Wlodzisław Duch
Department of Computer Methods,
Nicholas Copernicus University,
duch@phys.uni.torun.pl

CEREBRATIONS ON COMPUTATIONAL SCIENCE

This article describes some ideas on the present state and the future development of computational sciences. What is computational science, whom should we teach and what should we teach? I will try to give some answers to these questions based on many years of experience in teaching, developing algorithms, writing programs and running large-scale calculations in quantum chemistry and molecular physics. I will also write a bit about computational science in Europe in general, and on recent projects at my home institution, the Nicholas Copernicus University in Toruń, Poland.

Computational Science: what is it?

To understand what should we teach we have to answer first the question: what is computational science? In the introduction to the first textbook on the subject, prepared by the *Computational Science Education Project*, we read that "computational science is about using computers to analyze scientific problems". New branches of science are not created but rather emerge from activities of scientists focused on specific problems. Proliferation of journals on computational branches of almost all fields of science proves that eventually these sub-branches will acquire their own identity, as some of them have already done. The usual argument is: since computing becomes the common language of many branches of science we need good courses and curricula in computational science.

This "common language" looks like a reverse trend in the compartmentalization of science. Unification seems to be a temporary phenomena to me. Even in such well-established fields like computational physics or computational chemistry there is a strong tendency towards specialization of numerical methods. This tendency will grow with growing sophistication of computational fields. What is important for computational dentist[1] is quite irrelevant for computational molecular biologist. *Computational chemistry*, at present one of the best established computational fields, started as quantum chemistry 40 years ago. Only in the last decade sophistication of algorithms, computational resources and availability of black-box packages changed quantum into computational chemistry. The first 30 years were dominated by experts developing and testing algorithms. The last 10 years were dominated by users of software packages who do not understand the computational methods and have too much confidence in the results obtained. Computational chemistry, for most people working in this field, is restricted to the training in using one or two large systems of programs. The training rarely includes information on how to use computational resources, concentrating rather on intricacies of required input and understanding of the output. Another computational field that moves in the same direction is *neural networks modeling*. A number of black-box software packages have been created in the past decade and they are consuming a large amount of computer time. This is the future: once the sophisticated **tools** are there the number of users becomes much larger than the number of experts capable of producing and understanding the tools. Computational science, as a general field, will not change the users into experts. Specialization of algorithms in physics, chemistry or biology requires quite specific scientific background, a curriculum that would share only the core subjects with computational science.

There is no doubt that computational science is going to be successful. In the USA there are several driving forces behind the development of computational science. Perhaps the most important one is the move from military to civilian applications in the post-Cold War era. The motivation for the Grand Challenges is at least partially political: supercomputing technologies are perceived as the key to the technological dominance. The

[1] Yes! At a recent computational medicine conference there was a session devoted to computational dentistry.

access to powerful hardware is much easier in the USA than elsewhere. The teraflop computers and the gigabit networks are great goals to light up the imagination. Are the supercomputers really helpful for developing computational science? In the past large-scale computations meant usually struggling with the technical problems, constant updating and rewriting large systems of Fortran programs to keep up with the progress in hardware and systems software. We can happily forget now about JCL and 370 system routines required to change a line in the program stored on disk, but our life is not really simpler: Fortran 90 is much more complex than the older versions, computer architectures are much more complex and the problems we want to solve are more complex. Computing power by itself does not solve the problems. Most development of programs is carried on workstations but also large-scale computations are done on thousands of powerful workstations by a large number of scientists. Therefore educational issues concern many of us, not just the relatively small number of those working on the highest performance machines. Supercomputers are necessary but first one has to learn how to use them. We all know about applications that can consume an infinite amount of time of a fastest machine because they are programmed quite badly.

Higher-education programs in most European countries make it rather difficult to add new, interdisciplinary subjects of study. United Kingdom is the notable exception, most universities there encouraging students to take courses in diverse fields. European problems are purely formal, coming from the fact that the degrees are granted by the faculties and not by the universities. It is a real obstacle in accepting new branches of science. How do you start research in a new field if students can not make a degree because the field is formally not existing? This is the classical problem of the egg and the hen. Computational science and scientific computing are among many new fields of science that face this problem. For example, announcements of the positions in cognitive sciences outside the USA and UK are very rare, not only because of the requirement to teach in local language. The understanding of the potential impact computers will have on the society is greatest in the USA, as is evident in the government sponsored programs. According to a recent report the trend toward working at home accounted for almost half new jobs in the last 5 years and was reflected in the significant decrease in the numbers of suit sales!

