over from the Software Engineering curriculum at Universität Stuttgart and we implemented in the BGCE curriculum (BGCE, 2005). There, 4-8 students work together for 6-9 months on designing and implementing a complete piece of simulation software from the scratch. The desired functionality is given, an advisor provides support, and a “customer” checks whether the contract (milestones, deadlines, beta versions etc.) is fulfilled. For the internal organization (giving roles such as the project manager, implementing teamwork, enforcing deadlines, ensuring quality control etc.), help is given, but this all is, primarily, the team’s job. Obviously, the experience of doing it is much more in the focus than the product itself (topics so far have been a Virtual Wind Tunnel, Molecular Dynamics, and Geometric Modeling) is – which is untypical, but which turned out to be very helpful for learning the process of writing simulation software instead of just hacking code. Our experiences with these team projects are very good, but it is, definitely, just one way how to take into account the computational challenges in education in an appropriate way.

Theses on the Why, What, Where, How, and Whither

In this section, we want to directly address the central questions of the symposium. Thus, based on our experiences, a possible way how to adapt bachelor’s programs in physics is depicted. As already mentioned, this has to go beyond the mere fostering of modeling and simulation in physics.

Why changes?

The increasing influence of computing is neither comparable to some new kind of physics which may be postponed to the graduate level, nor is it dealt with seriously by simply saying that smart physicists won’t have any problems in working with a computer if they are good in physics. It, rather, represents a new and rapidly changing way of working in physics – which must be taught as early and as intense as necessary.

What changes?

The changes must be demand-driven – to avoid the errors done with the math education. Physics must provide the list (which, probably, should contain at least a solid training in programming, an introduction to numerical analysis covering classical discretization techniques as well as particle methods or Monte Carlo approaches, visualization, at least fundamentals in data base technology, an introduction to high-performance computing, and – at least as an elective topic – fundamentals of software engineering), but mathematicians and computer scientists must provide the courses.

Where to change?

Everywhere and on all levels. Specialized programs in Computational Physics, e.g., are important and often easier to implement, but they can not replace changes in the standard Physics programs. We need a topical face-lifting, and we need another step beyond the borders of the physics departments (as it was always done with the basic math education). Trans-disciplinarity does not mean only that one discipline starts thinking about topics from outside – there has to be a real co-operation between disciplines and the respective people, in research and education.

Where to head for?

In my feeling, time has come for basic undergraduate courses in “Advanced Computing” (or call it “Advanced Informatics” or ...), comparable to the math education. At Germany’s technical universities, there is a long and successful tradition of a 3-4-semester cycle in “Higher Mathematics”, compulsory for all engineering fields and physics (if students of physics do not share the courses with mathematicians) and with a selection of topics tailored to the respective needs. A comparable cycle “Advanced Computing” for beginners of, say, three courses with lectures, tutorials, and a practical/lab part could be offered by the computer science department and bundle all the aspects identified as crucial for a modern (and that is computer-oriented) physics education. Additionally, a group project comparable to the experiences in (Bernreuther & Bungartz, 2006) on a topic from computational physics (designing and implementing an elementary molecular dynamics package, e.g.) could bridge the gap between those advanced computing technologies taught and the main objective to solve problems from physics. Of course, this would imply that some credits are saved elsewhere, but this is not necessarily dramatic, since the existing courses or modules dealing partially with computing-relevant aspects could be skipped or reduced.

Concluding Remarks

This contribution presented some ideas how to re-shape the undergraduate (computational) physics education in order to take into account the increased importance of computing, and how to design innovative project-style courses to effectively teach more software-related topics. The concepts suggested are mainly based upon experiences made in the Computational Sciences education.

List of references

- Post, D.E., Viotta, L.G. (2005). Computational Science Demands a New Paradigm. *Physics Today* 58 (1), 35-41.
- Benioff, M.R., Lazowksa, E.D. (eds.). *Computational Science: Ensuring America’s Competitiveness* (report of the President’s Information Technology Advisory Committee (PITAC) 2005).

Bernreuther, M., Bungartz, H.-J. (2006). First Experiences with Group Projects in CSE Education. *Computing in Science and Engineering* 8 (4), 16-25.
CSE (2001). <http://www.cse.tum.de>
BGCE (2005). <http://www.bgce.de>