The School of Computational Science at Florida State University

Gordon Erlebacher and Janet Peterson

Florida State University, Tallahassee, FL 32312
erlebach@csit.fsu.edu
http://www.scs.fsu.edu/~erlebach

Abstract. Florida State University has recently launched a new program in Computational Science. In this paper, we describe the need for computational science, how we defined computational science, and explain the rational behind our computational science curriculum.

1 Introduction

Over the last few decades, computations have joined theory and experimentation to form the three pillars of scientific discovery and technological design. Moreover, in many cases scientific computations have superseded both theory and experimentation. Thus, whether one is studying subatomic particles or galaxies, whether one is designing minute nano-composites or huge skyscrapers, whether one is sequencing the human genome or protecting fragile ecosystems, whether one is studying the flow of blood in capillaries or predicting the winds in a hurricane, computations play a central role. The computations that enable these and a myriad of other studies depend on the invention, implementation, and testing of algorithms and software that computers use to solve scientific and engineering problems. This is the work of computational scientists.

The public has little awareness of the important and ubiquitous role of computational science in their everyday lives. Especially here in Florida, people hear on the Weather Channel that computer models predict that hurricane X will make landfall at city Y at time Z as a category W storm but they are completely unaware that CAT (computer aided tomography) and MRI (magnetic resonance imaging) scans are completely reliant on the contributions of computational scientists that allow for the transformation of measurements taken by the scanning device into images that a radiologist can read and interpret. Nor do they realize as they look out the window of a Boeing 777 that the wing they see was designed using algorithms developed by computational scientists that predict both its aerodynamical and structural behavior, with experiments used just to ensure the computations were right. Nor do they know that large retail companies such as Walmart and Target use sophisticated computational algorithms to manage the distribution and delivery of goods to their many stores in an efficient manner so that costs are kept down. In these and countless other settings, computational scientists are indispensable contributors.

The high national priority of computational science is amply illustrated by the fact that the President's Information Technology Advisory Committee (PITAC) in 2003 chose it, along with health care information technology and cyber security, as the three areas of greatest national importance related to information technology. The Committee's June 2005 report on Computational Science [2] states that the most scientifically important and economically promising research frontiers in the 21st century will be conquered by those most skilled with advanced computing technologies and computational science applications.

Based on this and earlier trends, Florida State University has established the School of Computational Science with the following missions:

1. Develop innovative interdisciplinary graduate training programs in scientific computing and its applications;
2. Foster research in scientific computing and its applications in a variety of disciplines;
3. Provide a supportive environment for high performance computing on the providing the best possible education and training in computational science should be an item of high priority at any research university.

In this paper, we concentrate on the first mission.

2 What Is Computational Science

There is no accepted definition of computational science. Many institutions equate computational science with computational frameworks, problem solving environments, etc. While that is certainly viable, we define it somewhat differently. At Florida State University (FSU), we state that Computational Science lies at the intersection of Mathematics, Computer Science, Applied Science and Engineering. Thus, the common themes that thread all these areas constitutes computational science (see Figure 1). Examples of such commonalities include algorithm development, multiscale techniques, transform methods, an appreciation for frameworks and problem solving environments, scientific visualization, stochastic methods, optimization, and others. Each of these subject areas encompass techniques that are used in almost all applied areas of science. Advances in multiscale algorithms and theory will further the development of the physics of turbulence, improve understanding of geological and climate processes, lead to better modeling of biological processes in the human body, etc. This definition is consistent with the following quote from the PITAC report[2]: "While it is itself a discipline, computational science serves to advance all of science."

Many of the important problems facing society today can only be solved by teams of individuals from several disciplines. Among these teams, one or more computational scientists, according to our definition, will serve as the glue that will help members from these different teams communicate with each other, translate each others languages, and facilitate the transfer of technology, whether algorithmic or deliverables.