The computational science field is still in a very early stage of development. At the beginning of 1994 no Internet newsgroups were devoted to the subject (there is a group *comp.parallel* for parallel systems) and the first gopher service was announced, although all NSF supercomputer centers had some WWW or gopher servers. There were no electronic computational science journals (nor paper versions for that matter). In contrast to this journals on computational physics, chemistry, or computer simulations in general, abound. The only electronic bulletin about high performance computing I have found, HPC-Wire, requires paid subscription. There is also a list on "Supercomputing in Central and Eastern Europe". The field of computational science will slowly gain its identity, distinguishing it from computational physics, chemistry and other sciences, but at this moment it is not yet clear what will it be.

Whom should we teach?

Computational science is interdisciplinary and may be approached from **two distinct angles**. First, computer science and mathematics students, with good background in mathematical modeling, may choose to learn computational science and later specialize in applications of computer methods to different fields of science. This is the computer/mathematics entry point. Second, students of biology, geography, music, arts and other subjects could choose as their specialization computational branches of their particular subject of study. Most programs in computational science are concerned with the first approach, disregarding the problems of further education. The electronic textbook developed within the *computational science education project* shows the fascination with the computer power. It is directed at an expert who knows the field and wants to learn computational methods rather than at a physics or biology students who want to learn something about the real world. I do not see how examples given in this textbook could get a biophysicists excited about the computation of bioelectric fields, but for a computational scientist, whose job is to provide the tools for biophysicist, it is a good introduction. Will the two be able to communicate? Most of us, interested in computational science, come from scientific and engineering culture well trained in Fortran and numerical methods. The Grand Challenge problems are attacked by computational physicist, who are skilled in building mathematical models of physical systems and have many years of experience in high-performance computing. It is relatively easy to create program in computational science for applied mathematicians and physicist. However, if they should also become experts in medicine or biology, courses to teach them the language and ways of thinking in these fields should be created.

Finding interesting problems in science is very important. Will the computational scientist find such problems? Chemists or biologists have many interesting problems but it is frequently not easy to translate these problems into computational models. Quantum chemistry is very well developed, but it has not helped real chemists to understand better the fundamental chemical concepts, such as valence. Computational chemists and experimental chemists still use a very different language. To understand chemical problems one has to learn the language of chemists. Computational science curricula should not aim only at education of scientists. High performance computing is useful for the industry, as is evident from the involvement of the supercomputer centers with industrial companies. Many commercial products are designed using computer methods nowadays, from sport shoes to infant dippers. Are the existing engineering courses sufficient to give an overview of the main ideas and problems to computational scientists?

I see the need for **two different kinds** of computational scientists. First, the numerical expert and computer wizard who has learned the language of a given branch of science and who is working on the forefront of computational technologies. His job is to push computational methods to the limits, use new computer architectures and get the solutions to the most demanding problems. His job is also to create the tools, sophisticated but easy to use, for his colleagues coming from other branches of science. I shall refer to such a person as **computational scientist** or an expert in **scientific computation**. His time is spend to a large degree on solving technical problems with nonstandard technology.

The second kind of computational scientist has quite different background, coming from particular sciences. He or she appreciates the value of computation as the "third force" in science, the means to model the world and learn about nature, but does not like to develop programs in low-level languages, think about speed, efficiency or communication of processes. I have seen biologists developing simulation models using spreadsheets - programs that are very inefficient but easy to create. Such a person should be called **computational biologist**, **geographer** or **physicist**. His knowledge of the subject of his study is deeper than most computational scientists will ever have, but his computer-related abilities are of lower order. He is using standard technology and sophisticated software tools to build his models without much effort. Perhaps one can also call such people sophisticated computer users. There will be many more of them than real computer scientists.

What should we teach?

The educational challenge is to create different programs that would appeal to the computational scientists but also to computational biologist, economist, engineers or medical scientists. These two type of programs should significantly differ.