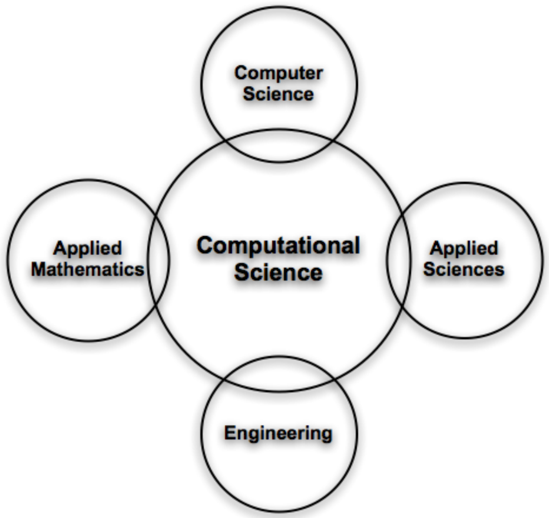

Fig. 1. The interdisciplinary nature of computational science

3 Multi- and Inter-disciplinary Research

In the past, the inter- and multi- disciplinary nature of computational science has been largely ignored in the training of computational scientists. That training has been mono-disciplinary in nature, with students largely confined to courses and research within just one of the many disciplines that intersect with the world of computational science. Thus, despite the fact that the need to improve the training of computational scientists is universally recognized and despite the fact that numerous reports (see references within [2]) have stated that the United States is facing a national crisis due to the its diminishing leadership role in computational science, not much has been done to improve the situation. A list of existing graduate programs in computational science is available at [1].

The Computational Science program at Florida State University is committed to improving the training of future computational scientists at the Ph.D. level through an innovative curriculum with courses designed to function across disciplines rather than *within* a single discipline.

Computational methods for solving problems in science and engineering are ubiquitous within the University. The vast majority of the natural and social science and engineering departments, centers, schools and institutes in the University include faculty, postdoctoral associates, and students who use computational methods in their research. Moreover, several non-scientific departments such as the film, dance and business departments have a need for generic tools, e.g., for the creation of animated films, for choreographing dances, for marketing and finance decision making, etc. Even the athletic department makes use of (although does not develop) advanced software with feature detection algorithms to extract important information used to help improve their athletes.

The University also has considerable strengths in those aspects of computational science that involve the invention, development, and testing of new computational methods. Faculty engaging in such activities are scattered throughout the campus, certainly including many mathematical and physical science departments and engineering departments. The research of the SCS is primarily devoted to these activities as well. Furthermore, the SCS, by its very mission, seeks to exploit, explore, and develop, with faculty and students both within the SCS and throughout the University, interdisciplinary connections that synergistically can enhance algorithmic development and deployment.

These different research programs often develop algorithms in isolation, although they might have strong overlaps in terms of concepts and even implementation. Working within a interdisciplinary framework can lead to algorithms that are more general and useful to a wider range of disciplines. The training provided to our future computational scientists must therefore impart the knowledge in those areas of science that are common most of the ongoing research activies in Applied Sciences, Engineering, and even the social sciences.

Programs at other universities also try to emphasize the interdisciplinary nature of computational science. For example, the Institute for Computational Engineering and Sciences (ICES) at the University of Texas at Austin is a consortium of research centers and groups that includes nine university based centers and programs ranging from Computational Materials to Computational Finance. George Mason University has a School of Computational Sciences organized into several semi-autonomous units with their own faculty and chairs. In both cases, the idea is to foster an environment that invites students to develop graduate students to do develop skills across multiple disciplines.

4 SCS Program

The stated goal of the Ph.D. program in Computational Science is to train of graduate students in the art of computational science and provide them with the opportunity to acquire expertise in a particular area of applied science or engineering. Thus the degree provides the student with breadth as well as depth. Graduates should be able to successfully collaborate with scientists in other disciplines. Ideally, students should learn to develop and analyze new computational procedures for use in a variety of fields.

The identifying characteristic of a Ph.D. degree is the ability of its graduates to perform independent research in a selected field rather than just accumulating credit hours. Although, there is much to learn, and some feel that more coursework is indicated, we have chosen to impose a number of credit hours consistent with the University's residence requirements. As a result, our primary emphasis is breadth across topics rather than depth in a given scientific discipline.

Students entering our degree program have a strong desire to do computational science. We broadly classify the students into two groups. The first group is more interested in developing and analyzing scientific algorithms for problems arising in a wide range of application areas, developing techniques for managing large volumes of digital information regardless of the application area, or perhaps in visualizing their data from areas as vastly different as medical imaging or weather prediction. The second group is interested in similar goals except that the students have already identified a science or engineering area of interest, in which they want to concentrate their research.

We offer two computational tracks within the SCS Ph.D. program: 1) a major (computational) track for students of the first group, and 2) computational tracks with a specialization within one of the disciplines from either the applied sciences or engineering for students in the second group. All tracks have the same number of course hours and the required (core) courses are identical. A specialization degree requires that the student register for a minimum of 9 credit hours in the specific area of specialization. Students in the first group are free to choose credits in more than one applied field. Based on the current expertise at the SCS, we currently offer major tracks in Computational Science, and Computational Science tracks with specialties in the areas of Atmospheric Science, Biochemistry, Biological Science, Geological Science, Materials Science, and Physics.

4.1 Curriculum

Since computational science is an interdisciplinary program, programs of study can be quite varied. Without a flexible Ph.D. program, it will be difficult to adapt to the fast-paced change in the scientific research community. For example, opportunities will surely exist in the future to combine nano-technology or material science with biological sciences, or imaging with structural biology. With the flexibility built into our program, it becomes possible to respond to these trends in a rapid and flexible manner. This translates directly into a robust set of course requirements, to be determined on an individual basis by the curriculum committee and the advisory committee of the student. The former has knowledge of the large course base available and can offer advise to the student and his advisory committee, while the later has detailed knowledge of the student goals and can offer advice and how to achieve them.

Every student must take three core courses, for a total of 11 credit hours (see next section). These courses are designed to cut across disciplines. A second set of courses, called core electives, provide training in computational science, in more detail. Students choose three courses among the electives. Three examples are scientific visualization, numerical solution of partial differential equations, numerical methods for earth and environmental sciences. Finally, 9 additional credit hours are selected from existing departmental courses in computer science, engineering, mathematics or an applied science. The department is chosen based on the stated specialization of the student (second group). There are a total of 29 credit hours of coursework in addition to a minimum of 9 credits of dissertation hours. Additional credit hours may be obtained through dissertation hours, or coursework, under the guidance of the student's supervisory committee.

4.2 The backbone of SCS: The Core Courses

The required core elements in our program are three new courses that provide the student with the programming skills, tools and algorithms necessary to tackle a wide range of real world problems. After completion of these courses, students are expected to have mastered the course content, and more importantly, understand the inter-relationships between the various concepts and feel comfortable utilizing the tools discussed. The goal we seek to achieve is to have students develop the instinct to choose the appropriate computational methods when faced with a new task.

The primary course on which the others depend is *Scientific Programming*, offered for the first time in Fall 2005. Programming languages form the backbone of virtually all areas of scientific research involving computers. Although many students have an elementary knowledge of a single compiled language such as C/C++ or Fortran, or an interpreted language such as Matlab, this knowledge is not sufficient for the development of large-scale scientific codes. Indeed, these programs are usually strewn over multiple files, multiple languages, and multiple operating systems. Students are confronted with these realities from the onset of the course. In addition, they are exposed to benchmarking, profiling and documentation. These concepts are integrated into the language concepts from day one and must be used in all their projects. The hope is that by the time the students are involved in their own research projects, these skills will be integrated into their work habits during program development, rather than as an afterthought. This course covers the common elements of Fortran 90, C++ and Java, three languages of widespread use in scientific computing. The students are taught to solve problems at the conceptual level using mainstream object-oriented programming ideas. The course has been well received with fourteen students evenly distributed across four departments (Mechanical Engineering, Physics, Mathematics, and Meteorology). SCS is planning to offer the course on an ongoing basis, both as a service course to science students in general and as a required course to the students enrolled in our program.

The remaining two required courses, Applied Computational Science I & II, are innovative new courses that combine classwork (4 hours) with a weekly computational lab (4 hours). We have already established that researchers in an interdisciplinary environment must possess a working knowledge of a wide variety of tools and algorithms. Even more important is the researcher's to combine these tools and understand their inter-relationships. These tools span the domains of computer science, mathematics and applied science. A major goal behind these courses is to foster a frame of mind where the student's first instinct is to use tools in combination rather than in isolation. We hope to achieve this through a combination of more conventional lectures, in which algorithms, software, tools are described and demonstrated, and laboratory work where the students get the chance to try the tools out and apply them to practical problems. They will also be required to write the software for some simple tools.