Most supercomputers, especially new parallel machines, have Fortran compilers, tools, numerical libraries and not much more. Workstations attached to supercomputers run various data visualization packages but the main work involved in numerical solution of the problem is done and will continue to be done with large Fortran programs. Most physicists concentrate on getting results: they write programs as quickly as possible, test their ideas, write a paper and move to other projects. The software created laboriously in languages such as Fortran is frequently unusable after some time even to the author. Large systems of programs were created over the past decades by many researchers and are basically a patchwork, hard to maintain and develop further. At present supercomputer centers use packages for computational chemistry, molecular biology and genetics, fluid dynamics, aerodynamics and some engineering problems. Computational scientist should create software for many users, therefore they should be well trained in Fortran, parallel and vector optimization techniques and methodology of programming. However, in this field there is no substitute for experience.

It is clear what should we teach computational scientists:

- basic mathematics, mathematical modeling, numerical analysis, optimization;
- computer architectures, networking;
- programming in C and Fortran 90, including advanced methodology of abstract data type and object-oriented programming, software engineering techniques, CASE tools;
- vector, parallel and distributed programming techniques;
- mathematical languages for symbolic computations, quick interactive model building and visualization (Maple, Mathematica, Macsyma or Axiom);
- visual programming languages for quick model building (Simula, ACSL, Stella);

- simulation techniques, continuous and discrete, simulation languages (ACSL, Simscript);
- graphics, visualization, animation and imaging techniques;
- elements of artificial intelligence.

Computational scientists are going to provide tools for the rest of the scientific and industrial communities interested in computational methods. They should be taught how to use the sophisticated program packages already running on the supercomputers. They should also be taught basic artificial intelligence techniques and visual programming languages enabling quick solution to simple problems. Elements of artificial intelligence are useful because sophisticated programs without built-in intelligence may be too complex to use: choices of many parameters and methods of solution available in some systems already require a lot of expertise. It may be quite dangerous to put too much confidence in the results, obtained by inexperienced users, of programs that are not able to justify the applicability of the algorithms used. In addition special courses in basic physics, chemistry, material science, fluid and aerodynamics, biology, genetics, neural networks, economics, geographical information systems and other subjects should be offered to help the students to learn the language of different fields and communicate with the experts in those fields. These special courses should give the main ideas of a given field and show how computational methods are used in it.

This is what we should teach the experts who will create specialized tools for Grand Challenge problems. I have not seen any such curricula so far.

What about the second, user-oriented route to computational science? Most of the programs running on supercomputers use rather unique algorithms, and they were created by computational physicists or chemists. To help the sophisticated computer users to create computational models without great effort we need higher level tools. Such tools are coming, although still more often on the personal computer or workstation platforms than on supercomputers. Symbolic algebra languages, such as Maple, Mathematica, Axiom, Macsyma or MATLAB make creation of mathematical models much easier. They are described as languages for mathematical computation and visualization. These languages are available on powerful workstations and some are appearing even on supercomputers. They contain very specialized knowledge related to heuristic methods of symbolic computation. Intelligent software (H. Abelson et. al., Comm. ACM 5/1989; G.H.F. Diercksen and G.G. Hall, Computers in Physics 4/94), or numerical computations coupled with expert system, could make the use of high performance computers significantly easier, but it will take some time before such software will arrive.

We have already stopped teaching most of our undergraduates programming languages in favor of teaching how to use high-level computer software. Educating sophisticated computer users perhaps we should leave the details of numerical solutions of PDE's to computational scientists and concentrate more on scientific problems. Are we going to teach students of biology or economy about the latest hardware developments, superfast computers and networks, and than fall back to such primitive ways of programming these machines as C or Fortran languages? Most students are spoiled by much better programming environments available on personal computers. Some of them have seen visual programming languages, such like Simula, ACSL or Stella. Visual programming languages (icon, form, or diagram-based) allow to specify a dynamical model in form of icons connected on the screen. Stella provides a visual programming language (on Macs or workstations) for building dynamic animated models in form of diagrams. AVS or Khoros are examples of such visual programming languages designed for scientific visualization of data. There are many problems which can quickly be solved using such languages. Computational scientist do not have to be always involved in such details as the choice of the methods of numerical integration.

Europe and Poland

Europe is significantly behind USA in computational science. Although there are many projects of European Community concerned with information technology, computer hardware and simulations it is enough to look into the Internet to find out that very little information on these programs is available in the electronic form. There is nothing like High Performance Computing and Communications Program, although there are organizations, like RARE, pushing for high-speed networking for research, so one may hope that Europe will eventually follow American example. It is much harder to organize things having 41 full country members in EARN (European Academic Research Network) and about the same number of different national alphabets with special characters. NSF supercomputing centers play a leading role in defining the identity of computational science. In Europe hardware for high-performance computing is available at many universities and other academic centers, including a few national supercomputing centers. Unfortunately these centers have no broad vision aimed at integrating computational scientists of different provenance.