Achieving this objective necessitates a delicate balance between theory and practice. In the theory section of the course, the students are exposed to a wide range of algorithms in fields the span differential equations, visualization, stochastic methods, imaging, optimization, eigenvalues, clustering, etc. While not describing each of these fields in every detail detail (after all, each of the aforementionned fields is easily a course onto itself, which in many cases can be taken as part of course electives), representative algorithms are described, pseudo-code is given, and the students are expected to generate working code that executes the algorithm. The novel part of the course is that whatever the project the student is working on, the results generated are analyzed/visualized, or further processed with other tools (visualization, clustering, etc.). These "other" tools are provided to the student (either obtained from the web or developed by students in the courses given in previous years). Naturally, code benchmarking and profiling are mandatory, regardless of the project underway. Some of the helper tools might themselves be developed from first principles during later sections of the course. Thus, to run a PDE solver, the student might be required to first generate a grid with some existing grid generation package. The lab work provides an environment to put theory into practice under the guidance of the instructors. This is where coding, debugging, and tool exploration happens. The lab exercises are designed to expand knowledge of many existing tools. In a fast changing world, it is unrealistic to teach students to use individual tools. More importantly, students must learn to adapt to new tools, learn how to use tools in isolation, and most importantly with each other. The course will teach students to understand the tools, construct the tools, and use the tools. In addition, they will learn to explore the web to find new tools knowing only some of the required specifications.

4.3 Support Courses: Electives

Elective courses are existing departmental courses and courses under development that relate to computational science, and serve to complement the core courses and provide more in depth of computational science concepts. Currently these courses are Computational Biophysics I&II, Computational Evolutionary Biology, Computational Finite Element Methods, Introduction to Bioinformatics, Molecular Dynamics, Monte Carlo/Markov Chain Simulations, Numerical Linear Algebra I&II, Numerical Methods for Earth and Environmental Sciences, Parallel Programming, Algorithms and Architectures, and Survey of Methods for Numerical PDEs. Additional courses, on topics that cut across disciplines will be added as needed. Examples might include Grid Generation, Multi-Scale Analysis, Transform Methods, Wavelet Algorithms, Applied Database Techniques, and more.

5 What Is Different?

Within any new degree program (and ours is no exception), the question arises: why could a student enrolled in our degree program not obtain essentially the same degree from an existing department. The reason is simple: computational science cuts across departments, concentrating on the sub-disciplines that are common to all: programming, algorithm development, visualization, statistics, just to give three examples. Providing students with knowledge in these various disciplines generates breadth rather than breadth. Both types of scientists are in demand in today's world. The computational scientist serves as a buffer between domain experts who have most of their knowledge in a single field. he also serves as a conduit to transfer technology from one discipline to another. Without our program, a student in a science or engineering degree program is only trained in computational methods appropriate to their chosen discipline. In the SCS, students are more involved with algorithm development across disciplines, rather than with the use of an existing numerical method to solve an outstanding problem in some applied area. While computations do play a large role in the training of a least some students in the Mathematics or Computer Science departments, algorithm development usually plays a minor role. It is the application itself that drives the research. In our school however, it is the need for algorithms that are applicable *across* disciplines that is the driver. Nonetheless, we recognize the need to expose students to one or more applied disciplines to gain an appreciation for the algorithms being developed: to keep a foot on the ground rather than get lost in abstractions. The SCS is not in competition with existing programs. Rather, the objective is to train students in the emerging discipline of computational science, which combines aspects of computational mathematics, computer science, computational statistics as well as application areas in science and engineering.

6 Conclusions

We aim to train students for careers that focus on interdisciplinary research on the invention, development, and testing of new computational methods. Thus, it builds on the existing strengths, found within narrower disciplinary avenues, in the use and development of such methods. The program also recognizes the truly interdisciplinary nature of computational science and the need to train future generations of computational scientists in ways that take advantage of that nature. Our program does not intend to replace the training of computational scientists in a specialized discipline within a department. Rather, it aims to offer students a novel and needed type of training with a broadness that they cannot provide. Instead the departments inculcate any additinal specialization not provided by the SCS.

References

1. Computational Science Graduate Programs. http://www.cs.mun.ca/~wlodek/comp-sci-links.html .
2. PITAC Report to the President on Computational Science: Ensuring America's Competitiveness (June 2005). http://www.nitrd.gov/pitac/reports/20050609_ computational/computational.pdf .