Schools and conferences in computational physics are organized quite frequently by the European Physical Society (EPS) and by various National Physical Societies. EPS has an interdivisional Computational Physics Group, promoting European collaboration in computational physics (EPS is accessible via Mosaic or Gopher at the www.nikhef.nl address). Physicists are the driving force behind the high-performance computing and networking programs. World-wide-web project was started at CERN, European Nuclear Research Facility.

Poland has made a significant effort in 1993 to improve the availability of high performance computational resources. Three supercomputer centers were started, at the University of Warsaw, Kraków and Poznań. The hardware was bought from Cray (YMP/EL98) and Convex (C-3820). A number of older Convex machines are in use in academic centers since several years[2]. Polish Ministry of Education and the Committee for Scientific Research (CSR) became involved in negotiations of software licenses for scientific needs. Software for workstations for visualization, molecular biology and computational physics and chemistry has been bought and offered at high discounts to the CSR grantholders. Negotiations for license on high-level languages such as Maple and MATLAB are under way.

Polish school of **mathematics** has been prominent already at the beginning of this century. Monte Carlo method has been invented by Stanisław Ulam, a mathematician from Lvov who escaped to the USA. Theoretical orientation of mathematics departments, some of them at the top world-class level, does not, however, contribute to the development of computational science. On the contrary, there is a large inertia in mathematical community: theoreticians obviously want to train more theoreticians. The government science policy is to support those fields of science that already have high standard, meaning there should be more of the same.

Nicholas Copernicus University (NCU): attempts to create computational science

At the Nicholas Copernicus University in Toruń (over 12.000 students, 9 faculties, more than 1000 academic teachers) we have made some efforts in the past few years to introduce computational science subjects at different levels of teaching. Most fields of study take 5 years and, as a rule, end with the master thesis. A small number of students continue with Ph.D. programs which takes another 4 years. This system is similar to German and other European systems.

NCU has an older Convex machine (C-120), an IBM mainframe(IBM-4381) that is connected via 64kb satellite link to the Internet and operates its Gopher (and soon WWW) information center (gopher vm.cc.uni.torun.pl). A multi-gigabyte data server will be set up in 1994 and a fiber network backbone is under construction (unfortunately parts of the campus are spread around the city). NCU has good chances to host a supercomputer center for northern part of Poland in the near future.

Recognizing the need to teach computational science we have proceeded in two directions. In 1989 a new field of study, called **informatics**, was started within the mathematics department. Scientists teaching there have made serious efforts to reorient some of their courses towards computer science. The number of freshmens is limited to about 80 due to the current limitations of staff and our technical facilities. We have tried to ensure that the curriculum is as close as possible to computer science courses at other Polish universities. However, informatics was started with the understanding that some of the students will specialize in **computational science**. I am very much afraid that this **direct attempt** to create a route to computational science has **failed**. Why?

High-school graduates associate "informatics" with computers and end up studying things they are not interested in. Only a few exceptional students are taking courses at other departments. Our mathematicians were always theory-oriented and see no need of "degrading" to the level of applied science or encouraging the students to take lectures in computational science, or even artificial intelligence or neural networks. In short, mathematics institute became self-sufficient and created theoretical computer science. May be in the next few years some changes to the curricula will be made - we are still working on it, but at the moment I do not see any forces

[2] In 1992 we were still getting inspections from the USA government agencies, checking if the Spark 1+ stations were used according to their license agreement. COCOM regulations were a great nuisance - for a while even the sophisticated sawing machines were banned from export to the East-block countries.

within mathematics pressing in the computational science direction. Computer science for most people associates with "computers" and "programming" while computational science is still unknown. Universities in Poland are still not too receptive to the job market needs. Are computational scientists needed outside of academia? Perhaps in the USA. Some international companies moved now to Central Europe. AT&T has opened a large software laboratory near Toruń but they seem to be interested in a typical computer science graduates.

What about indirect routes to computer science? In 1988, parallel to the work on computer science curriculum, we have established a *Department of Computer Methods* at NCU, to teach basic computer science in all departments except that of mathematics and physics, where the local staff was able to undertake these responsibilities. This new department collaborates with and is trying to encourage economics, biology, chemistry and other related sciences to develop stronger computational branches. When we started only a few educational initiatives in the area of scientific computing were known (actually the first I've heard about was G. Fox, Computers in Physics 1 (1990) 112) but with the availability of cheap computer power it seemed that there will be a rapidly growing demand from business and industry for this type of university training. A number of projects loosely related to the main goal of developing computational science were undertaken. In 1989 the Institute of Physics started a unique 5-year program of study in the physical foundations of microelectronics. This program relies very much on advanced computer methods and computer laboratories. Several postgraduate one-year retraining courses have been organized to teach business, administration staff and school teachers basic computer science and the applications of computer methods. In the *Institute of Mathematics* and the *Institute of Physics* there are three types of courses for students training to be teachers at different levels of education. Perhaps the most interesting step toward computational science was made in 1988 when we have created a specialization called Computer Physics.

• *Computer Physics*

In 1989 a group of physicist and quantum chemists started a new specialization, computer physics [3]. It starts after the third year of studying physics, takes 2 years to complete and leads to the MSc degree. The program of this specialization is supervised and partially taught by the staff of the Department of Computer Methods. In a total of about 700 hours of lectures, seminars and laboratories students learn:

- Advanced quantum mechanics (1 semester)
- Computational methods of quantum mechanics (1 semester)
- Numerical analysis (2 semesters)
- Partial differential equations, computational aspects (1 semester)
- Seminar: "Theoretical Physics on a Computer" (1 semester)
- Introduction to computer networks and Unix system (1 semester)
- Introduction to advanced mathematical systems (Reduce, Maple, MathCAD) (1 semester)
- Advanced Fortran programming (curriculum for all physicists includes 3 semesters of programming, mostly in Pascal) (1 semester)
- Introduction to artificial intelligence (1 semester)
- Neural networks (2 semesters)
- Computers in physics teaching, laboratory (1 semester)
- Various monographical lectures (4 semesters)
- Applications of computers in physics (2 semesters)

"Applications of computers in physics" course is rather unique since it is taught by 10 staff members, each taking about his or hers field of expertise. It includes also experimentalists, who need experts in scientific data processing, imaging techniques and visualizations to help them in their work. Some courses are a combination of lectures and computer laboratory projects. This specialization is fairly popular among students - in the last year more than a half of all students selected it (other specializations are: theoretical physics; experimental physics; applied and technical physics; lasers and optoelectronics). Master thesis of "Computer Physics" students are mostly in computational molecular, atomic and solid state physics, quantum chemistry, computer science (numerical compression techniques, visualization), neural networks, some MSc projects were also performed in experimental physics groups, on the interface of hardware and software, and some in the educational applications of computers.

---

[3] At that time introducing a new specialization required the approval of the Ministry of Education; that was a real work!

After a few years of running the computer physics specialization we have decided to create similar specializations in other branches of science and integrate the core of these specializations into computational science. However, the staff members outside of physics were really not prepared for such an ambitious task.

• *Tempus Project "Computer Aided Education"*

In 1992 we have proposed to the European community CEC TEMPUS office a three-year project in collaboration with Cambridge University, Cambridge, UK; The University of Leeds, Leeds, UK; Paul Sabatier University, Toulouse, France; Universite de Reims Champagne-Ardenne, France, Technical University of Munich, Germany and the Max Planck Institute of Astrophysics, Munich, Germany. The project is aimed at development of new, computationally oriented branches of science at the Nicholas Copernicus University. The project was approved and started in fall 1992 with a budget of about 250.000 US $ (slightly higher in the second year). It has a rather broad scope: retraining NCU staff members at the participating institutions, exchanging students, organizing courses conducted by scientists visiting NCU, developing our own new computationally oriented courses, developing computer packages for educational purposes, improving and building new computer laboratories.

As a result of this activity this year we should start computational chemistry specialization, a number of new courses were created in geography (mostly climatology, meteorology but also geographical information systems), biology, economy, the Institute of Education started postdiploma specialization "Computers in Education". Talks are under way to start "Computers in Medicine". The *Department of Computer Methods* supported by other educational units at our University should be expanded in the near future (probably 1995) into a *Center of Computational Sciences*.

There are of course serious problems facing us, connected not only with the poor economical situation but also with the rapid private industry growth. Many scientists left academic positions in favor of much better paid industry jobs. It is hard indeed to find competent young people willing to stay in research or education. In the long run, however, we hope that our approach to developing computational science will also be successful.